PREDIQT: A Method for Model-based Prediction of Impacts of Architectural Design Changes on System Quality

Submitted to the Faculty of Mathematics and Natural Sciences at the University of Oslo in partial fulfillment of the requirements for the degree Philosophiae Doctor (Ph.D.) in Computer Science

August 2011
Abstract

The increasing criticality of the ICT systems and the steady growth of their mutual collaboration impose new challenges with respect to adapting system architecture to new needs while preserving the required system quality. Architectural adaptations are inevitable for accommodating the systems to the new services, processes, technologies, standards, or users. However, due to criticality of the systems involved, planning, implementation, testing and deployment of changes can not involve downtime or similar degradation of quality of service. Instead, the systems have to quickly and frequently adapt at runtime, while maintaining the required quality of service. Architectural adaptations which realize the new collaborations often also have to provide conditions for compatible quality levels across systems. We argue that quality has to be predictable in order for systems in collaboration to be trustworthy. There are often trade-offs between quality characteristics. Moreover, one quality goal may be possible to achieve through several different architectural designs. Before adapting a system to changes, there is therefore a need for model-based decision making support which facilitates understanding of the effects of architectural design changes on system quality. Abstract models could then be used to make informed decisions about possible alternatives to architectural design and their effects on quality. By using model-based analysis, needs for testing environments or systems replica, as well as downtime resulting from premature change deployment, may be reduced.

This thesis proposes a method, called PREDIQT, for model-based prediction of the effects of architectural design changes on the quality characteristics of a system. The PREDIQT method aims at establishing the right balance between practical applicability of the process and usefulness of the predictions. The PREDIQT method makes use of models that capture: 1) the architectural design of the target system, 2) the quality notions of relevance for the target system, and 3) the interplay between architectural design and quality characteristics of the target system. The model-based predictions result in propagation paths and the modified values of the parameters which express the quality characteristic fulfillment at the different abstraction levels.

As part of the evaluation of the PREDIQT method, we have conducted two industrial case studies performed on real-life systems. Both case studies have given strong indications of feasibility of PREDIQT. The experiences and results obtained indicate that the PREDIQT method can be carried out within acceptable resources, on a real-life system and result in useful predictions. Furthermore, the observations indicate that the method, particularly its process, facilitates understanding of the system architecture and its quality characteristics, and contributes to structured knowledge management through system modeling. The preliminary evaluation also indicates cost-effectiveness of the method.
Acknowledgments

This work has been conducted as a part of the DIGIT (180052/S10) project funded by the Research Council of Norway, as well as a part of the NESSoS network of excellence funded by the European Commission within the 7th Framework Programme.

I owe my deepest gratitude to my main supervisor Ketil Stølen for all his encouragement and involvement, as well as for sharing his extensive knowledge and passion for science. Ketil has always and at any time been available, helpful, ready to discuss research or just have a spontaneous chat. It has been an enriching experience, a true pleasure and a privilege to be a doctoral student of Ketil Stølen. I sincerely thank him for the four years of PhD studies during which, thanks to him and the motivating research environment he has provided, I have developed personally and professionally.

I also wish to express my gratitude to my second supervisor Birger Møller-Pedersen, for all the valuable feedback and support received through his active participation at the many DIGIT project activities.

I also sincerely thank the guest scientists in the DIGIT project: Fabio Massacci, Christian Damsgaard Jensen, Isabelle Simplot-Ryl, Audun Josang and late William Winsborough. I am also thankful to the members of the DIGIT advisory board. Both the guest scientists and the advisory board members have, through the regular meetings, provided useful feedback and advice.

I have during the PhD studies been employed by SINTEF ICT as a research fellow at the Department of Networked Systems and Services. I owe my thanks to the research director Bjørn Skjellaug and all the colleagues for providing a very pleasant and motivating working environment.

I have at many occasions received useful feedback and encouragement from Amela Karahasanovic, who I have also co-authored a paper with. Many thanks to Amela for the joyful collaboration and the advice.

I have also co-authored papers with Atle Refsdal, Bjørnar Solhaug, Jon Ølnes, Per Myrseth, Håvard Grindheim and Anette Andresen. I wish to express my gratitude to all of them for their collaboration and support.

I have also at a number of occasions received useful feedback on presentations of my research and some of the written material from Olav Ligaarden, Erik Gösta Nilsson, Gyrd Brændeland, Neil Loughran, Andre Hauge, Fredrik Seehusen, Mass Soldal Lund and Tormod Vaksvik Håvaldsrud. Thanks for the insights you have provided and for being among the many great colleagues who I have been so lucky to collaborate and socialize with.

Thanks to the many members of the administrative and academic staff at the University of Oslo. They have happily and quickly provided help and advice whenever I indicated the need for assistance.

I would especially like to thank all my dear friends for making my life balanced, for
the patience, and for all the joyful time we spend together.

Most of all I would like to thank my mom, my brother and his wife, my boyfriend and the relatives for their never-ending and unconditional support, encouragement, patience and at times sacrifice. Thanks to my late and deeply missed father for having established the roots for my openness, courage and curiosity since the early days.
List of original publications


2. Aida Omerovic and Ketil Stølen. A Practical Approach to Uncertainty Handling and Estimate Acquisition in Model-based Prediction of System Quality. Accepted for publication in the International Journal on Advances in Systems and Measurements. The first version of this paper received a best paper award at the 2nd International Conference on Advances in System Simulation (SIMUL’2010)


The publications 1 - 5 are available as Chapters 9 - 13 in Part II of this thesis. In the case of publications 1, 4, and 5, we have included the full technical reports which are extended, slightly revised versions of the published papers.
List of original publications
# Contents

Abstract iii

Acknowledgments v

List of original publications vii

Contents ix

List of figures xiii

## I Overview

### 1 Introduction

1.1 Objective ........................................ 6

1.2 Contribution ...................................... 7

1.2.1 The process of the PREDIQT method ............... 7

1.2.2 The approach to modeling in PREDIQT ............... 8

1.2.3 The approach to uncertainty handling in PREDIQT ... 9

1.2.4 The approach to traceability handling in PREDIQT ... 9

1.3 Structure of the thesis ............................ 10

## 2 Problem characterization

2.1 Conceptual clarification .......................... 13

2.1.1 System ........................................ 13

2.1.2 System architecture .......................... 13

2.1.3 Architectural design change ...................... 14

2.1.4 System quality ................................ 15

2.1.5 Quality characteristic .......................... 15

2.1.6 Model .......................................... 15

2.1.7 Quality models ................................ 16

2.1.8 Prediction ..................................... 17

2.1.9 Model-based prediction ........................ 17

2.2 Success criteria .................................. 17

2.2.1 The process of the PREDIQT method ............... 18

2.2.2 The approach to modeling in PREDIQT ............. 19

2.2.3 The approach to uncertainty handling in PREDIQT ... 21

2.2.4 The approach to traceability handling in PREDIQT ... 23
7 Discussion 81
  7.1 Fulfillment of the success criteria 81
  7.1.1 The process of the PREDIQT method 81
  7.1.2 The approach to modeling in PREDIQT 83
  7.1.3 The approach to uncertainty handling in PREDIQT 85
  7.1.4 The approach to traceability handling in PREDIQT 87
  7.2 How our contributions relate to state of the art 89

8 Conclusion 93
  8.1 Developed artifacts 93
  8.2 Evaluation of artifacts 94
  8.3 Directions for future work 95

Bibliography 99

II Research Papers 109

9 Paper 1: A Feasibility Study in Model-based Prediction of Impact of Changes on System Quality 111

10 Paper 2: A Practical Approach to Uncertainty Handling and Estimate Acquisition in Model-based Prediction of System Quality 191

11 Paper 3: Uncertainty Handling in Weighted Dependency Trees – A Systematic Literature Review 211

12 Paper 4: Evaluation of Experiences from Applying the PREDIQT Method in an Industrial Case Study 253

13 Paper 5: Traceability Handling in Model-based Prediction of System Quality 279
# List of Figures

1.1 The topic of the thesis ........................................... 5

3.1 Method for technology research – main steps (adopted from Solheim and Stølen [113]) ......................................................... 29

3.2 Strategies for testing (adopted from McGrath [85]) ................. 30

3.3 Research method for development of artifacts (illustration adopted from Refsdal [104] and modified) .............................................. 33

3.4 Main stages of the research method applied on the second case study . 35

5.1 The four contributions and their relationship .......................... 58

5.2 A simplified overview of the process of the PREDIQT method ...... 59

5.3 An overview of the elements of the prediction models, expressed as a UML class diagram .................................................... 62

5.4 Excerpt of an example DV with fictitious values ....................... 63

5.5 Use cases for the DV tool ............................................. 65

5.6 Excerpt of a DV with intervals and confidence level ................. 68

5.7 A meta model for trace-link information, expressed as a UML class diagram 72

5.8 Main capabilities of the traceability approach, expressed as a feature diagram ................................................................. 73

5.9 Use cases for the traceability tool ........................................ 74

5.10 A screen shot of an extract of a trace-link report from the prototype traceability tool .......................................................... 76
Part I
Overview
Chapter 1

Introduction

ICT systems are involved in environments which are constantly evolving due to changes in technologies, standards, users, business processes, requirements, or the ways systems are used. Both the systems and their operational environments frequently change over time and are shared. The systems are increasingly heterogeneous, critical and involved in complex and dynamic collaborations. A specific service that is consumed by an end-user may be hosted by several providers. Moreover, the service may be recomposed several times over a given period, in order to enable one or more functionalities.

The new needs are often difficult to foresee, as their occurrence and system life time are insufficiently known prior to system development. Architectural adaptions are inevitable for accommodating the systems to the new services, processes, technologies, standards, or users. With respect to management of development processes and systems, the notion of “continuous improvement” [106] has been adopted by the practitioners. However, due to criticality of the systems involved, planning, implementation, testing and deployment of changes can not involve downtime or similar degradation of quality of service. Instead, the systems have to quickly and frequently adapt at runtime, while maintaining the required quality of service.

In a keynote talk, Ghezzi argues [3]: “Modern software systems are increasingly embedded in an open world that is constantly evolving, because of changes in the requirements, in the surrounding environment, and in the way people interact with them. The platform itself on which software runs may change over time, as we move towards cloud computing. These changes are difficult to predict and anticipate, and their occurrence is out of control of the application developers. Because of these changes, the applications themselves need to change. Increasingly, changes in the applications cannot be handled by re-playing the development process off-line in the maintenance phase, but require the software to self-react by adapting its behavior dynamically, to continue to ensure the desired quality of service. The big challenge in front of us is how to achieve the necessary degrees of flexibility and dynamism required by software without compromising the necessary dependability.”

As pointed out by Erdil et al. [46]: “The traditional view of software maintenance deals with the correction of faults and errors that are found after the delivery of the product. However other significant changes are made to the product as software evolves. These changes can happen when the product needs to meet the new environment or new user requirements, or even to increase the products maintainability. Adaptive, perfective, and preventive maintenance deal with these changes and these three types of maintenance are considered software evolution”.

3
Independent of whether the systems undergoing changes are in the operation or in the development phase, important architectural design decisions are made often, quickly and with lack of sufficient information. When adapting the system architecture, the design alternatives may be many and the design decisions made may have unknown implications on the system and its quality characteristics (such as availability, security, performance or scalability). A change involving increased security may, for example, compromise performance or usability of a system.

As argued by Beck [23], the cost of change rises exponentially over time. The total cost of system maintenance is estimated to comprise at least 50% of total life cycle costs [119]. Zelkowitz et al. [130] also point out that in many large-scale software systems, only one-fourth to one-third of the entire life cycle costs can be attributed to software development. Most effort is spent during the operations and maintenance phase of the software life cycle, as shown by Erdil et al. [46]. The challenge is therefore how to achieve the necessary flexibility and dynamism required by software, while still preserving the necessary overall quality. Thus, there is a need for decision-making support which facilitates the analysis of effects of architectural adaptions, on the overall quality of a system as a whole.

The software product quality standard ISO 9126 [61] defines quality as “the totality of features and characteristics of a software product that bear on its ability to satisfy stated and implied needs”. The ISO 9126 standard provides an established specification of decomposed quality notions with their qualitative and quantitative definitions. This is however only a specification and no support for analysis or decision making is offered. The goal/question/metric [20,21] is an approach for specification of the quality goals and their metrics. It provides guidance and notation for decomposition of the quality notions, but does not support “what-if” analysis.

For most enterprises, ensuring the right level of system quality and minimizing time-to-market of services, is critical for maintaining their businesses. On the one hand, there is a risk of insufficient system quality, while on the other hand there is a risk of overspending and reaching a level of quality that is not really necessary. Being able to adapt to change more efficiently reduces the overall cost of change and of the project [46]. Moreover, foreseeing the implications of a change at an early stage, makes it easier to handle the risks. As pointed out by Erdil et al. [46]: “The further the developers get into the development process, the more costly it becomes to fix risks. Therefore, it is important to fix them early. If a problem arises or the user requirements change, it will be much easier to accommodate changes.” To enable the adaptability and interoperability of systems, methods for predicting the effects of architectural design changes on a system’s quality prior to change implementation, are needed.

The current state-of-the-art and state-of-the-practice offer numerous more or less established methods and notations for modeling of system architectures and specifications of quality notions (for example, quality characteristics, ratings, indicators and metrics). The main weakness of the traditional approaches is the lack of practically feasible and scalable methods for high-level analysis and decision making at system level, and with respect to entire quality characteristics and their trade-offs. The existing approaches are generally either: too fine-grained and therefore not scalable at system level or quality characteristic level; devoted to modeling and not analysis, prediction or decision making; or focused on specification of either system architecture or quality, and not the interplay between the two.

Most traditional system quality research has concentrated on defect modeling, and
relies on statistical techniques such as regression analysis [51]. Tian [116] presents risk identification techniques like principal component analysis, discriminant analysis, tree-based reliability models and optimal set reduction. Common for all these techniques is that they analyze and, to some degree, predict defects, based on raw low-level data.

According to Fenton and Neil [48], most prediction models use size and complexity metrics to predict defects. Others are based on testing data, the quality of the development process, or take a multivariate approach. In many cases, there are fundamental statistical and data quality problems that undermine model validity. In fact, many prediction models tend to model only part of the underlying problem and seriously misspecify it. Fenton and Neil [48] argue that traditional statistical approaches are inadequate and recommend causal, holistic models for software defect prediction, using Bayesian Belief Networks (BBNs) [54,87]. However, there is currently no self-contained approach that schematically relates BBNs to system architecture or system quality, and that provides a process to develop expressive models for analysis of the interplay between system architecture and system quality.

Moreover, a drawback of both statistical and BBN-based models, is their lack of scalability. Providing sufficient data for statistical models requires tedious data acquisition effort and knowledge of statistical techniques. The number of estimates needed for a BBN in the general case is exponential to the number of nodes. The predictions obtained from the traditional methods may be more exact and fine-grained, but a holistic approach where architecture of a large target system and several quality characteristics are addressed, may often be infeasible within limited resources.

Dobrica and Niemel [42] as well as Mattsson et al. [82] provide surveys of the software architecture analysis methods (SAAM, ATAM, ALPSM, SAEM etc.). The methods are comprehensive and provide a high-level architecture assessment. However, they are not predictive. ATAM [31,66,67] addresses an architecture as is, and identifies its risks and trade-off points with respect to quality goals, but it does not offer an analysis toolset of how a changed architecture impacts quality. Neither does it predict quality with respect to architectural changes.

As illustrated by Figure 1.1, there is a mutual inference between the factors such
as business processes or requirements on the one hand and the system architecture on the other hand. Changes in the requirements or business processes dictate the architectural changes, while the opposite is also common, although not desirable. Business processes also have effects on the system quality through the way the systems are used, while the system quality may, to a degree, dictate the business processes and usage profile through for example high or low scalability, performance or availability of the system. The focus of this thesis lies in the intersection between system architecture and system quality, as illustrated by the bold parts of Figure 1.1. By modeling 1) system architecture; 2) the quality notions of relevance for the system; and 3) the interplay between system architecture and system quality, we examine how the changes in the architectural design influence system quality characteristics.

In the following sections we state the objective of our work and give an overview of the main contributions. Then, the structure of the thesis is presented.

1.1 Objective

Generally, predictions may be based on simulations, systems replica, tests, historical data, expert judgments, etc. In order to facilitate decision making in the context of “what-if” analyses when attempting to understand the implications of architectural design changes on quality of a system, models are a useful means for representing and analyzing the system architecture. Instead of implementing the potential architectural changes and testing their effects, model-based prediction is an alternative. Model-based prediction is based on abstract models which represent the relevant aspects of the system. A prediction based on models may address a desired number of architectural changes, without affecting the target system. As such, it is a quicker and less costly alternative to traditional implementation and testing performed in the context of understanding the effects of changes on system quality.

The main objective of this thesis is to provide better methodological support for model-based prediction of the effects of architectural design changes on the quality characteristics of a system. An analysis based on the method should provide useful results when performed in a practical setting within limited resources. This means that the process of the method should be practically applicable, yet structured and well documented; that the prediction models should be expressive enough to provide useful results and handle the intended range of changes on the target system; and that the process and the models should be comprehensible for the stakeholders involved.

In summary, our objective is to develop a method for model-based prediction of the effects of architectural design changes on the quality characteristics of a system, that is:

1. useful in the sense that the predictions are sufficiently correct;
2. applicable in an industrial context and within acceptable effort;
3. comprehensible for the stakeholders involved; and
4. cost-effective.
1.2 Contribution

The main contribution of this thesis is the PREDIQT method, where the term *method* should be understood as "A means or manner of procedure, especially a regular and systematic way of accomplishing something" [7]. PREDIQT is a method for model-based prediction of impacts of architectural design changes on system quality characteristics. PREDIQT has been developed to support the understanding of the system design and to facilitate the reasoning about alternatives for potential improvements, as well as the reasoning about existing and potential weaknesses of architectural design, with respect to individual quality characteristics and their trade-offs. The predictions obtained from the models provide propagation paths and the modified values of the estimates which express the degree of quality characteristic fulfillment at the different abstraction levels. The intended benefits of PREDIQT include improved understanding of the system design and the alternatives for potential improvements, as well as existing and potential weaknesses of architectural design, with respect to individual quality characteristics and their trade-offs. PREDIQT undertakes a preventive approach, where analysis of effects of the architectural design changes, facilitate both the decision making prior to deployment of a change and discovery of effects of the change on quality.

In relation to the PREDIQT method, in this thesis we have particular focus on the following four artifacts:

1. The process of the PREDIQT method.
2. The approach to modeling in PREDIQT.
3. The approach to uncertainty handling in PREDIQT.
4. The approach to traceability handling in PREDIQT.

In the sequel, we give a brief overview of each of these artifacts.

1.2.1 The process of the PREDIQT method

A PREDIQT-based analysis is managed by an analysis leader (also called "analyst") who has expertise in the PREDIQT method. The analyst is the one guiding the participants of the analysis through its various steps. The analyst also documents the results.

The process of the PREDIQT method is meant to serve as a stepwise and modular guide for the analyst to systematically and correctly conduct a PREDIQT-based analysis. The process produces and applies a multi-layer model structure, called prediction models. The process uses a stepwise approach to modeling system architecture and system quality, in order to document and analyze dependence of system quality characteristics upon the architecture. The resulting prediction models are used to analyze the effects of architectural design changes on system quality, as well as deduce the degrees of quality characteristic fulfillment at various abstraction levels.

The PREDIQT process consists of three overall phases, each relying on the previous one: "Target modeling", "Verification of prediction models", and "Application of prediction models". During the first phase, the target of the analysis is characterized and initial prediction models are developed.
The “Verification of prediction models” is an iterative phase that aims to validate the prediction models (with respect to the structure and the individual parameters). Data acquisition with the necessary statistical power and model fitting are conducted, in order to adjust the prediction models to the evaluation results. Then, the prediction models are fitted and evaluated as a whole with respect to their completeness, correctness and consistency.

When the prediction models are approved, the “Application of prediction models” phase is initiated. During this phase, a potential architectural change is specified on the relevant prediction models and the effects of the change on the quality characteristic values are analyzed based on the models.

1.2.2 The approach to modeling in PREDIQT

The approach to modeling in PREDIQT defines the structure of prediction models as well as the notation, terminology and guidance for development and usage of the prediction models. The prediction models represent three system relevant perspectives: quality notions of the target system (through “Quality Models”), architectural design of the target system (through “Design Models”), and the dependencies between architectural design and system quality characteristics (through “Dependency Views”). The Design Models are used to specify the target system and the changes whose effects on quality are to be predicted. The Quality Models are used to specify the target system relevant quality notions by defining their relationships, interpretations and metrics. The Dependency Views (DVs) are weighted dependency trees that express the interplay between the system architectural design and the quality characteristics of the target system. The values and the dependencies modeled through the DVs are based on the definitions provided by the Quality Models.

The approach to modeling in PREDIQT aims to provide comprehensive support for development of prediction models capable of analyzing the effects of changes on system quality. To this end, the above mentioned structure of the prediction models (representing the three perspectives: architectural design, quality and interaction between architecture and quality) is proposed. Our experience is that existing modeling notations, like UML [108], BPEL [98] and TOGAF [9], offer sufficient support for development of the Design Models. Similarly, ISO 9126 [61] standard and goal/question/metric [20,21] approach, are sufficient for developing the Quality Models. PREDIQT therefore makes use of existing notations to express these perspectives. We were however unable to find a comprehensible, scalable and expressive notation for modeling and analysis of dependencies between system architecture and system quality. Therefore, PREDIQT offers a method specific notation, the DVs, for this purpose.

PREDIQT offers rules for developing the structure of DVs, estimating and propagating their parameters, as well as providing the semantics of the DVs. PREDIQT also defines the terminology used in the DV context. The DV-related rules aim to establish pragmatic and comprehensible principles making sure that the DVs have the functionality needed for analyzing the effects of changes at the abstraction levels needed and with useful accuracy. We have also developed a prototype tool for change propagation and sensitivity analysis related to the DVs.

Apart from the above mentioned analyst, domain experts are among the mandatory participants of a PREDIQT-based analysis. The domain experts have knowledge and expertise within the particular field relevant for the target system. Thus, the role of
the domain experts is to provide the empirical input necessary to develop the models which represent the aspects relevant for the architecture of the target system and its quality. The target groups of the approach to modeling in PREDIQT are the domain experts and the analyst. The approach to modeling in PREDIQT is, during model development and verification, used by domain experts and analyst to model the dependencies between the architectural design and the quality of a system. During application of prediction models, the approach to modeling in PREDIQT is by the analyst used to analyze the effects of the architectural changes, on system quality characteristics. The predictions provide propagation paths and the updated values of the annotated quality parameters at the abstraction levels affected.

1.2.3 The approach to uncertainty handling in PREDIQT

Due to its empirical nature, input to the DVs is associated with uncertainty. By handling the uncertainty in the DVs, quality of the prediction models and accuracy of the predictions are made explicit, thus indicating which changes are predictable and whether further model fitting is needed.

In a real-life setting, making the right balance between the accuracy and the practical feasibility is what characterizes the overall challenge in selecting the appropriate approach to uncertainty handling in prediction models.

Based on a set of criteria identified for uncertainty handling in PREDIQT, we have proposed a self-contained and comprehensible approach for handling uncertainties in the DVs. Our approach is based on intervals with associated confidence level, and supports representation, propagation and analysis of all the parameters associated with uncertainty.

Based on empirical trials of PREDIQT, we have provided guidelines (presented in Section 6 of Paper 2) for the use of the uncertainty handling approach in practice. The guidelines address the ways of obtaining the empirical estimates as well as the means and measures for reducing uncertainty of the estimates.

The target groups of the uncertainty handling approach in PREDIQT are domain experts and the analyst. The uncertainty handling approach is intended to serve the domain experts in correctly and comprehensibly expressing the uncertainty in a simple yet sound manner, while the analyst uses the approach to obtain, represent, analyze and propagate the uncertainty estimates of the empirical input associated with the DVs.

1.2.4 The approach to traceability handling in PREDIQT

The analyst needs correct prediction models and schematic guidelines for using them, in order to obtain reliable and valid predictions. Reliability of predictions implies that the results of applying the models are independent of the competence of the analyst and consistent if replicated. Validity of the predictions implies that the prediction results are correct, useful and relevant. Usefulness and correctness of the prediction models require a structured documentation of both the relations between the prediction models and the rationale and assumptions made during the model development. This structured documentation is what we refer to as trace-link information.

We have deduced traceability needs in PREDIQT by providing structured and detailed guidelines for application of the prediction models and the trace-link information.
The guidelines for application of prediction models (also referred to as “PREDIQT guidelines”) are presented in Appendix 2 of Paper 5. In addition to providing instructions for use of the trace-link information, the guidelines identify the relevant trace-links. The traceability needs are defined by a traceability scheme. A prototype tool which can be used to define, document, search for and represent the trace-links needed, has been developed. Since the intended application of the prediction models does not include implementation of change on the target system, but only analysis of effects of the independent architectural design changes on quality of the target system (in its currently modeled state), maintenance of prediction models is beyond the scope of PREDIQT. Therefore, trace maintenance is not needed. Hence, traceability in PREDIQT involves planning, recording and use of the traces.

The traceability approach in PREDIQT serves the analyst in systematically obtaining correct and well documented prediction models, as well as using the prediction models along with the traceability tool, correctly. The traceability approach focuses on the most influential aspects of prediction models and their use, in order to properly balance practical applicability with model accuracy, within acceptable resources allocated for an analysis.

### 1.3 Structure of the thesis

The Faculty of Mathematics and Natural Sciences at the University of Oslo recommends that a dissertation is presented either as a monograph or as a collection of research papers. We have chosen the latter.

This dissertation is based on a collection of 5 research papers and structured into two main parts. Part I (the introductory part) provides the context and an overall view of the work, while Part II contains the collection of research papers. The purpose of the introductory part is to explain the overall context of the artifacts presented in the research papers and to explain how they fit together. Part I is organized into the following chapters:

**Chapter 1 - Introduction** motivates the need for model-based system quality prediction, and presents the main objective and the contribution of this thesis. It also outlines the structure of this thesis.

**Chapter 2 - Problem characterization** provides background material and argues why the main objective of this thesis is desirable. In addition, success criteria that should be fulfilled by our artifacts in order to reach the overall objective, are presented.

**Chapter 3 - Research method** presents a method for technology research and explains how this method has been applied in the research on which this thesis reports.

**Chapter 4 - State of the art** presents the state of the art of relevance for this thesis.

**Chapter 5 - Achievements: the overall picture** provides an overview of the developed artifacts and how they fit together.
Chapter 6 - Overview of research papers provides publication details of each research paper of this thesis.

Chapter 7 - Discussion addresses the extent to which our artifacts satisfy the success criteria stated in Chapter 2. We also discuss how our work relates to state of the art.

Chapter 8 - Conclusion presents conclusions and possible areas of future work.

Each research paper in Part II is meant to be self-contained and can be read independently of the others. The papers therefore overlap to some extent with regard to explanations and definitions of the basic terminology. We recommend that the papers are read in the order they appear in the thesis.
Introduction
Chapter 2

Problem characterization

In Chapter 1 we presented the overall motivation and objective for our research. In this chapter we refine this objective into a set of success criteria. In Section 2.1 we clarify the interpretation of a set of core concepts that play an essential role throughout the thesis. In Section 2.2 we present the success criteria. That is, for each artifact, a set of criteria that should be fulfilled for a successful accomplishment of our objective, is characterized in Section 2.2.1 through Section 2.2.4.

2.1 Conceptual clarification

This section characterizes the main terminology used throughout the thesis. Concepts such as system architecture, quality model and prediction are presented and their meaning clarified in the context of this work.

2.1.1 System

One of the initial steps of an analysis is characterization of its target, that is, the scope that the analysis should cover and the aspects that should be addressed. The targets of the different analyses may have varying purpose, extent, users, etc. However, in the PREDIQT context, the target of the analysis is always a computerized system. In the context of computerized systems, a system is characterized as a set of components which interact and operate as a whole. Most systems share common characteristics, such as structure, behavior, interconnectivity and functions. By the term system we therefore mean “A group of interacting, interrelated, or interdependent computerized elements forming a complex whole”, which is a direct adaptation of the general definition given in the American Heritage Dictionary of the English Language [44].

2.1.2 System architecture

According to a definition proposed by the Katholieke Universiteit Leuven [8], system architecture refers to the architecture of a specific construction or system. System architecture is the result of a design process for a specific system and specifies the functions of components, their interfaces, their interactions, and constraints. This specification is the basis for detailed design and implementation steps. Architecture is basically an abstract representation of the internal organization or design of a system, with respect to aspects of the system such as its software or hardware.
components, their interfaces, their interactions, interaction rules, storage, dataflow, organization of components and data, etc. A system architecture can represent an existing or a to-be-created system. It is used to express in an abstract manner the content of the elements comprising a system, the relationships among those elements, and the rules governing those relationships. The components and the relationships between these components that an architecture represents may consist of hardware, software, documentation, facilities, manual procedures, or roles played by organizations or people. A system architecture is primarily concerned with the internal interfaces among the components of a system, and the interface between the system and its external environment.

There is no universally agreed definition which specifies the aspects that constitute a system architecture. Architecture is by the International Organization for Standardization Reference Model for Open Distributed Processing (RM-ODP) [12] defined as “A set of rules to define the structure of a system and the interrelationships between its parts”. The IEEE Standard for describing the architecture of a software-intensive system [1] defines a system architecture as “The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution”. In this thesis, we have embraced the definition of architecture provided by IEEE.

2.1.3 Architectural design change

The current development in usage of computerized systems is directed towards dynamic, evolving systems with high demands on their quality. Common characteristics are collaboration and distribution of systems, services and users. This has resulted in new demands on open standards, data quality, as well as on flexibility, availability, security, usability and configurability of the systems. Groups and organizations (their structures and collaboration patterns) change more often, resulting in needs for adjustments of the related workflow models, policies, data structures and interfaces. Thus, we need to be able to easily and timely perform frequent modifications of architectures while still preserving the necessary levels of quality and being able to verify it. Unnecessary degree of flexibility or flexibility beyond the needed time scope is often too costly. Therefore, the architecture needs to be adaptable and flexible to a degree which is sufficient without being unnecessary. Moreover, the architecture needs to be consistent with the change rate of workflow within and across the organizations that are involved. In order to achieve these goals, thorough planning in form of re-design of the system architecture has to be performed prior to implementing the architectural adaptations according to the new design. Such re-design of the system architecture is what we refer to as architectural design change. It is assumed that an architectural design change is followed by an implementation of the new architectural design of the system.

Change is an inevitable and intrinsic part of any kind of effort involving software architecture. Architectural design changes may include the changes on components, their interfaces, their interactions, and constraints. Removing, adding or modifying the components, interfaces, interactions, or constraints, are examples of architectural design changes. An architectural design change may be motivated by a changed functional or non-functional requirement. The functional requirements address the way the system works and interacts with its environment and users. The non-functional
requirements address the quality characteristics of the system. The functional and
the non-functional requirements may in turn be initiated by new or changed working
processes, standards, technologies, users or needs.

An important aspect in this context is the modularity and flexibility of system archi-
tectures in terms of adapting to new collaborations, offering new services, or generally
adjusting the existing technical conditions (syntax, semantics, dataflow mechanisms,
security mechanisms, general data quality, performance) to new technical requirements
and user needs (workflow, use areas, regulations, compliance to standards). Based on
the above stated definition of architecture, we define architectural design change as
“The intended change in organization of a system in terms of software or hardware
components of the system, their relationships to each other and the environment, as
well as change in the principles governing the design and evolution of the system”.

2.1.4 System quality

Product quality or quality of service is one of the major objectives of software and
system engineering. Quality is most often associated with a limited selection of non-
functional requirements relevant for a specific target system. There exist many defini-
tions of quality which focus on aspects such as: conformance to specification, timeliness
or fitness for purpose. Generally, quality is concerned with the degree to which the
relevant non-functional requirements are fulfilled. The first international standard for
software quality management (ISO 9126) [61] defines quality as “The totality of fea-
tures and characteristics of a software product that bear on its ability to satisfy stated
and implied needs”. In order to include computerized systems in general rather than
“only” software, we adapt the definition of quality provided by ISO 9126. Thus we
propose using the following definition of quality in the context of this thesis: “The
totality of features and characteristics of a system that bear on its ability to satisfy
stated and implied needs”.

2.1.5 Quality characteristic

In context of quantitative prediction, we must be able to define quality in terms of
specific non-functional attributes relevant to the target stakeholders and the system
in question. That is, it is necessary to specify how to measure the extent to which
the attributes are fulfilled in our systems. Such attributes are also called quality
characteristics or factors. Quality is by ISO 9126 [61] decomposed into six factors:
functionality, reliability, efficiency, usability, maintainability and portability. Each can
be refined through multiple levels of sub-characteristics. The clear advantage of this
universal standard for expressing quality characteristics, is the possibility of comparing
one system with another. The definitions of the characteristics and their decomposition
which are provided by ISO 9126 have been used throughout this thesis.

2.1.6 Model

According to Kleppe et al. [71], “A model is a description of (part of) a system written
in a well-defined language”. Another definition of model is proposed by OMG: “A
simplified or idealized description or conception of a particular system” [97]. The term
model is often used to refer to abstractions above code. Models may come in different
forms, most usually as equations or diagrams. These show how variables or parts are related, so that we can examine these relationships and make judgments about them.

The above mentioned RM-ODP, for example, uses an object modeling approach to describe systems. Two structuring approaches are used to simplify the design in complex systems: five “viewpoints” provide different perspectives of describing the system; and eight “transparencies” identify specific problems which distributed system standards may address. Each viewpoint is associated with a language, which can be used to describe systems from that viewpoint.

In this thesis, we embrace the characterization of model provided by Fenton and Pfleeger [49]: “An abstraction of reality, allowing us to strip away detail and view an entity or concept from a particular perspective”.

2.1.7 Quality models

Models are needed to specify quality characteristics in a comprehensible and measurable form. There is a distinction between internal and external quality characteristics. The former ones are often available early in the life cycle, whereas the external ones are dependent on both machine environment and the user, and therefore measurable only when the product is complete. Internal characteristics are often easier to measure than the external ones. Moreover, the internal characteristics may be predictors of the external characteristics. Quality models are usually constructed in a tree-like fashion, where the upper branches hold important high-level characteristics of software quality, such as reliability or usability, that an analyst would like to quantify. Then, each quality factor is decomposed into lower level criteria, such as for example completeness or accuracy. The criteria may be shared among the high-level factors. The criteria are more fine grained and thus easier to measure than the factors. The criteria are then decomposed into actual measures or metrics. The tree represents the relationships between the factors and the criteria, so that the factors can be interpreted and measured in terms of the dependent criteria. Both McCall [84] and Boehm [25] have described quality in terms of models, using a stepwise approach. The work of McCall was used as a basis for the first international standard for software quality management (ISO 9126) [61].

As outlined by Fenton and Pfleeger [49], the tree-based taxonomies are helpful, as we may use them to monitor software quality in two different ways:

1. The “fixed model” approach – we assume that all important quality factors needed to monitor a project are a subset of those in a published model.

2. The “define your own quality model” approach – we accept the general philosophy that quality is composed of many factors, but we do not adopt the characterization of quality provided by a given model. Instead, we meet with stakeholders to reach a consensus on which quality attributes are important for a given product. Together, we decide a decomposition (possibly guided by an existing model) in which we agree on specific measures for the (lowest-level) attributes (also called criteria or indicators) and specific relationships between them. Then we measure the quality attributes objectively to see if they meet specified, quantified targets.

In this thesis, we have applied the latter approach.
2.1.8 Prediction

As argued earlier, predictions may generally be based on simulations, knowledge, experience, systems replica, tests, historical data, models (mathematical, statistical, graphical), expert judgments, etc. A prediction or a forecast is usually a statement about a future state of an object of analysis. The terms prediction and forecast are often used interchangeably, but the main distinction is that a prediction expresses an explicit outcome, while a forecast may cover a range of possible outcomes. Although certain statements in many cases are impossible, prediction is necessary to allow plans to be made about possible developments. We consider prediction as the result of an analysis based on informed and well documented judgments. In the context of this thesis, we adopt the definition of prediction provided by Collins English Dictionary: “The act of forecasting in advance, or something that is forecast in advance” [111].

2.1.9 Model-based prediction

Our assumption is that quality controlling measures are easier and less costly to realize during the change planning, rather than during or after the deployment of an architectural change. In order to facilitate informed planning of architectural changes, their effects on system quality need to be predictable at the stage of architectural design. That is, we aim to predict the effects of the early decisions regarding changes in system architecture, on the quality of the system. Thus, a schematic approach to designing architecture, is needed. Such an approach should support not only documenting the architectural design, but also prediction of the effects of the architectural design changes with respect to the resulting system quality. Prediction can also help if there are multiple design alternatives which fulfill the quality goals. In such a case, systematic trade-off analyses can be conducted, in which the analyst may maximize the utility-to-cost ratio, in collaboration with the stakeholders. Since an architectural design is available prior to the implementation of the system architecture, it may be represented by abstract models. A prediction of the effects of the architectural design changes on quality should therefore be based on the models. We have above defined the notions of model and prediction. Based on the above stated definitions of model and prediction, which have been adopted by this thesis, we define model-based prediction as “An in-advance forecasting based on an abstraction of reality which is represented by models”.

2.2 Success criteria

In Chapter 1 we outlined the problem area that motivated the work of this thesis and argued that there is a need for a method that supports model-based quality prediction with respect to architectural design changes. As explained in Section 1.1 our objective is to develop a method for model-based prediction of the effects of architectural design changes on the quality characteristics of a system, that is:

1. useful in the sense that the predictions are sufficiently correct;
2. applicable in an industrial context and within acceptable effort;
3. comprehensible for the stakeholders involved; and
4. cost-effective.

Developing a complete method for system quality prediction is an extensive task and lies beyond the scope of this thesis. We have focused on developing four artifacts, each of which is necessary within such a method:

1. The process of the PREDIQT method.
2. The approach to modeling in PREDIQT.
3. The approach to uncertainty handling in PREDIQT.
4. The approach to traceability handling in PREDIQT.

We consider these four artifacts to be the building blocks of the PREDIQT method. That is, the four artifacts are necessary and related parts of PREDIQT, although they do not cover all the needs of a method for system quality prediction.

The relation between (1), (2), (3) and (4), and their particular contribution to the overall objective of a system quality prediction method, are summarized in Chapter 5. The main hypothesis for each part is that it manages to fulfill its intended purpose, and that it is feasible to use by its intended users. As argued in Section 1.2, the different contributions target different user groups as follows:

- The target group of the process of the PREDIQT method is the analyst.
- The target group of the approach to modeling in PREDIQT are the domain experts and the analyst.
- The target group of the approach to uncertainty handling in PREDIQT are the domain experts and the analyst.
- The target group of the approach to traceability handling in PREDIQT is the analyst.

Below we have identified a set of success criteria that the artifacts presented in Section 1.2 should fulfill. We start each section by motivating the success criteria for the artifact in question. Each success criterion is more closely explained under its statement.

### 2.2.1 The process of the PREDIQT method

By a process we mean a specified series of ordered actions or steps conducted in order to achieve a particular objective. In the PREDIQT context, the process aims to provide the guidance for the analyst during development, validation and application of the prediction models.

The analyst is the one managing the PREDIQT-based analysis and documenting the results. This implies that the analyst leads the process, understands and in some cases participates in the development of the prediction models, and documents the results. The analyst does however not necessarily have expertise on the target system under analysis, but should understand it sufficiently and be capable of collecting and processing the input needed in order to manage the development of the prediction models. One objective for the analyst is to successfully conduct and document the analysis within the frame of the limited resources allocated.
**Success criterion 1** The process of the PREDIQT method provides sufficient and comprehensible guidance for an analyst to conduct a PREDIQT-based analysis.

By guidance we mean description of steps and actions to be conducted by the analyst during a PREDIQT-based analysis. By an analyst, we mean a person familiar with the PREDIQT method and trained to conduct a PREDIQT-based analysis. The analyst is however not expected to be familiar with the functional, technical or operational properties (or requirements) of the target system. Likewise, the analyst is not expected to have domain expertise in architectural design or quality characteristics (or requirements) of the target system. The analyst is however expected to have sufficient expertise in modeling languages needed for development of prediction models, as well as expertise in the PREDIQT method (including the above specified four artifacts).

**Success criterion 2** The process of the PREDIQT method results in predictions that are sufficiently correct.

The Success Criteria 1 and 2 imply that the PREDIQT process should provide comprehensible and sufficient guidance for the analyst so that the resulting predictions are sufficiently correct in terms of needed precision and coverage. The analyst aims to capture through the prediction models the relevant knowledge, information and requirements on system architecture and system quality, as well as usage profile, assumptions and constraints. Thus, the process should provide the instructions for modeling, interaction with the overall stakeholders, as well as correct and consistent usage of the prediction models. The prediction models developed during the process should be sufficiently expressive and precise for adopting the relevant and realistic architectural design changes, as well as efficiently analyzing their effects on quality. A structured process as well as correct and expressive prediction models are a prerequisite for correctness of the model-based predictions. Sufficient correctness of the predictions involves both the needed prediction certainty as well as the ability of the prediction models to handle the relevant architectural design changes.

**Success criterion 3** The process of the PREDIQT method is applicable in an industrial context and with acceptable effort.

This criterion addresses the feasibility of conducting the PREDIQT process within limited resources in a realistic setting. The PREDIQT-based analysis should be feasible within the allocated resources, while still providing useful predictions in a real-life industrial setting. By useful predictions, we mean such predictions that informed decisions can be made on relevant changes. By relevant changes we mean realistic architectural changes on a real-life target system. By informed decisions we mean reasoning based on the predictions which are sufficiently certain and cover the aspects needed. Moreover, the predictions should be reliable, that is, provide consistent results when replicated. A process which provides useful prediction models within the limited resources allocated for an analysis, is cost-effective, which is a part of the overall objective.

### 2.2.2 The approach to modeling in PREDIQT

By a modeling approach we mean an approach for providing a complete set of models with a pre-defined purpose. A modeling approach should contain: a notation (graphical
or textual) – also called syntax, a specification of meaning of the notation and terminology used – also called semantics, and a specification of actions and rules needed for developing and using the models – also called pragmatics.

When developing the prediction models, the analyst closely interacts with the domain experts and guides them. During model development, the analyst only has the role of facilitator and is responsible for explaining the modeling approach and documenting the models. The domain experts directly provide input during the interactive model development. The domain experts receive (from the analyst) the instructions on usage as well as syntax and semantics for the modeling notations. The guidelines for how to instruct the domain experts are provided to the analyst through the PREDIQT process guidelines. The overall empirical input to the models is based on documentation and measurements.

Moreover, when applying the prediction models and obtaining the quality predictions (in terms of propagation paths and quantitative estimates), the analyst uses the prediction models to apply the changes and to interpret the results of the predictions. The guidelines for how to correctly apply the prediction models, are provided to the analyst through the PREDIQT process guidelines. The approach to modeling in PREDIQT aims to facilitate knowledge management through discussions and structured brainstorming, during modeling of dependencies between architecture and quality of a target system. Since model development is based on empirical input – mainly structured brainstorming among experts with different professional backgrounds, the approach to modeling in PREDIQT has to be easy to understand and use. Still, the modeling approach has to provide the expressiveness, granularity and precision needed for useful prediction models.

**Success criterion 4** *The approach to modeling in PREDIQT provides a sufficiently expressive and self-contained set of models and instructions needed for predicting effects of architectural design changes with respect to system quality.*

The necessary modeling support in terms of syntax, semantics and pragmatics, is in the interest of both the domain experts and the analyst. The fulfillment of this criterion ensures that the necessary expressiveness for both high-level and low-level modeling and analysis of dependencies between system architecture and system quality, is offered by the modeling approach. For dependency analysis, a complete model structure addressing system architecture, system quality and the interplay between the two is a prerequisite. This criterion implies that the structure of the prediction models is complete and sufficiently expressive for this purpose.

**Success criterion 5** *The approach to modeling in PREDIQT offers the sufficient support for analyzing effects of architectural design changes on system quality.*

The analyst needs language support for obtaining predictions of effects of architectural design changes on system quality, in terms of change propagation paths and quality estimate values. The language should offer inference model which is compliant with its syntax and semantics.

**Success criterion 6** *The approach to modeling in PREDIQT is suitable for handling the relevant kinds of empirical input.*
2.2 Success criteria

By empirical input we mean the input acquired from experience, rather than theory. Examples of empirical input include expert judgments, logs or measurements. Examples of origins of empirical input include cases studies, experiments or observation. During the model development, it is in the interest of the domain experts and the analyst that both expert-judgment-based and measurement-based input can, with the precision available, be directly related to and expressed by the notions of the modeling approach. Moreover, it is during both model development and application, in the interest of the analyst that the arithmetic operations needed for propagation fulfill the needs for inference of all kinds of empirical input.

Success criterion 7 The complexity of the prediction models is limited and scalable in a practical setting.

As a part of the overall objective regarding practical feasibility of the approach to modeling in PREDIQT, minimized complexity of the prediction models is in the interest of the analyst who needs to obtain the input for model structure and estimates from the domain experts or measurements.

Success criterion 8 The approach to modeling in PREDIQT is easily comprehensible by the domain experts and the analyst.

The fulfillment of this criterion ensures that it is easy for the domain experts (with various degrees of technical background) to develop the prediction models. Communication and a common understanding of the models should be facilitated by the modeling approach. Moreover, it should be easy for the analyst to analyze the effects of architectural design changes on quality, based on the prediction models.

2.2.3 The approach to uncertainty handling in PREDIQT

The values assigned to the prediction models originate from domain expert judgments and measurements on the system. However fine grained, the prediction models contain a certain degree of uncertainty due to lack and inaccuracy of empirical input. The approach to uncertainty handling in PREDIQT addresses representation, propagation and analysis of uncertainties in prediction models. Such an approach is essential to facilitate model fitting (that is, adjustment of models during verification), identify the kinds of architectural design changes which can be handled by the prediction models, and indicate the value of added information.

The empirical input is always associated with a degree of uncertainty. Uncertainty is generally categorized into two different types: aleatory (due to inherent randomness of the system or variability of the usage profile) and epistemic (due to lack of knowledge or information about the system) [70]. In majority of the system quality prediction models, aleatory uncertainty is negligible in terms of magnitude and impact, while the epistemic one is crucial. It is therefore the epistemic uncertainty we focus on when dealing with the parameters in the prediction models.

The success criteria for the uncertainty handling approach are based on the experiences obtained during the two case studies presented in Paper 1 and Paper 4, respectively. The success criteria reflect the properties of the empirical input and its sources, that is, domain expert judgments and measurements. Precision and comprehensibility are two main goals. Moreover, the success criteria require soundness of the
propagation and possibility of mapping the estimate representation to the representation form of both kinds of the empirical input. The design decisions made aim to balance coverage of the needs identified with simplicity of the solution, in order to provide an approach which is useful and practically applicable within acceptable resources and as a part of a PREDIQT-based analysis. The simplicity also makes it less prone to unstable over-fitting, as well as bias or inaccuracy of the estimations.

The objective of the uncertainty handling approach is to serve the domain experts in correctly and comprehensibly expressing the uncertainty in a simple yet sound manner, while the analyst uses the approach to:

- obtain and represent the uncertain input from the domain experts and the measurements;
- propagate the uncertainty estimates during analysis (that is, when applying the prediction models to obtain predictions);
- statistically analyze the uncertainty estimates.

Success criterion 9 The approach to uncertainty handling in PREDIQT is applicable for representing the uncertainty estimates related to both expert-judgment-based and measurement-based input.

The analyst needs to be able to represent the quantitative uncertainty estimates based on both expert judgments and measurements. Fulfillment of this criterion implies that the representation form of uncertainty is applicable for all kinds of empirical input, regardless of the need to transform the representation form from the one the input is obtained in to another one. Moreover, the representation of the uncertainty estimates has to offer the precision needed, so that the actual uncertainty of the input can be expressed and, if needed, vary across the estimates.

Success criterion 10 The approach to uncertainty handling in PREDIQT offers sufficient and comprehensible guidelines to the analyst for obtaining the input along with its uncertainty, based on domain expert judgments and measurements.

The analyst needs guidelines for correctly obtaining the input based on domain expert judgments and measurements. That is, the input and the uncertainty associated with it, need to be requested systematically and in a comprehensible manner. In relation to the domain expert judgments, the right questions need to be posed, and the the experts need to be guided when providing the estimates of the parameter values and the associated uncertainty estimates. Likewise, when conducting measurements on the system, the validity of the measurements needs to be related to the uncertainty estimates.

Success criterion 11 The number of uncertainty estimates needed is linearly proportional to the number of nodes in the model.

To make the approach scalable in practice and be able to obtain the estimates within limited resources, the analyst needs reduced complexity. To this end, the number of the uncertainty estimates needs to be minimized.
2.2 Success criteria

**Success criterion 12** The approach to uncertainty handling in PREDIQT supports propagation of the uncertainty estimates.

The fulfillment of this criterion implies that the uncertainty estimates can be propagated correctly. The arithmetic operations supported should fulfill the needs for correct inference of all kinds of empirical input, so that the predictions can be expressed with a measure of uncertainty.

**Success criterion 13** The approach to uncertainty handling in PREDIQT is comprehensible for the domain experts.

For validity and reliability of the input, it is in the interest of the analyst that correct estimates are obtained and that a common understanding of the uncertainty estimates can be reached among the domain experts. Thus, correspondence between the empirical input and the representation form of the approach, with respect to uncertainty, should be simple and intuitive for the domain experts. For validating correctness of the inferred parameters, it is also in the interest of the analyst that propagation is simple and can, while performed by a tool, be understood by the domain experts.

**Success criterion 14** The approach to uncertainty handling in PREDIQT supports statistical analysis of the uncertainty estimates.

Statistical analysis are preformed by the analyst during model validation. In this context, validity and reliability of the input is analyzed so that the models can be fitted in case of bias or deviations. Moreover, simple sensitivity analysis is offered by the DV tool during change propagation. In relation to both validation and change propagation, statistical operators such as mean, standard deviation, regression analysis and skewness, are by the analyst used for evaluating the confidence in the values obtained.

### 2.2.4 The approach to traceability handling in PREDIQT

Usefulness of the prediction models requires a structured documentation of both the relations between the prediction models and the rationale and assumptions made during the model development. This structured documentation is what we refer to as trace-link information. Traceability is necessary for correctness and correct usage of the prediction models in order to facilitate valid and reliable predictions. IEEE [10] defines traceability as:

1. the degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another; for example, the degree to which the requirements and design of a given software component match;

2. the degree to which each element in a software development product establishes its reason for existing.

The approach to traceability handling in PREDIQT addresses documenting and retrieval of the rationale and assumptions made during the model development, as well as the dependencies between the elements of the prediction models. Trace-link
Problem characterization

information is, by the analyst, documented during model development and used during application of the prediction models.

The traceability efforts are resource demanding and have to be goal-oriented in order to obtain the accuracy necessary within the limited resources allocated for a PREDIQT-based analysis. Thus, the challenge lies in covering the right trace-link information with the sufficient granularity for being applicable in a practical setting. The amount of the trace info provided has to be manageable within the resources allocated for a PREDIQT-based analysis. That is, the approach should facilitate documentation (during the model development), as well as searching and reporting (during the application of a change on prediction models) of the trace info in realistic settings.

Success criterion 15 The approach to traceability handling in PREDIQT offers the support necessary for an analyst to handle the needed kinds of the trace-links.

According to the guidelines for application of the prediction models, two overall kinds of the trace-links may be created during the development of the prediction models:

- The relations between traceable elements of the prediction models;
- Structure or contents of traceable elements documented through 1) external documents, or 2) a specification of the rationale and assumptions made.

The analyst needs the sufficient support for handling all the needed kinds of the trace-links.

Success criterion 16 The approach to traceability handling in PREDIQT offers searching facilities with model types and model elements as input parameters.

Retrieval of traces is needed by the analyst when obtaining the predictions. Certain steps of the guidelines for application of the prediction models assume a facility for searching trace-links with model types and model elements as input parameters. By a model type we mean one of the three kinds of prediction models: Design Model, Quality Model and DV. By model elements we mean elements belonging to the diagrams of the prediction models.

Success criterion 17 The approach to traceability handling in PREDIQT offers reporting facilities which show the linked elements and the trace-link properties.

This success criterion implies that the search results can be represented and visualized so that each retrieved trace-link is specified with the traceable elements involved and the properties of the trace-link.

Success criterion 18 The approach to traceability handling in PREDIQT is flexible with respect to granularity of trace information.

Granularity of trace information may vary depending on needs for, importance and availability of trace information. Thus, the solution needs to support varying amount and precision of the trace-link information recorded.
Success criterion 19 The approach to traceability handling in PREDIQT supports real-life applications of PREDIQT.

In order to be applicable in a practical setting, it is necessary that sufficient trace-links can, within acceptable resources, be provided during an analysis. The trace-links provided need to be complete and sufficiently accurate for fulfilling the needs of an analyst, during the application of the prediction models.
Chapter 3

Research method

In terms of its close connection to both theory and practice, computer science is in a unique position compared to the natural sciences. While theories in sciences mainly focus on the *what*, computer science is most often driven by the need for explaining *how* something can be achieved through the close interaction between theory and the technology that realizes it [28].

Moreover, computer science is a relatively young discipline, and it has even been questioned whether computer science at all qualifies as a science [13, 29, 40]. Abelson and Sussman [13] claim that computer science is not a science, and substantiate this by contrasting computation with classical mathematics. They do not explain what it takes to be a science or why computer science does not qualify, but claim that the difference between computation and classical mathematical subjects is that mathematics provides a framework for dealing with notions of *what is*, while computation provides a framework for dealing with notions of *how to*.

Similarly, Brooks [29] argues that science is concerned with the discovery of facts and laws, as opposed to computer science which is an engineering discipline concerned with building things. According to Brooks, the invention and publication of languages and computing algorithms is not in itself an end. Instead, computer science should recognize its artifacts as tools, and these tools should be tested by their usefulness and cost.

However, there is a widely established agreement that computer science indeed is a science. This view is usually substantiated by claiming that computer science shares many features with other sciences, in terms of the research method. Denning [40] states that the activities within computer science are a blend of science, engineering and mathematics. Based on studying the information processes, Denning argues that computer science follows the scientific paradigm – the process of forming and testing hypotheses where successful hypotheses become models that explain and predict phenomena in the world. The author furthermore claims that there are numerous examples of computer science research being settled in the scientific paradigm. For example, software engineering researchers hypothesize models for how programming is done and how defects arise. By testing the models, they aim to understand which models work well and how they can be used to create better software products. The view that computer science is not a science because it studies man-made objects is dismissed on the grounds that computer science studies information processes that may be either artificial or natural.

In a similar way as Denning, Tichy [117] considers the subjects of inquiry in computer science to be not computers, but information processes. Computers and software
products are seen as models of such processes. Although in computer science the subject of inquiry is information rather than matter, Tichy argues that this makes no difference in the applicability of the traditional scientific method of observing phenomena, formulating explanations and theories, and testing them. Tichy argues that computer scientists should experiment more to test their theories.

In the sequel we briefly present a method for technology research that rests on the scientific paradigm. Subsequently we outline the different strategies for evaluation and describe how we have applied this method in the work that has led to this thesis.

### 3.1 A technology research method

The research on which this thesis reports can be labeled *technology research* and has been driven by an iterative process after a schema proposed by Solheim and Stølen [113]. In cases of both classical research and technology research, the starting point is an overall hypothesis of the form B solves the problem A. In classical research A is the need for a new theory, whereas in technology research A is the need for a new artifact. While the classical researcher aims to gain new knowledge about reality, the technology researcher is interested in how to produce a new and improved artifact [113].

Despite the difference between the basic questions leading the classical researcher on the one hand and the technology researcher on the other hand, Solheim and Stolen assert that the research process in both cases follows the same principal phases, and that technology research should be founded on the classical research paradigm. They claim that technology research has a lot in common with classical research and should be conducted in accordance with the main principles of the hypothetico-deductive method of classical research: 1) recognition of a theoretical problem; 2) proposal of conjectured solution to the problem (hypothesis); 3) testing of the hypothesis; and 4) conclusion: retain hypothesis if test is positive, otherwise reject hypothesis as falsified and possibly devise new hypotheses/problems [73].

The development of an artifact is motivated by some kind of a *need*. This need gives rise to *success criteria*. The overall hypothesis of technology research is that the new artifact *satisfies the success criteria*.

A common way of verifying that the overall hypothesis is satisfied, is to formulate a set of sub-hypotheses and corresponding predictions whose falsification imply that the overall hypothesis is discredited. Hence, the predictions serve as a basis for gathering evidence about the validity of the overall hypothesis.

Technology research, like classical research, is an iterative process [113]. Technology research can be characterized by the following three main phases [113]:

1. **Problem analysis** – The researcher surveys a potential need for a new and improved artifact by interacting with potential users and other interested parties. During this stage the researcher identifies a set of success criteria for the artifact.

2. **Innovation** – The researcher attempts to construct an artifact that satisfies the potential need. The overall hypothesis is that the artifact satisfies the need. In order to evaluate the overall hypothesis, the researcher has to formulate sub-hypotheses about the properties of the artifact.

3. **Evaluation** – Predictions about the artifact are made based on the identified need.
3.2 Strategies for evaluation

The optimum strategy for hypothesis testing is to use several different strategies in parallel that have different strengths. According to McGrath [85], research evidence is gathered to maximize three things: (A) generality, (B) precision, and (C) realism. The best would be to choose a strategy that scores high on generality, precision and realism, but that is, according to McGrath, not possible. He divides research strategies into four categories:

- artificial setting;
- natural setting;
- independent of empirical evidence;

Figure 3.1: Method for technology research – main steps (adopted from Solheim and Stølen [113])

The researcher then evaluates the validity of the predictions. If the evaluation results are positive, the researcher may argue that the artifact satisfies the need.

Figure 3.1 is adopted from Solheim and Stølen [113] and summarizes the three main steps of the iterative process in technology research.

Technology development is closely related to technology research. According to Solheim and Stølen [113], technology research is distinguished from technology development by the former giving rise to new knowledge of interest to others. More precisely, this means that to qualify as technology research, the invented artifact must represent new knowledge, this knowledge must be of interest to others, and the results and the new knowledge must be documented in a way that enables examination and evaluation by others.
As illustrated in Figure 3.2 (adopted from McGrath [85]), strategies within the different categories have different strengths and weaknesses with respect to the criteria (A), (B) and (C). The figure shows that the three properties are far from each other on the circle. Thus, strategies in the middle of the dark grey area score high on precision, but low on generality and realism. In order to obtain both generality, realism and precision, it is necessary to choose several strategies that complement each other. A given category may contain strategies with different strengths and weaknesses. For example, setting-independent strategies may be either strong on generality or strong on precision depending on their depth and width.

In the following, we summarize the most common strategies and discuss their strengths and weaknesses with respect to the criteria listed by McGrath.

- **Laboratory experiments** are attempts to represent the essence of some general class of systems by controlling the extraneous features of such systems. Laboratory experiments score high on precision but lack realism and generality.

- **Benchmarking** is a quantitative method for evaluating performance related features. Benchmarking has the benefit of being controllable and therefore easy to replicate, but is weak on generality and realism.

- **Field studies** refer to efforts to make direct observations of running systems. Field studies are strong on realism but lack precision and generality as they are difficult to replicate.

- **Field experiments** are similar to field studies, with one major difference; the deliberate manipulation of some feature whose effects are to be studied. This method might be applied in a computer science setting for example by observing the effects of deliberately increased workload of the computer system in a certain usage scenario. It is more realistic than laboratory experiments and simulations, but lacks precision and generality.
• **Case studies** involve an in-depth examination of a product or a setting: a case. Kitchenham [69] refers to a case study as an evaluation of a method or tool that has been used in practice on a real project. A case study can be seen as a variant of a field experiment and has similar strengths and weaknesses. The technical definition of a case study is, according to Yin [128], as follows: “A case study is an empirical inquiry that: investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident”. The case study inquiry copes with the technically distinctive situation in which there will be many more variables of interest than data points, and as one result; relies on multiple sources of evidence, with data needing to converge in a triangulating fashion, and as another result; and benefits from the prior development of theoretical propositions to guide data collections and analysis. A case study method is, according to Yin [128], used when the researcher deliberately wants to uncover contextual conditions – believing that they might be highly pertinent to the phenomenon of study. A case study comprises an all-encompassing method – covering the logic of design, data collection techniques, and specific approaches to data analysis.

• **Action research** is a mixture of a field experiment and a case study where the researcher takes part in the case being studied. It is a method often used to improve processes or systems. The involvement of the researcher may give an even deeper understanding of the case or situation, but with the sacrifice of generality and the risk of observations being subjective.

• **Sample survey** refers to the gathering of information from a broad and well-devised sample of respondents through the use of questionnaires or interviews. A survey is less controlled than an experiment and therefore lacks precision. However, it scores high on generality, and might be used in computer science to assess how certain properties of an artifact are considered by a population as a whole. There is, however, a likelihood of bias on the part of the respondents, which weakens the realism of a survey. The advantage of surveys is that they allow the investigation of a greater number of variables than laboratory experiments. Moreover, since they are independent of the setting they have a high degree of generality.

• **A literature review** examines existing publications related to a topic and a scope. The method is often used to identify the current state of art and state of practice within a field. A literature review has the benefit of a large publication database which most often is available, but a literature review may be biased in the selection of published works.

• **Qualitative screening** is a feature-based evaluation performed by an individual who determines the features to be assessed on a product. This approach can also be called proof-of-concept. It demonstrates that a particular configuration of ideas or an approach achieves its objectives. It may be difficult to evaluate the value of the artifacts serving the role of a proof-of-concept, because the finding may be qualitative, for example increased functionality. In this context, what is considered better depends on subjective judgments.
• **Judgment studies** are efforts to get responses from a selected sample of “judges” about a systematically configured set of stimuli. Judgment studies are considered to score high on precision, but low on generality.

• **Formal analysis.** Formal analysis refers to a non-empirical argument based on logical reasoning. It scores high on generality, but low on realism and precision.

• **Simulation** attempts to model a specific real-life system or a class of such systems. It is a study in which an effort is made to represent a system that is sufficiently similar to some class of naturally occurring systems. It scores low on precision. Since it is system-specific, it scores higher on realism than formal analysis, but lower on generality.

When selecting evaluation strategies, there are several aspects to consider [69,113]:

• The nature of the predictions – do they concern quantitative features, qualitative features or formal features of an artifact. The three cases can be evaluated using quantitative methods, qualitative methods (such as for example a case study), and a formal analysis, respectively.

• The maturity of the artifact. For example, is there sufficient information to conduct a survey, or is only a case study feasible in an early phase of a research project?

• Is the strategy realizable? Aspects such as time, cost and the possibility of involving people should be considered. For example, computer simulations do not involve humans. They are therefore quick and easy to perform, if it is possible and relevant.

3.3 **How we have applied the research method**

The method applied in the research leading up to this thesis is based on the technology research method described in Section 3.1. The work has been conducted as an iterative process in which the artifacts and the success criteria for them have been changed and improved as new insight was gained while the work progressed.

Part I of this thesis documents the three phases of the research process illustrated in Figure 3.1. Based on an initial problem analysis, we identified the need for a model-based method for prediction of effects of architectural design changes on system quality. Our overall goal is to contribute to the development of such a method. We also identified a set of tasks necessary to fulfill the overall goal, such as defining a process and a modeling approach. Moreover, we refined the goal into a set of success criteria for the identified tasks. Chapter 2 documents the problem analysis and the success criteria for artifacts. The results of the innovation effort are described in Chapter 5, as the invented artifacts. Finally, Chapter 7 documents the evaluation of the artifacts.

Figure 3.3 illustrates in more detail the process which was followed in the development of each new artifact. In the case of each new artifact, we first identified the success criteria for the artifact. The success criteria were based on our goal for developing the new artifact. Then, we initialized development of an artifact with the aim to satisfy the success criteria. Finally, we evaluated the invented artifact with respect
3.3 How we have applied the research method

![Flowchart](figure.png)

Figure 3.3: Research method for development of artifacts (illustration adopted from Refsdal [104] and modified)

...to the identified success criteria. As shown in Figure 3.3, we mainly performed four kinds of evaluation.

1. Checking the artifact against results of case studies. This manner of evaluation is applicable for instance when proposing a process, a modeling approach, or testing feasibility of a new method. A method, a modeling language or a process can not be proven correct even though it is precise and unambiguous. Instead, it has to be checked qualitatively through case studies if it is useful, comprehensible and practically scalable.

2. Checking the artifact against results of applying examples. This manner of eval-
3. Checking the artifact against results of thought experiments. Thought experiments can for example be conducted as a part of an empirical trial. Through a thought experiment, a group of domain experts simulates a specified setting and estimates the outcome. The estimate is assumed to be sufficiently close to the reality. Then, the results obtained using the artifacts are checked against the thought experiment-based estimates.

4. Checking the artifact against findings of literature reviews. A literature review can be conducted to first provide an overview of the properties of the existing artifacts within a pre-defined scope. Then, the results of a literature review can be used to evaluate a new artifact against the alternative existing artifacts (based on their publications), with respect to the desirable properties. Alternatively, the results of a literature review can be used to deduce the needs for a new artifact, based on the (missing) properties of the existing ones.

The process of evaluation not only led to new insight about the artifact itself, but also about the success criteria for the artifact. Hence, as indicated in Figure 3.3, it was sometimes necessary to revise the success criteria for the artifact in light of the insight gained from evaluation.

The process illustrated in Figure 3.3 was applied for each of the four artifacts developed in this thesis. In the sequel, we describe in more detail how each of the four artifacts was evaluated.

### 3.3.1 The process of the PREDIQT method

The main objective of the process of the PREDIQT method is its applicability in an industrial context and within acceptable resources, provided that a PREDIQT-based analysis results in useful predictions. The process itself should also facilitate knowledge management in terms of exchanging, documenting and consolidating the knowledge regarding system architecture and system quality. The process was initially proposed prior to the first case study presented in Paper 1. Feasibility of the process was evaluated through a case study on a relatively complex real-life system for validation of electronic identifiers (signatures and certificates). The case study covered all steps of the process of the PREDIQT method, and was conducted in a fully realistic setting. The experiences from the case study and a lightweight postmortem evaluation [41], provided the input for revising both the success criteria for the process and the process specification. As a result, the success criteria for the process became more concrete and the process specification was provided more details.

With the revised success criteria and a more detailed process specification, a second case study was conducted in a different domain and on an even more complex and mature system. Again, all stages of the process of the PREDIQT method were undergone. The research method was motivated by the guidelines for case study research
3.3 How we have applied the research method

<table>
<thead>
<tr>
<th>Case study design</th>
</tr>
</thead>
<tbody>
<tr>
<td>The research question</td>
</tr>
<tr>
<td>Units of analysis</td>
</tr>
<tr>
<td>Success criteria</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PREDIQT-based analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment</td>
</tr>
<tr>
<td>Evaluation of predictions</td>
</tr>
<tr>
<td>Written feedback after the analysis</td>
</tr>
<tr>
<td>Verbal feedback during the analysis</td>
</tr>
<tr>
<td>Observations made during the analysis</td>
</tr>
</tbody>
</table>

Figure 3.4: Main stages of the research method applied on the second case study

provided by Yin [128]. A deductive approach was undertaken, where the pre-defined process of the PREDIQT method was exposed to a case study in a realistic setting. The main stages of the research method are depicted by Figure 3.4. The case study design included characterization of research question, the units of analysis and the success criteria, as the main outcomes. The success criteria deduced addressed concerns such as cost-effectiveness, comprehensibility and knowledge management. The PREDIQT-based analysis was first performed by following the pre-defined process of the PREDIQT method. However, instead of performing predictions of effects of future changes during the last workshop (as specified by the PREDIQT process), we chose to demonstrate how prediction models can be applied by predicting the effects of reversing of a very large already implemented change. As such, the model application phase is not fully covered, but only demonstrated. The affected Design Model elements and DV elements were identified and their modified parameter values estimated by the domain experts. Thereafter, the change propagation on the DVs was performed by the analyst. Additionally, in order to evaluate the predictions obtained, a thought experiment regarding the effect of the change on the root nodes of the respective DVs, was performed by the domain experts. Thus, this was a part of the method assessment. The overall assessment measures included:

- Written feedback from the analysis participants (affiliated with the customer organization) provided (through an evaluation template) upon completion of the analysis and the above mentioned thought experiment-based evaluation. The evaluation template was designed with the aim to address the pre-defined success criteria.

- Verbal feedback during the analysis from the analysis participants.

- Observations made by the analyst during the analysis.

Based on the results of the PREDIQT-based analysis and the assessment, an evaluation with respect to the success criteria was provided. The details of the case study design and deduction of the success criteria are presented in Paper 4.
Based on the experiences from the two case studies, detailed guidelines for the analyst regarding 1) input acquisition and 2) application of prediction models, are specified and documented in Papers 2 and 5, respectively.

3.3.2 The approach to modeling in PREDIQT

The objective of the approach to modeling in PREDIQT is to offer useful and practically applicable support for modeling dependencies between system architecture and quality characteristics of a system. The approach to modeling in PREDIQT proposes a structure of the prediction models but focuses particularly on syntax, semantics and pragmatics of the DVs. Thus, Design Models, Quality Models and the DVs are a part of the approach to modeling in PREDIQT. It was however early realized that the first two kinds of models could be based on the existing modeling notations.

A Design Model represents the relevant aspects of system architecture, such as for example process, dataflow, structure and rules. The Design Models can be based on the existing notation, such as UML [108]. The Quality Models could also be based on the UML profile for quality of service modeling [11], Goal/Question/Metric [20,21], or ISO standard for product quality [61]. Thus, for the Design Models and the Quality Models, the existing languages and notations could be adopted. However, for the development and use of DVs and specification of their relationship with the overall prediction models (Design Models and the Quality Models), we had to come up with a novel approach. This need emerged due to lack of an existing scalable language for modeling of such dependencies at system level and with respect to quality characteristics. That is, a language which is both practically applicable (in terms of comprehensibility, scalability of the analysis and the effort needed), as well as sufficiently expressive and fine-grained to provide useful predictions when used by an analyst and domain experts in a practical setting.

Thus, in evaluating the different success criteria of the approach to modeling in PREDIQT, we addressed concerns such as: expressiveness of the models, comprehensibility of the models, correctness of the predictions obtained, and scalability of the modeling approach in a realistic setting. The evaluation of the approach to modeling in PREDIQT was based on:

- Experiences obtained from the first case study presented in Paper 1. The DV-based modeling approach was for the first time outlined and applied during this case study. It was chosen due to its simplicity relative to the alternative approaches. To offer visualization, sensitivity analysis and automatic value propagation, the DV tool support was also developed and used during the first case study. Through the PREDIQT-based analysis itself, concerns such as feasibility and comprehensibility were evaluated.

- Thought experiment conducted as a part of first case study presented in Paper 1. The predictions obtained from the prediction models were evaluated against thought experiment performed by a group of domain experts. By comparing the values obtained from the predictions (based on the prediction models) to the thought experiment-based ones, both the parameter values, the structure of the models and the propagation model were addressed.

- Experiences from applying the modeling approach (including the DV tool) during the second case study presented in Paper 4. The evaluation was based on
3.3 How we have applied the research method

the PREDIQT-based analysis and the post-mortem evaluation. The former addressed scalability and comprehensibility of the modeling approach, while the latter included a thought experiment which evaluated the predictions. The post-mortem evaluation also included written feedback which addressed both usefulness and applicability of the approach to modeling in PREDIQT.

- A systematic literature review. The evaluation was based on a set of specified evaluation criteria and addressed usefulness and practical applicability of the alternative approaches for modeling dependencies between system architecture and quality characteristics of a system, as well as uncertainty handling in them. PREDIQT proposes the DV-based approach for modeling dependencies between system architecture and quality characteristics of a system. There are however existing alternatives to DVs, which score differently on the various criteria extracted from PREDIQT. The various contexts of the PREDIQT-based analyses may favor the different criteria to varying degrees. Therefore, the evaluation surveys the alternatives and their performance with respect to such criteria. The research method of this study is outlined in the next subsection. The evaluation and the details of the research method are presented in Paper 3.

The evaluation of the approach to modeling in PREDIQT from the first case study indicated need to support handling of uncertainties in the DVs, in order to: 1) offer representation and propagation of the uncertain empirical input, and 2) to assign uncertainty to predictions so that their usefulness could be interpreted and so that the value of added information also could be deduced. This need is addressed through the approach to uncertainty handling in PREDIQT.

The evaluation of the approach to modeling in PREDIQT from the second case study indicated need for improved tool support in order for the analyst to easily change the model structure. Moreover, the need to relate parts of the DVs to their rationale, assumptions, interpretations and relationships to the overall prediction models, was identified. This need is partly addressed through the approach to traceability handling in PREDIQT. Although the existing DV tool does not solve this need, the traceability approach in PREDIQT is supported by a prototype tool and addresses needs for tracing beyond the DVs.

3.3.3 The approach to uncertainty handling in PREDIQT

The evaluation based on systematic literature review (presented in Paper 3) examined the uncertainty handling in the existing approaches for modeling of dependencies through weighted dependency trees (WDTs). The rationale for focusing on WDTs is that the WDTs are widely used in approaches to system analysis. Moreover, any directed acyclic graph (DAG) may be represented as a general WDT, by duplicating the nodes in the DAG. The overall process was divided into two stages.

Firstly, a literature review was conducted. The literature review consisted of a mapping study (that is, a search in several digital resources, based on pre-defined keywords) and a systematic literature review. The relevant publications related to the state of the art with respect to the uncertainty handling approaches, were identified. The main contents of the relevant publications were extracted, indicating the potentially useful approaches for uncertainty handling and their main properties with respect to a pre-specified objective. The systematic literature review provided a preliminary evaluation
and an initial list of the evaluation criteria. Based on the results of the systematic literature review and the experienced needs for the evaluation, the evaluation criteria were iteratively refined through multiple discussions among the researchers. The deduction of the criteria was partially guided by an existing taxonomy for system acceptability.

Secondly, an evaluation of the approaches identified and analyzed throughout the systematic literature review was conducted. The evaluation consisted of two stages: a high-level evaluation and a low-level evaluation, respectively. The high-level evaluation classified the approaches identified and summarized their main properties. The classification was organized according to the kind of the approach (e.g., bayesian, fuzzy, interval-based, hybrid, etc.). Moreover, each classified kind of the approach was assessed as to whether it should be further evaluated as a part of low-level evaluation, that is in the context of the WDTs and with respect to the refined evaluation criteria.

In the low-level evaluation each approach was instantiated on a WDT, in order to test its applicability. Section 3 as well as Appendices A and B of Paper 3, report on the details of the research method regarding the literature review and the evaluation of the approached for uncertainty handling on WDTs.

Based on the practical experiences from the first case study, success criteria for an uncertainty handling approach in PREDIQT, were deduced. The success criteria addressed both precision and the practical applicability (in terms of scalability and comprehensibility) of the uncertainty handling approach in PREDIQT. The evaluation based on systematic literature review indicated a need for a novel uncertainty handling approach in PREDIQT, such that the pre-defined success criteria for uncertainty handling in PREDIQT, could be fulfilled. Subsequently, the interval-based approach (presented in Paper 2) was proposed. Its evaluation was based on demonstrating its use on an example, as well as by applying the approach on prediction models from the first case study (presented in Paper 1). The application of the approach on an example allowed demonstration of fulfillment of the basic functional properties. The application of the approach on prediction models from the first case study allowed evaluation of applicability, particularly scalability and practical usability of the approach, on a representative example.

Moreover, we have evaluated feasibility of the alternative uncertainty handling approaches with respect to our success criteria (for uncertainty handling in PREDIQT) and argued that the approach we have proposed is most applicable in the PREDIQT context. The approach also includes guidelines for the analyst, for acquiring the empirical input along with its uncertainty estimates. The guidelines are based on the practical experiences obtained during the two case studies. Our uncertainty handling approach and the evaluation of it with respect to the alternative uncertainty handling approaches, are presented in Paper 2.

### 3.3.4 The approach to traceability handling in PREDIQT

The practical experiences from the two case studies suggested that important preconditions for model-based prediction are correctness and proper usage of the prediction models. The process of the PREDIQT method guides the development and use of the prediction models, but the correctness of the prediction models and the way they are applied are also highly dependent on the creative effort of the analyst and his/her helpers. Therefore, we deduced the PREDIQT guidelines (as documented in Appendix 2 of Paper 5), and used them as a starting point for deriving the specific needs for
3.3 How we have applied the research method

traceability support in PREDIQT. The guidelines cover the “Application of prediction models”-phase and instruct the analyst on how to use the prediction models. The guidelines also identify where the traceability support is needed, and propose the use of it. The guidelines are based on the experiences from the two case studies. As such, the guidelines are not exhaustive but serve as an aid towards a more structured process of applying the prediction models and accommodating the trace information during the model development.

Upon having deduced the needs for traceability support, success criteria for a traceability handling approach in PREDIQT were specified. The success criteria addressed both accuracy and the practical applicability of the traceability handling approach in PREDIQT. Based on the success criteria, traceability handling approach for PREDIQT was defined by a traceability scheme. The traceability scheme for PREDIQT is basically a feature diagram specifying capabilities of the solution and a meta-model for the trace-link information. A prototype traceability tool was implemented in order to address the success criteria regarding practical applicability, as well as functionality for searching for trace-links and reporting the search results. The application of the approach was illustrated on parts of prediction models from the first case study (presented in Paper 1). The traces were both documented and applied, and the prototype traceability tool was used to support both recording and retrieval of the traces. The application of the approach on a representative example allowed demonstration of fulfillment of the basic functional properties, as well as partial evaluation of its practical applicability.

Moreover, we have conducted a literature review of the existing traceability handling approaches. Based on the literature survey, we have evaluated feasibility of the existing traceability handling approaches with respect to our success criteria (for traceability handling in PREDIQT) and argued that the approach we have proposed is most applicable in the PREDIQT context. Our approach to traceability handling in PREDIQT, the above mentioned guidelines, the results of the application of the approach, and a summary of the literature review based evaluation, are reported in Paper 5.
Chapter 4

State of the art

In this chapter, we first give an overview of state of the art related to prediction in general. Then, in Section 4.2 through Section 4.5 we present state of the art of relevance for the four artifacts developed in this thesis. For a more detailed discussion on the relationship of our artifacts to the literature, the reader is referred to Section 7.2 and to the related work sections in the attached papers.

4.1 Model-based prediction

In Chapter 2 we introduced the notion of model-based prediction. Model-based prediction is widely used in fields such as cost estimation, economics, control theory, weather forecasting, etc. The terminology, the goals and the approaches vary considerably among the different fields. Central in the model-based prediction is a prediction system which contains a prediction model and a set of procedures for determining the unknown values and interpreting the results. The prediction models represent the dependencies or the relationships among the relevant aspects or elements of the reality or the system which is being modeled. The elements and their relationships can be annotated by names and values (which may for example represent probability, frequency, strength, degree of dependency). The prediction models can generally be categorized into the ones based on causality or correlation. Regardless of the category, the prediction models aim to predict a future value of a function which has a pre-defined purpose and interpretation. Such a function is most often called an indicator. Input to the indicator is provided through one or more measurements. Moreover, in order to achieve a needed degree of accuracy, the prediction system needs to be validated. This section presents relevant literature on prediction, with focus on measurement, indicators, prediction models and their validation. For further details on metrics and measurement theory, we refer to Fenton and Pfleeger [49].

4.1.1 Measurement

Measurement is a widespread kind of empirical input that the models are based on. During quantitative estimation, such as for example estimation of system quality, certain indicators often need to be measured. Similarly, many organizations define a standard set of measurements, so that systems can be compared and contrasted. For this purpose, design of a measurement program [43, 49] has to be conducted. The measurement programs schematically describe how to determine the indicator values.
Comprehensible guidelines for analysis of measurement data based on statistical techniques, are provided by Fenton and Pfleeger [49].

As argued by Fenton and Pfleeger [49], “The representation condition requires every measure to be associated with a model of how the measure maps the entities and attributes in the real world to the elements of a numerical system. These models are essential in understanding not only how the measure is derived, but also how to interpret the behavior of the numerical elements when we return to the real world. But we also need models even before we begin the measurement process.”

In many cases, an attribute can be directly mapped to its value. But when there are complex relationships among attributes, or when an attribute must be measured by combining several of its aspects, a model of how to combine the related measures is needed. Thus, we distinguish between direct and indirect measurement. Direct measurement of an attribute or an entity involves no other attribute or entity, while the indirect measurement uses a model of several related direct measurement to deduce a value of an attribute. Indirect measurement is often useful in making visible the interactions between direct measurements [49]. Measurements can be made for the purpose of assessment and for the purpose of prediction. The distinctions between the two kinds of measurement is not always clear-cut [49]. Generally, measurement for the purpose of prediction always requires some kind of a model that relates the attributes to be predicted to some other attributes that are measurable.

4.1.2 Indicators

Measurement is tightly related to indicators, in the sense that an indicator represents a function with a pre-defined purpose and interpretation, which is evaluated through measurements. Indicators consist of metrics relating two or more measures and a baseline or expected result. Metrics are not only used to indicate the current performance of systems or processes, but also to predict future performance [24]. An indicator is either basic or composite. A basic indicator is measurable in terms a single value, for example the average time between generation of a given event. A composite indicator is the aggregation of two or more basic indicators. Composite indicators can be used in the context of prediction, by applying their aggregation model to obtain an aggregated value of the indicator.

Indicators should be valid, accurate and precise. Being valid involves that what is being measured is reflected by the indicator used. Accuracy addresses the deviation between real and measured value of the indicator. In order to be considered precise, variability among repeated measurements of the same indicator should be low [65].

The economic literature distinguishes between leading, coincident and lagging indicators. A coincident indicator reflects current state of what is being measured, while leading and lagging indicators reflect the states that exist respectively before or after a lag. A lagging indicator with a short lag time is preferred over one with a long latency period, since any needed response to an observed change can take place earlier. As argued by Jansen [62], it is important to recognize lagging indicators and, if they are used, to be prepared to handle the intrinsic delay and associated limitations.
4.1.3 Models

Depending on the kind of dependency modeling they are based on, models for quantitative prediction may generally be categorized into two kinds: statistical models which represent relationships in terms of correlation between attributes or entities being modeled, and causal models which represent relationships in terms of causal dependencies between attributes or entities being modeled.

In statistics, regression analysis [51] includes techniques for modeling and analyzing relationships between multiple variables. We distinguish between a dependent variable and one or more independent variables. Regression analysis is primarily used to understand how the independent variables are correlated with the dependent variable, and to explore the forms of these relationships. A regression function is used for this purpose. Regression analysis also models the variation of the dependent variable around the regression function. This variation is represented by a probability distribution. Multiple methods for carrying out regression analysis exist, for example linear regression and least squares regression. The performance of regression analysis methods in practice depends on the data that the analysis is based on, and how it relates to the regression approach being used. The choice of the regression approach, in turn, is based on the assumptions regarding the underlying data and particularly the data-generating process (which is often unknown). These assumptions are sometimes testable if a large amount of data is available. Regression analysis is widely used for prediction and forecasting. Regression models for prediction are often useful although they may not perform optimally. However, regression methods may in many cases give misleading results, especially when causality is present in the observational data.

Quantitative predictions can also be based on causal models. A causal model is an abstract model that describes the causal mechanisms of a system. The model must express more than correlation because correlation does not imply causation. A typical causal model consists of a set of nodes defined as being within the scope of the model being represented. Any node in the model is connected by an arrow to another node with which it has a causal influence – the arrowhead delineates the direction of this causal relationship. For example, an arrow connecting variables A and B with the arrowhead at B indicates a relationship where a qualitative or quantitative change in A may cause change in B. A Bayesian Belief Networks (BBNs) [87] presented in Section 4.3.3 is an example of a causal network with an explicit requirement that the relationships are causal.

Structural equation modeling (SEM) [56] is a statistical technique for testing and estimating causal relations using a combination of statistical data and qualitative causal assumptions. SEM is suited for both theory testing and theory development, since it allows both confirmatory and exploratory modeling. Confirmatory modeling is most often based on a hypothesis which is represented by a causal model. The model is then tested against empirical data to determine how well the model fits the data. Thus, relevant parts of the model must be measurable or possible to relate to empirical input. The causal assumptions embedded in the model often have falsifiable implications which can be tested against the data [27]. An initial model may require fitting (adjustment) in light of new evidence. SEM models can be developed inductively by first specifying an initial model and then using data to estimate the values of unknown parameters. Among the strengths of SEM is the possibility of constructing latent variables, that is, variables which are not measured directly, but estimated from several measured
variables. In such cases, latent variables are based on predictions which, if inconsistent, allow revealing the needs for fitting of model structure or parameter values. So-called factor analysis, path analysis and regression all represent special cases of SEM.

However, a model is not sufficient for obtaining the prediction. In addition to the model, we need a means for determining the model parameters, plus a procedure to interpret the results. Therefore, in addition to the model itself, a prediction system is necessary, as argued by Fenton and Pfleeger [49]. A prediction system consists of a mathematical model together with a set of prediction procedures for determining unknown parameters and interpreting results [78].

### 4.1.4 Validation

Validating a prediction system in a given environment is the process of establishing the accuracy of the prediction system by empirical means; that is, by comparing model performance with known data in the given environment [49]. The degree of accuracy acceptable for validation depends on kind of system and acceptance range. We distinguish between deterministic prediction systems (which always give the same output for a given input) and stochastic prediction systems (for which the output for a given input will vary probabilistically). An acceptance range for a prediction system is a statement of the maximum difference between prediction and actual value. As mentioned above, validation involves comparing predictions against known, empirical input. The empirical input may for example be based on measurements, expert judgments or thought experiments. Measurement is introduced in Section 4.1.1, while the latter two are briefly presented below.

The Delphi method is a structured communication technique, originally developed as a systematic, interactive forecasting method which relies on a panel of experts [77]. In the standard version of the Delphi method, the domain experts answer questionnaires in two or more rounds. After each round, a facilitator provides an anonymous summary of the forecasts from the previous round as well as the reasons the experts provided for their judgments. Thus, the experts are encouraged to revise their earlier answers in light of the replies of other members of their panel. It is assumed that during this process the range of the answers will decrease and the group will converge towards the “correct” answer. Finally, the process is stopped after a pre-defined stop criterion (e.g. number of rounds, achievement of consensus, stability of results) and the mean or median scores of the final rounds determine the results [107].

Moreover, a thought experiment [30] is a mental exercise which may consider a hypothesis or a theory. Thought experimentation is the process of employing imaginary situations in order to understand the way things actually are or how they will be in the future. The understanding comes through reflection upon this imaginary situation. Thought experiments are conducted within the imagination and never in fact. Regardless of their intended goal, all thought experiments display a patterned way of thinking that is designed in order to explain, predict and control events of interest. The common goal of a thought experiment is to explore the potential consequences of the hypothesis in question. Scientists use thought experiments prior to a real experiment, or when realistic experiments are impossible to conduct.
4.2 The process of the PREDIQT method

The Architecture Tradeoff Analysis Method (ATAM) [31, 66, 67] is a method aimed to support the designers in evaluating the strengths and weaknesses of a system. The ATAM method helps evaluate the current system architecture, and architectural design of a new system. ATAM aims to guide the designers to ask the right questions and select the right degree of quality for a specific system. ATAM process aims to help the developers to discover tradeoffs and sensitivity points. The tradeoff points are defined as the dependencies between attributes. The sensitivity points are the areas of the system that may be significantly impacted if system architecture is changed. The sensitivity points may be monitored by an architect each time a change to a system is considered on an architectural level.

There are multiple development methodologies that focus on the ability of a project to accommodate to changes. Agile and Rational Unified Process (RUP) are iterative development methodologies that focus on adapting to user requirements throughout the development process. An iteration is performed for each new feature added to the system. Iterative development implements just-in-time planning, in which decisions are made as changes occur. Some popular methods of agile development are Extreme Programming, SCRUM, Crystal, and FDD [46].

Lean Software Development (LSD) [99] is the application of lean principles to software development. LSD is more strategically focused than other Agile methodology. The goal is to develop software in one-third of: the time, the budget, and the defect rate. LSD is in itself not a development methodology, but it offers principles that are applicable for improving software development.

Six Sigma [100] is a customer-focused, data driven, and robust methodology for process improvement, which is well rooted in mathematics and statistics. A typical process for Six Sigma Quality Improvement has six phases: define, measure, analyze, improve, control and technology transfer. For each step, many quality improvement methods, tools and techniques are used. The process establishes deliverables at each phase and creates engineering models over time. Each of the six phases answers some target question towards a continuous improvement of the process.

Statistical Process Control (SPC) [122] involves using statistical techniques for measuring and improving the quality of processes. The intent of SPC is to monitor system quality and maintain processes to pre-defined targets. SPC is used to monitor the consistency of processes used to develop a system as designed. SPC can ensure that a system is being implemented as designed and intended. Thus, SPC will not improve a poorly designed system, but can be used to maintain the consistency between the implementation and the design. SPC uses mainly control charts, a graphical representation of certain descriptive statistics, for quantitative measurements of the process. The measurements are compared to the goals and the comparison detects unusual variation in the development process. Control charts also show system measurements and are used to analyze process capability towards continuous process improvement.

Some development processes are more mature than others, as noted by Software Engineering Institute’s (SEI’s) reports on process maturity and capability maturity [57]. The SEI has suggested five levels of process maturity: ad hoc (the least predictable and controllable), repeatable, defined, managed and optimizing (the most predictable and controllable). The SEI distinguishes one maturity level from another in terms of key process activities going on at each level. Capability Maturity Model Integration
(CMMI) [34] is a well-known and standardized framework for model-based assessment and improvement of system development processes. That is, CMMI is a model for planning and guidance of improvement of system development processes. In the same way that models are used to guide analysis on how to build systems, CMMI is used to guide system development processes. Other established process improvement models include ISO 9000, SPICE and Bootstrap [49]. All of these models share a common goal and approach, namely that they use process visibility as a key factor that distinguishes the different maturity levels. Thus, the more visibility in the overall development process, the higher the maturity and the better the process can be controlled and understood.

4.3 The approach to modeling in PREDIQT

This section presents state of the art on the three modeling perspectives relevant for the prediction models: modeling of system architecture, modeling of quality and dependency modeling.

4.3.1 Architectural design modeling

Model-driven architecture (MDA) is a framework for software development defined by the Object Management Group (OMG) [6]. The de facto modeling language of MDA is the Unified Modeling Language (UML) [53]. UML is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system. UML was developed in an effort to consolidate a large number of object-oriented development methods [108].

Business Process Execution Language (BPEL) [98] is an open and standardized language used to connect, organize and orchestrate web services according to a workflow. Being an orchestration language, BPEL specifies an executable process that involves message exchange with collaborating systems, such that the message exchange sequences are specified and controlled by the orchestration designer.

There is no standard graphical notation for BPEL, so some vendors have invented their own notations. These notations take advantage of the fact that most constructs in BPEL are block-structured. Others have proposed use of a substantially different business process modeling language, namely Business Process Modeling Notation (BPMN) [98]. BPMN is a standardized graphical notation for drawing business processes in a workflow.

The Open Group Architecture Framework (TOGAF) [9] is one of the most known enterprise architecture frameworks. TOGAF provides a comprehensive approach for designing, planning, implementation, and governance of an enterprise architecture. TOGAF is a holistic approach to design, which is typically modeled at four levels: business, application, data, and technology. It aims to give a high level starting model to information architects. TOGAF relies on modularization, standardization and already existing technologies.

4.3.2 Quality modeling

The software product quality standard ISO 9126 [61] provides an established specification of decomposed quality notions with their qualitative and quantitative definitions.
In the software evaluation process, the ISO 9126 addresses the quality model, external metrics, internal metrics and the quality in use metrics. This international standard defines six characteristics that describe, with minimal overlap, software quality. The characteristics of the software quality are categorized into: functionality (including suitability, security, compliance, accuracy and interoperability), reliability (including maturity, recoverability and fault tolerance), usability (including learnability and operability), efficiency (including quality based on resource and time), maintainability (including stability, changeability and testability) and portability (including installability and adaptability). ISO 9126 defines a quality model which is applicable to software, but it does not provide guidelines for how to specify the requirements for the software.

UML Profile for Modeling Quality of Service (QoS) and Fault Tolerance (FT) Characteristics and Mechanisms [11] defines a set of UML extensions to represent quality of service and fault-tolerance concepts. The extensions facilitate description of quality of service and fault-tolerance properties in UML 2.0. The profile integrates two frameworks (QoS Modeling Framework, and FT Modeling Framework) into a general framework. The general framework supports describing vocabulary that is used in high quality technologies, such as real-time and fault-tolerant.

The goal/question/metric (GQM) [20, 21] is an approach for specification of the quality goals and their metrics. It provides guidance and notation for decomposition of the quality notions. GQM is goal-driven and facilitates developing and maintaining a meaningful metrics program that is based on three levels, Goals, Questions and Metrics. The approach uses metrics to improve the software development process (and its resulting software systems) while maintaining compliance with business and technical goals of the organization.

### 4.3.3 Dependency modeling

Heyman et al. [55] introduce an approach for decomposing security objectives into security patterns, which again are decomposed into security measures, which serve as indicators. The resulting dependency graphs are developed for the security quality attribute. Pattern based quantitative security assessment is presented by Yautsiukhin et al. [126], where threat coverage metrics are associated with security patterns and aggregated to an overall security indicator.

Bayesian Belief Networks (BBNs) [54, 87] are quantitative probabilistic networks that can be used to illustrate the relationships between a system state and its causes. A BBN is a directed acyclic graph in which each node has an associated probability distribution. Observation of known variables (nodes) allows inferring the probability of others, using probability calculus and Bayes theorem throughout the model (propagation). BBNs are however demanding to parametrize and interpret the parameters of, since the number of estimates needed is exponential with respect to the number of nodes. The application of BBNs is extensive. Application of the BBNs on large scale systems has been reported by Fenton and Neil [48]. The reported empirical evaluation of the Bayesian Belief Networks is promising.

In the context of traditional dependability analysis, there exist various tree-based languages and notations which support the different phases of the analysis and documentation process. These notations typically use a certain set of constructs to specify how a set of circumstances can lead to a state, or what consequences a state can lead
to. In the sequel, we outline some of the most known ones of such notations.

A goal tree \([33,125]\) is a representation for modeling how goals are achieved. Well-defined inference procedures for finding “optimal” solutions from a goal tree can be used in finding the preferred way to satisfy a requirement. The goals can correspond to requirements, issues or decisions. Each of these perspectives can be represented by a goal tree. The nodes of a goal tree are of two types: OR nodes which represent choices, and AND nodes which represent simultaneous subgoals that must have compatible solutions.

Most known of the tree-based notations are fault trees and event trees. The fault tree \([58]\) notation is used in fault tree analysis (FTA) to describe the causes of an event. Fault trees are well known and widely used in security analysis. The notation enables specifying the order of events towards an unwanted incident represented by the top node. The children nodes represent different events that can lead to the event represented by the top node. The children nodes are related to the top node by logical AND and OR gates. Fault trees allow both qualitative and quantitative analysis of the events which lead to an unwanted incident. Probabilities can be assigned to the events.

Event trees \([59]\) of event tree analysis (ETA) have the opposite intuition compared to the fault trees – from an initial unwanted incident, the consequences of the incident are modeled. A binary tree is formed during the modeling, so that each branching point is a distinction between presence or non-presence of an event. As a result, the construction of an event tree faces the risk of size explosion. The probability of each consequence is calculated by aggregating the probabilities of the successes and failures that lead to this consequence. Similarly to fault trees, the event trees allow both qualitative and quantitative analysis.

An attack tree \([83]\) is a variant of fault tree. Attack trees aim to provide a schematic way of modeling the attacks a system may be exposed to. The notation uses a tree structure similar to fault trees, with the attack goal as the top node and different ways of reaching the attack goal as leaf nodes. Since the attack tree represents different ways of attacking the system, the focus is on human behavior rather than system behavior.

A cause-consequence diagram \([105]\) combines the properties of both fault tree and event tree. Starting from a certain state, the causes are identified in the manner of a fault tree and the consequences are modeled in the manner of an event tree. Cause-consequence diagrams are qualitative and cannot be used as a basis for quantitative analysis \([102]\).

### 4.4 The approach to uncertainty handling in PREDIQT

The empirical input is always associated with a degree of uncertainty. Uncertainty is generally categorized into two different types: aleatory (due to inherent randomness of the system or variability of the usage profile) and epistemic (due to lack of knowledge or information about the system) \([70]\).

The aleatory uncertainty is a property of the system associated with variability. It is irreducible even by additional measurements. Aleatory uncertainty is typically represented by continuous probability distributions and forecasting is based on stochastic models. Epistemic uncertainty, on the other hand, is reducible, non-stochastic and of a discrete nature. It is considered as uncertainty which may be originating from a range
4.4 The approach to uncertainty handling in PREDIQT

of causes that defy pure probabilistic modeling. Examples of such causes are lack or inaccuracy of input, which impedes the specification of a unique probabilistic model. Epistemic quantities have fixed values which may be unknown or poorly known. For example size or cost of an existing system are values which are fixed and existing, but may be difficult to reveal or deduce.

As opposed to, for example, weather forecasting models which are of stochastic and continuous nature and where the aleatory uncertainty is the dominating one (due to uncontrollable variabilities of many simultaneous factors), models of deterministic artifacts are characterized by rather discrete, sudden, non-stochastic and less frequent changes. As a result, the aleatory uncertainty is, in the deterministic artifacts, negligible in terms of magnitude and impact, while the epistemic one is crucial. It is therefore the epistemic uncertainty we focus on when dealing with the deterministic artifacts.

The Dempster-Shafer structures [50] offer a way of representing uncertainty quantified by mass distribution functions. A mechanism for aggregation of such representation stored in distributed relational databases, is proposed by Scotney and McClean [109]. The Dempster-Shafer approach characterizes uncertainties as intervals with degrees of certainty (that is, sets of values with weights which add up to one). It can be seen as a generalization of both interval analysis and probability theory. Weights of evidence are put on a collection of intervals and the structures may overlap. Dempster-Shafer evidence theory is a possible approach for epistemic uncertainty analysis. It relaxes the assumptions of the probability theory and allows for combining conflicting or ambiguous evidence from multiple sources. In probability theory, however, evidence is associated with only one possible source or event. Dempster-Shafer evidence theory models the epistemic uncertain input variables as sets of intervals. Each variable may be defined by one or more intervals. The user specifies a probability assignment to each interval. The probability assignment indicates how likely it is that the uncertain input falls within the interval. The probability assignments for a particular uncertain input variable must sum to one. The intervals may overlap, chain or gap relative to one another. Dempster-Shafer has two measures of uncertainty, belief and plausibility. The intervals are propagated to calculate belief and plausibility. Together, belief and plausibility define an interval-valued probability distribution. Dempster-Shafer interval calculation is computationally expensive due to the number of the interval combinations (within and across each variable) to be included in the propagation. Minimum and maximum function values are searched for within each interval combination. The minimum and maximum values are aggregated to create the belief and plausibility. The accuracy of the Dempster-Shafer approach depends on the number of samples and the number of interval combinations.

Another kind of approach to uncertainty handling is the interval-based one. Definitions of intervals, their arithmetics and central tendency operators are provided by Ferson et al. [50]. Additional comprehensive references on the interval-based approach are Kearfott [68] and Kreinovich et al. [75]. The practical interval solution depends on the monotonicity properties (the relationship between model input and output) of the model. The probability of occurrence of any value within an interval can follow any given arbitrary distribution. As argued by Majumdar et al. [80], interval arithmetics can serve as a tool to obtain interval extensions of real functions. However, due to a so-called overestimation effect, also known as dependency problem, interval arithmetic does not provide the exact range of a function. The dependency problem is due to the memoryless nature of interval arithmetic in cases when a parameter occurs multiple
times in an arithmetic expression, since each occurrence of an interval variable in an expression is treated independently. Since multiple occurrences of interval parameters cannot always be avoided, the dependency problem often causes overestimation of the actual range of an evaluated function. A way to handle this issue is to use interval splitting [80], where the input intervals are divided and the arithmetics are performed on the subintervals. The final results are then obtained by computing the minimum of all lower bounds and the maximum of all upper bounds of the intermediate results. Skelboe [112] shows that the results obtained from the interval splitting converge to the actual range when the width of the subintervals approaches zero. The application of this technique on performance models is reported by Majumdar and Ramadoss [81].

BBNs allow incorporating both model uncertainty and parameter uncertainty. BBNs can represent and propagate both continuous and discrete uncertainty distributions. In a BBN, probability distributions are calculated according to Baye’s rule, based on conditional probability tables (CPTs) and the probabilities assigned to the specified states on leaf nodes. The initial uncertainty is placed in the prior distribution of each input parameter. The prior distribution is then updated to a posterior distribution based on the observed data associated with each parameter. Additional data acquisition is undertaken until an acceptable certainty is reached. Bayesian approach for uncertainty analysis in software reliability modeling is for example applied by Yin and Trivedi [127]. A way of handling uncertainty in a BBN is by introducing states in each node and assigning a probability to each state. The uncertainty representation is in that case contained in both the granularity (the number of states on the nodes) and their respective probabilities. Then, Baye’s rule is applied for the propagation. The prior estimates consist of the probabilities associated with each state on the leaf nodes, as well as the dependencies estimated through the values in the conditional probability tables (CPTs). Due to the number of the estimates needed, the BBN-based approach has limited scalability. Furthermore, since the uncertainty representation of several states for each node requires extensive CPTs and a demanding propagation, the BBN-based approach suffers from low comprehensibility. The precision depends on the number of states introduced for each node, which means that for example new evidence of higher precision than the one defined through the existing states, may be difficult to represent. Hence, combining evidence of different uncertainty (for example, expert judgments and measurements) on a node may be difficult.

Fuzzy logic provides a simple way to draw definite conclusions from vague, ambiguous or imprecise information, and allows for partial membership in a fuzzy set. Fuzzy logic allows modeling complex systems using higher levels of abstraction originating from the analyst’s knowledge and experience [121]. A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership function, which assigns to each object a grade of membership ranging between zero and one [129]. Using the fuzzy membership functions, a parameter in a model can be represented as a crisp number, a crisp interval, a fuzzy number and a fuzzy interval. In fuzzy approach the algebraic operations are easy and straightforward, as argued and elaborated by Suresh et al. [115]. The fuzzy approach is expressive in terms of both estimate representation and arithmetics. There is a wide variety of arithmetic operators which can be selected and applied. Although the propagation is non-trivial, tool support is available. The precision level can be selected.

Subjective logic [63] is a framework for artificial reasoning, which consists of a belief model called opinion and set of operations for combining opinions. A single
opinion $\pi$ is uniquely described as a point \{b,d,i\} in an “Opinion Triangle”, where b, d and i designate belief, disbelief and ignorance, respectively. For each opinion, the three notions sum up to unity. The operations formally defined include: conjunction, disjunction, negation, consensus, recommendation and ordering.

In the imprecise probability theory [120], sets of probability functions capture the notion of partial lack of probabilistic information. The imprecise probabilities approach the probability theory through lower and upper probabilities, rather than probabilities. Cozman [39] presents an overview of graphical models that can handle imprecision in probability values. A review of the algorithms for local computation with imprecise probabilities is presented by Cano and Moral [32], where the objective is to carry out a sound global computation by mainly using the initial local representation. These algorithms try to solve problems of propagation (calculation of conditional or unconditional probabilities) in cases in which there is a large number of variables. There are two main types depending on the nature of the assumed independence relationships. In both of them the global knowledge is composed of several pieces of local information.

The ISO approach to handling measurement uncertainty [60] uses a probabilistic representation with normal distribution, and treats both aleatory and epistemic uncertainty equally. It also contains instructions on the propagation of the uncertainties. Such an approach however does not explicitly account the notion of ignorance about the estimates, thus failing to intuitively express it.

A simulation mechanism, which takes into account both aleatory and epistemic uncertainty in an interval-based approach, is proposed by Batarseh and Wang [22]. It concentrates on stochastic simulations as input for the interval estimates, when significant uncertainties exist. Moreover, Ferson et al. [50] propose considering a hybrid approach comprising both probabilistic and interval representation, in order to account for both aleatory and epistemic uncertainty.

A hybrid Monte Carlo and possibilistic method for representation and propagation of aleatory and epistemic uncertainty is presented by Baraldi et al. [19]. The method is applied for predicting the time to failure of a randomly degrading component, and illustrated by a case study. The hybrid representation captures the aleatory variability and epistemic imprecision of a random fuzzy interval in a parametrized way through $\alpha$-cuts and displays extreme pairs of the upper and lower cumulative distributions. The Monte Carlo and the possibilistic representations are jointly propagated. The gap between the upper and the lower cumulative distributions represents the imprecision due to epistemic variables. The possibility distributions are aggregated according to the so called Ferson method. The interpretation of the results in the form of limiting cumulative distributions requires the introduction of a degree of confidence directly connected with the confidence on the value of epistemic parameters.

4.5 The approach to traceability handling in PREDIQT

Traceability is the ability to determine which documentation entities of a software system are related to which other documentation entities according to specific relationships [72]. IEEE [10] also provides two definitions of traceability:

1. Traceability is the degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another; for
example, the degree to which the requirements and design of a given software component match.

2. Traceability is the degree to which each element in a software development product establishes its reason for existing.

IEEE [10] defines a trace as “A relationship between two or more products of the development process.” According to the OED [110], however, a trace is defined more generally as a “(possibly) non-material indication or evidence showing what has existed or happened”. A traceability link is a relation that is used to interrelate artifacts (e.g., by causality, content, etc.) [124]. In addition to the different definitions, there is no commonly agreed basic classification [124]. A taxonomy of the main concepts within traceability is suggested by Knethen and Paech [72].

Traces can exist between both model- and non-model artifacts. The means and measures applied for obtaining traceability are defined by so-called traceability schemes. A traceability scheme is driven by the planned use of the traces. The traceability scheme determines for which artifacts and up to which level of detail traces can be recorded [124]. A traceability scheme thus defines the constraints needed to guide the recording of traces, and answers the core questions: what, who, where, how, when and why. Additionally, there is tacit knowledge (such as why), which is difficult to capture and to document. A traceability scheme helps in this process of recording traces and making them persistent.

As argued by Aizenbud-Reshef [14], the first approach used to express and maintain traceability was cross-referencing. This involves embedding phrases like “see section x” throughout the project documentation. Thereafter, different techniques have been used to represent traceability relationships including standard approaches such as matrices, databases, hypertext links, graph-based approaches, formal methods, and dynamic schemes [14]. Representation, recording and maintenance of traceability relations are by Spanoudakis and Zisman [114] classified into five approaches: single centralized database, software repository, hypermedia, mark-up, and event-based.

According to Wieringa [123], representations and visualizations of traces can be categorized into matrices, cross-references, and graph-based representations. As elaborated by Wieringa, the links, the content of the one artifact, and other information associated with a cross reference, is usually displayed at the same time. This is however not the case with traceability matrices. So, compared to traceability matrices, the user is (in the case of cross-references) shown more local information at the cost of being shown fewer (global) links. As models are the central element in MDE, graph-based representations are the norm. A graph can be transformed to a cross-reference. Regarding the notation, there is, however, no common agreement or standard, mostly because the variety and informality of different artifacts is not suitable for a simple, yet precise notation. Requirements traceability graphs are usually just plain box-and-line diagrams [123].

Knethen and Paech [72] argue that the existing traceability approaches do not give much process support. They specify four steps of traceability process: 1) define entities and relationships, 2) capture traces, 3) extract and represent traces, and 4) maintain traces. Similarly, Winkler and Pilgrim [124] state that traceability and its supporting activities are currently not standardized. They classify the activities when working with traces into: 1) planning for traceability, 2) recording traces, 3) using
4.5 The approach to traceability handling in PREDIQT

Trace models are usually stored as separate models, and links to the elements are (technically) unidirectional in order to keep the connected models or artifacts independent. Alternatively, models can contain the trace-links themselves or define the links as bidirectional. While embedded trace-links pollute the models, navigation is much easier [124]. Thus, we distinguish between external and internal storage, respectively. Anquetil et al. [17] argue: “Keeping link information separated from the artifacts is clearly better; however it needs to identify uniquely each artifact, even finned-grained artifacts. Much of the recent research has focused on finding means to automate the creation and maintenance of trace information. Text mining, information retrieval and analysis of trace links techniques have been successfully applied. An important challenge is to maintain links consistency while artifacts are evolving. In this case, the main difficulty comes from the manually created links, but scalability of automatic solution is also an issue.”

As outlined by Aizenbud-Reshef [14], automated creation of trace-links may be based on text mining, information retrieval, analysis of existing relationships to obtain implied relations, or analysis of change history to automatically compute links.

Reference models are an abstraction of best practice and comprise the most important kinds of traceability links. There is nothing provably correct about reference models, but they derive their relevance from the slice of practice they cover. Nevertheless, by formalizing a reference model in an appropriate framework, a number of elementary desirable properties can be ensured. A general reference model for requirements traceability is proposed by Ramesh and Jarke [101], based on numerous empirical studies.

Various tools are used to set and maintain traces. Surveys of the tools available are provided by Knethen and Paech [72], Winkler and Pilgrim [124], Spanoudakis and Zisman [114], and Aizenbud-Reshef [14]. Bohner and Arnold [26] found that the granularity of documentation entities managed by current traceability tools is typically somewhat coarse for an accurate impact analysis.

Almeida et al. [16] propose an approach aimed at simplifying the management of relationships between requirements and various design artifacts. A framework which serves as a basis for tracing requirements, assessing the quality of model transformation specifications, meta-models, models and realizations, is proposed. They use traceability cross-tables for representing relationships between application requirements and models. Cross-tables are also applied for considering different model granularities and identification of conforming transformation specifications. The approach does not provide sufficient support for intra-model mapping. Possibility of representing the various types of trace-links and traceable elements is unclear, although different visualizations on a cross-table are suggested. Tool support for searching and reporting recorded traces, is not available.

Event-based Traceability (EBT) is another requirements-driven traceability approach aimed at automating trace-link generation and maintenance. Cleland-Huang, Chang and Christensen [35] present a study which uses EBT for managing evolutionary change. They link requirements and other traceable elements, such as design models, through publish-subscribe relationships. As outlined by Galvao and Goknil [52], “Instead of establishing direct and tight coupled links between requirements and dependent entities, links are established through an event service. First, all artifacts are registered

traces, and 4) maintaining traces. Traceability activities are generally not dependent on any particular software process model.
to the event server by their subscriber manager. The requirements manager uses its event recognition algorithm to handle the updates in the requirements document and to publish these changes as event to the event server. The event server manages some links between the requirement and its dependent artifacts by using some information retrieval algorithms. The notification of events carries structural and semantic information concerning a change context. Scalability in a practical setting is the main issue, due to performance limitation of the EBT server [52]. Moreover, the approach does not provide sufficient support for intra-model mapping.

Cleland-Huang et al. [38] propose Goal Centric Traceability (GCT) approach for managing the impact of change upon the non-functional requirements of a software system. Softgoal Interdependency Graph (SIG) is used to model non-functional requirements and their dependencies. Additionally, a traceability matrix is constructed to relate SIG elements to classes. The main weakness of the approach is the limited tool support. This limits both scalability in a practical setting and searching support.

Cleland-Huang and Schmelzer [37] propose another requirements-driven traceability approach that builds on EBT. However, it presents a different process for dynamically tracing non-functional requirements to design patterns. Although more fine-grained than EBT, there is no evidence that the method can be applied with success in a practical real-life setting. Searching and reporting facilities are not provided.

Many traceability approaches address trace maintenance. Cleland-Huang, Chang and Ge [36] identify the various change events that occur during requirements evolution and describe an algorithm to support their automated recognition through the monitoring of more primitive actions made by a user upon a requirements set. Mader and Gotel [79] propose an approach to recognize changes to structural UML models that impact existing traceability relations and, based on that knowledge, provide a mix of automated and semi-automated strategies to update the relations.

Ramesh and Jarke [101] propose another requirements-driven traceability approach where reference models are used to represent different levels of traceability information and links. The granularity of the representation of traces depends on the expectations of the stakeholders [52]. The reference models can be implemented in distinct ways when managing the traceability information. As reported by Galvao and Goknil [52], “The reference models may be scalable due to their possible use for traceability activities in different complexity levels. Therefore, it is unclear whether this approach lacks scalability with respect to tool support for large-scale projects or not. The efficiency of the tools which have implemented these meta-models was not evaluated and the tools are not the focus of the approach.” The reference models proposed by Ramesh and Jarke are of a broad scope and general, and their focus is on requirements traceability.

Tool support based on parts of reference models proposed by Ramesh and Jarke [101] is developed by Mohan and Ramesh [86] in the context of product and service families. Mohan and Ramesh discuss a knowledge management system, which is based on the traceability framework by Ramesh and Jarke [101]. The system captures the various design decisions associated with service family development. The system also traces commonality and variability in customer requirements to their corresponding design artifacts. The tool support has graphical interfaces for documenting decisions. The trace and design decision capture is illustrated using sample scenarios from a case study.

A modeling approach by Egyed [45] represents traceability information in a graph structure called a footprint graph. Generated traces can relate model elements with
other models, test scenarios or classes [52]. Galvao and Goknil [52] report on promising scalability of the approach. It is however unclear to what degree the tool supports searching and reporting, since semantic information on trace-links and traceable elements is limited.

Aizenbud-Reshef et al. [15] outline an operational semantics of traceability relationships that capture and represent traceability information by using a set of semantic properties, composed of events, conditions and actions [52]. Galvao and Goknil [52] argue that: the approach does not provide sufficient support for intra-model mapping; a practical application of the approach is not presented; tool support is not provided; however, it may be scalable since it is associated with the UML.

Limon and Garbajosa [76] analyze several traceability schemes and propose an initial approach to Traceability Scheme (TS) specification. The TS is composed of a traceability link dataset, a traceability link type set, a minimal set of traceability links, and a metrics set for the minimal set of traceability links [52]. Galvao and Goknil [52] argue that “The TS is not scalable in its current form. Therefore, the authors outline a strategy that may contribute to its scalability: to include in the traceability schema a set of metrics that can be applied for monitoring and verifying the correctness of traces and their management.” Hence scalability in a practical setting is limited. Moreover, there is no tool support for the employment of the approach. Thus, searching and reporting of the recorded traces is not supported.

Vanhooff and Berbers [118] have defined a UML profile that represents and supports transformation traceability links. Their approach allows the addition of semantically rich transformation traceability links into UML models, while keeping its consistency. The approach represents a traceability link using stereotype specifications and is independent of model transformation languages. The authors do not report on a tool that gives support to their approach.

Some approaches [47, 64, 74] that use model transformations can be considered as a mechanism to generate trace-links. Tool support with transformation functionalities is in focus, while empirical evidence of practical applicability of the approaches in a practical setting, is missing.
Chapter 5

Achievements: the overall picture

In this chapter we give an overview of our main artifacts and explain how they relate to each other. We also refer to the relevant chapters in Part II for definitions and further explanations.

5.1 The overall picture

As already explained, the objective of this thesis is to provide a useful and practically applicable method for model-based prediction of effects of architectural design changes on system relevant quality characteristics. Developing a complete method for system quality prediction is an extensive task and lies beyond the scope of this thesis. We have instead focused on four main artifacts, each of which is necessary within such a method:

1. The process of the PREDIQT method;
2. The approach to modeling in PREDIQT;
3. The approach to uncertainty handling in PREDIQT;
4. The approach to traceability handling in PREDIQT.

Figure 5.1 illustrates how our four main artifacts relate to each other. Artifact (1), the process of the PREDIQT method, uses artifact (2), the approach to modeling in PREDIQT. The approach to modeling in PREDIQT supports the process of the PREDIQT method, in relation to development, verification and application of the prediction models. The approach to modeling in PREDIQT provides the structure of the prediction models as well as syntax, semantics and rules for the Dependency Views (DV) – all necessary for modeling the dependencies between system architecture and system quality.

Artifact (1), the process of the PREDIQT method uses artifact (3), the approach to uncertainty handling in PREDIQT by applying the guidelines (for the analyst) for how to acquire the prior estimates and their uncertainty values. Although developed and documented as a part of artifact (3), the guidelines for estimate acquisition are considered to represent a refinement of the process of the PREDIQT method. Artifact (1) also uses artifact (3) for representation, propagation and analysis of uncertainty, during a PREDIQT-based analysis.
Figure 5.1: The four contributions and their relationship

Artifact (1), the process of the PREDIQT method uses artifact (4), the approach to traceability handling in PREDIQT, by applying the traceability approach during a PREDIQT-based analysis, in relation to recording and use of structured information about both relationships (links) between prediction models as well as the rationale and assumptions behind them. Moreover, the traceability approach includes structured guidelines for the analyst for how to apply the prediction models together with the traceability solution. The guidelines are considered to represent a refinement of the process of the PREDIQT method.

Moreover, the process of the PREDIQT method integrates the approach to uncertainty handling into the approach to modeling in PREDIQT, by incorporating the representation and propagation of uncertainties into the DVs. Therefore, the approach to modeling in PREDIQT indirectly uses the approach to uncertainty handling. This is represented by the dashed line between the approach to modeling in PREDIQT and the approach to uncertainty handling in PREDIQT on Figure 5.1. Although uncertainty handling is based on the needs of PREDIQT and particularly DVs, the approach to modeling in PREDIQT is not dependent on the uncertainty handling, and can be applied independently.

Similarly, the process of the PREDIQT method integrates the approach to traceability handling into the approach to modeling in PREDIQT, by systematically capturing, storing and using the trace information about the prediction models during a PREDIQT-based analysis. Therefore, the approach to modeling in PREDIQT indirectly uses the approach to traceability handling by assigning trace-link information to the prediction models and applying it. This is represented by the dashed line between the approach to modeling in PREDIQT and the approach to traceability handling in PREDIQT on Figure 5.1. Although traceability handling is based on the needs of PREDIQT and particularly prediction models, the approach to modeling in PREDIQT is not dependent on the traceability handling, and can be applied independently.

5.2 The process of the PREDIQT method

The purpose of the process of the PREDIQT method is to characterize and specify the phases to be undergone during a PREDIQT-based analysis. The process is fully documented in Paper 1, which includes a process overview, recommended organization
5.2 The process of the PREDIQT method

Phase 1: Target modeling
- Sub-phase 1: Characterization of the target and the objectives
- Sub-phase 2: Development of Quality Models
- Sub-phase 3: Mapping of Design Models
- Sub-phase 4: Development of Dependency Views

Phase 2: Verification of prediction models
- Sub-phase 1: Evaluation of prediction models
- Sub-phase 2: Fitting of prediction models
- Sub-phase 3: Approval of the final prediction models

Phase 3: Application of prediction models
- Sub-phase 1: Specification of a change
- Sub-phase 2: Application of the change on prediction models
- Sub-phase 3: Quality prediction

Figure 5.2: A simplified overview of the process of the PREDIQT method

The process of the PREDIQT method consists of three overall phases. Each phase is decomposed into sub-phases, as illustrated by Figure 5.2. The first phase involves development of the initial prediction models. The stakeholders involved deduce a high level characterization of the target system, its scope and the objectives of the prediction analysis, by formulating the system boundaries, system context (including the usage profile), system lifetime and the extent (nature and rate) of design changes expected. Quality Model diagrams are created in the form of trees, by defining the quality notions with respect to the target system. The Quality Model diagrams represent a taxonomy with interpretations and formal definitions of system quality notions. The total quality of the system is decomposed into characteristics, sub-characteristics and quality indicators. For each quality characteristic defined in the Quality Model, a quality characteristic specific DV is deduced from the Design Model diagrams and the Quality Model diagrams of the system under analysis. This is done by modeling the dependencies of the architectural design with respect to the quality characteristic that the DV is dedicated to, in the form of multiple weighted and directed trees. Each set of nodes having a common parent is supplemented with an additional node called “Other”, for completeness purpose. The DV parameters are assigned by providing the estimates on the arcs and the leaf nodes, and propagating them according to an inference algorithm for DVs, defined by the approach to modeling in PREDIQT.

The “Verification of prediction models” is an iterative phase that aims to validate the prediction models (with respect to the structure and the individual parameters), before they are applied. A measurement plan with the necessary statistical power is developed, describing what should be evaluated, when and how. Both system-as-is and change effects should be covered by the measurement plan. Model fitting is conducted
in order to adjust the DV structure and the parameters, to the evaluation results. The objective of the “Approval of the final prediction models” sub-phase is to evaluate the prediction models as a whole and validate that they are complete, correct and mutually consistent after the fitting. If the deviation between the model and the new measurements is above the acceptable threshold after the fitting, the target modeling is re-initiated.

The “Application of the change on prediction models” phase involves applying the specified architectural design change on the prediction models. The phase presupposes that the prediction models are approved. During this phase, a specified change is applied to the Design Model diagrams and the DVs, and its effects on the quality characteristics at the various abstraction levels are propagated on the respective DVs. The change specification should clearly state all deployment relevant facts, necessary for applying the change. When an architectural design change is applied on the Design Model diagrams, it is according to the definitions in the Quality Model, reflected to the relevant parts of the DV. Thereafter, the DV provides propagation paths and quantitative predictions of the new quality characteristic values, by propagating the change throughout the rest of each one of the modified DVs, based on the general DV propagation algorithm.

A PREDIQT-based analysis aims to facilitate the decision making prior to deployment of the changes. Thus, the intended application of the prediction models does not include actual system change, but only analysis of effects of the independent architectural design changes on system quality. Hence maintenance of prediction models as well as maintenance of trace information are beyond the scope of PREDIQT.

5.3 The approach to modeling in PREDIQT

The purpose of the approach to modeling in PREDIQT is to provide sufficient support for modeling dependencies between architectural design and quality of a system. To this end, PREDIQT develops and makes use of a structure of the above mentioned three subsets of the prediction models. For the Design Model, the approach to modeling in PREDIQT does not imply a specific notation as long as the Design Model is based on a notation that includes the expressiveness for modeling the architectural design of the target characterized. Similarly, for the Quality Model, the approach to modeling in PREDIQT does not imply a specific notation as long as the Quality Model is based on a notation that includes the above mentioned properties of a weighted dependency tree, whose parameters are defined qualitatively and quantitatively with respect to the target system. In the cases of Design Model and the Quality Model, the existing notations, approaches and standards such as UML, GQM and ISO 9126, are found to be applicable.

As for modeling of the dependencies of system quality upon system architecture, experiences from the case studies and results of an evaluation of the alternatives to the DV-based approach (with support for uncertainty handling, as reported in Paper 3) have indicated need for a novel approach which is practically useful and comprehensible. Therefore, the approach to modeling in PREDIQT offers the DV-based approach presented below. The approach to modeling in PREDIQT assumes Design Model and Quality Model to be provided as input, before development of DVs is initiated. Therefore, the approach to modeling in PREDIQT focuses mainly on DVs. The approach
5.3 The approach to modeling in PREDIQT

to modeling in PREDIQT and the results of its application on real-life systems, are
documented in Papers 1 and 4. In the following, we outline the structure of the pre-
diction models and the DV-based approach, and briefly present the tool support for
the DV-based modeling and analysis.

5.3.1 Structure of the prediction models

The PREDIQT method produces and applies a multi-layer model structure, called
prediction models, which represent system relevant quality concepts (through “Quality
Model”), architectural design (through “Design Model”), and the dependencies be-
tween architectural design and quality (through “Dependency Views”). The Design
Model diagrams are used to specify the architectural design of the target system and
the changes whose effects on quality are to be predicted. The prediction models cover
the aspects, the scope and the detail level characterized (during the first sub-phase of
the PREDIQT process) as a part of the objective of the analysis. A Design Model
represents the relevant aspects of system architecture, such as for example process,
dataflow, system structure and rules. The Quality Model diagrams are used to for-
malize the quality notions and define their interpretations. In addition, the prediction
models comprise DVs, which are deduced from the Design Models and the Quality
Models of the system under analysis. The DVs model the dependencies of the archi-
tectural design with respect to the quality characteristic that the DV is dedicated to,
in the form of multiple weighted and directed trees. The values and the dependencies
modeled through the DVs are based on the definitions provided by the Quality Model.
The DVs express the interplay between the system architectural design and the quality
characteristics. Once a change is specified on the Design Model diagrams, the affected
parts of the DVs are identified, and the effects of the change on the quality values are
automatically propagated at the appropriate parts of the DV.

Figure 5.3 provides an overview of the elements of the prediction models, expressed
as a UML [108] class diagram. A Quality Model is a set of tree-like structures which
clearly specify the system-relevant quality notions, by defining and decomposing the
meaning of the system-relevant quality terminology. Each tree is dedicated to a tar-
get system-relevant quality characteristic. Each quality characteristic may be decom-
posed into quality sub-characteristics, which in turn may be decomposed into a set
of quality indicators. As indicated by the relationship of type aggregation, specific
sub-characteristics and indicators can appear in several Quality Model trees dedicated
to the different quality characteristics. Each element of a Quality Model is assigned a
quantitative normalized metric and an interpretation (qualitative meaning of the ele-
ment), both specific for the target system. The Quality Model diagrams are used to
formalize the quality notions and define their interpretations.

A DV is a weighted dependency tree dedicated to a specific quality characteristic
defined through the Quality Model. As indicated by the attributes of the Node class,
the nodes of a DV are assigned a name and a QCF (Quality Characteristic Fulfillment).
A QCF is value of the degree of fulfillment of the quality characteristic, with respect to
what is represented by the node. The degree of fulfillment is defined by the metric (of
the quality characteristic) provided in the Quality Model. Thus, a complete prediction
model has as many DVs as the quality characteristics defined in the Quality Model.
Additionally, as indicated by the Semantic dependency relationship, semantics of both
the structure and the weights of a DV are given by the definitions of the quality

61
characteristics, as specified in the Quality Model. A DV node may be based on a Design Model element, as indicated by the Based on dependency relationship. As indicated by the self-reference on the Class Node, one node may be decomposed into children nodes. Directed arcs express dependency with respect to quality characteristic by relating each parent node to its immediate children nodes, thus forming a tree structure. Each arc on a DV is assigned an EI (Estimated Impact), which is a normalized value of degree of dependence of a parent node, on the immediate child node. Thus, there is a quantified dependency relationship from each parent node, to its immediate children. The values on the nodes and the arcs are referred to as parameter estimates. We distinguish between prior and inferred parameter estimates. The former ones are, in the form of empirical input, provided on leaf nodes and all arcs, while the latter ones are deduced using the propagation model for DVs, which is exemplified by Equation 1 below.

5.3.2 Dependency Views

As explained above, the DVs express the interplay between the system architectural design and the quality characteristics. DVs are deduced from the Design Models and the Quality Models of the target system. A DV node may be based on an element from the Design Models. One DV is, according to the process of the PREDIQT method, developed for each quality characteristic defined by the Quality Models of the target system. The DVs model the dependencies of the system quality characteristic that the DV is dedicated, with respect to architectural design of the target system, in the form of multiple weighted and directed trees. An arc represents a dependency relationship through a dashed arrow which is directed from a dependent node to a dependee node. An arc is annotated by a value of degree of dependence of parent node on the child node, or impact of the child node on the parent node. A node is represented by a rectangle annotated by a name and a value. The value expresses degree of fulfillment of a quality characteristic (that the DV is dedicated to) by what is represented by the node. The values and the dependencies modeled through the DVs are based on the definitions provided by the Quality Model. Thus, the structure of a DV expresses the dependencies of a parent node on its immediate children nodes, with
5.3 The approach to modeling in PREDIQT

Data protection
QCF=0.94

Encryption
QCF=1.00

Authentication
QCF=0.95

Authorization
QCF=0.90

Other
QCF=0.90

EI=0.30  EI=0.25  EI=0.30  EI=0.15

Figure 5.4: Excerpt of an example DV with fictitious values

respect to the quality characteristic addressed by the DV. Similarly, the parameter values on the arcs (representing the degree of dependency) and the nodes (representing the degree of quality characteristic fulfillment), are estimated with respect to the quality characteristic addressed by the DV. A DV comprises two notions of parameters:

1. EI: Estimated degree of Impact between two nodes, and
2. QCF: degree of Quality Characteristic Fulfillment.

Each arc pointing from the node being influenced is annotated by a quantitative value of EI, and each node is annotated by a quantitative value of QCF.

Figure 5.4 shows an excerpt of an example DV with fictitious values. In the case of the Encryption node of Figure 5.4, the QCF value expresses the goodness of encryption with respect to the quality characteristic in question, for example, security. A quality characteristic is defined by the underlying system specific Quality Models, which may for example be based on ISO 9126 product quality standard [61]. A QCF value on a DV expresses to what degree the node (representing system part, concern or similar) is realized so that it, within its own domain, fulfills the quality characteristic. The QCF value is based on the formal definition of the quality characteristic (for the system under analysis), provided by the Quality Models. The EI value on an arc expresses the degree of impact of a child node (which the arc is directed to) on the parent node, or to what degree the parent node depends on the child node. The EI of an arc captures the impact of the child node on its parent node, with respect to the sub-characteristics (defined in the Quality Models) of the quality characteristic under consideration. Once a total contribution of the sub-characteristics is obtained on each arc pointing to children nodes with a common parent, the EI values are normalized so that they sum up to one. “Initial” or “prior” estimation of a DV involves providing QCF values to all leaf nodes, and EI values to all arcs. The terms “prior estimate” and “initial estimate” are used interchangeably, and regard the estimates directly assigned to the EIs and leaf node QCFs, that is, the parameters based on the empirical input and assigned before the non-leaf node QCFs may be inferred.

Input to the DV parameters may come in different forms (for example, from domain expert judgments, experience factories, measurements, monitoring, logs, etc.), during the different phases of the PREDIQT method. Once the initial parameter values are assigned, the QCF value of each non-leaf node is recursively (starting from leaf nodes and moving upwards in the tree) propagated by multiplying the QCF and EI value for each immediate child and summing up these products for all the immediate children. This is referred to as the general DV propagation algorithm. For example, with respect
Achievements: the overall picture

to Data protection node on Figure 5.4 (denoting: DP: Data protection, E: Encryption, AT: Authentication, AAT: Authorization, and O:Other):

\[
QCF_{(DP)} = QCF_{(E)} \cdot EI_{(DP\rightarrow E)} + QCF_{(AT)} \cdot EI_{(DP\rightarrow AT)} + QCF_{(AAT)} \cdot EI_{(DP\rightarrow AAT)} + QCF_{(O)} \cdot EI_{(DP\rightarrow O)} \tag{1}
\]

The DV-based approach constrains the QCF of each node to range between 0 and 1, representing minimal and maximal characteristic fulfillment (within the domain of what is represented by the node), respectively. This constraint is ensured through the formal definition of the quality characteristic rating (provided in the Quality Models). The sum of Els, each between 0 (no impact) and 1 (maximum impact), assigned to the arcs pointing to the immediate children must be 1 (for model completeness purpose). Moreover, all nodes having a common parent have to be orthogonal (independent). The dependent nodes are placed at different levels when structuring the tree, thus ensuring that the needed relations are shown at the same time as the tree structure is preserved. The overall concerns are covered by the nodes denoted Other, which are included in each set of nodes having a common parent, thus making the DV complete.

The general DV propagation algorithm, exemplified by Equation 1, is legitimate since each quality characteristic DV is complete, the Els are normalized and the nodes having a common parent are orthogonal due to the structure. A DV is complete if each node which is decomposed, has children nodes which are independent and which together fully represent the relevant impacts on the parent node, with respect to the quality characteristic that the DV is dedicated to.

The rationale for the orthogonality is that the resulting DV structure is tree-formed and easy for the domain experts to relate to. This significantly simplifies the parametrization and limits the number of estimates required, since the number of interactions between the nodes is minimized. Although the orthogonality requirement puts additional demands on the DV structuring, it has shown to represent a significant advantage during the estimation.

5.3.3 Tool support for the DV-based modeling and analysis

The purpose of a tool support is to facilitate development and use of the DVs in an industrial setting, where visualization aids comprehensibility and communication. The tool should ideally hide the unnecessary details, and automate as much of the DV development and DV-based analysis as possible. This is expected to save time during the analysis, which contributes to the part of the overall objective regarding possibility of conducting a PREDIQT-based analysis within acceptable resources. Moreover, tool support aims to facilitate interactive model development in groups, trigger discussions and aid documentation of DVs that are commonly understood and agreed upon by the domain experts. Tool-supported analysis should facilitate providing correct and quick predictions to the analyst.

We have developed a prototype tool based on MS Excel [4] for visualizing the DVs as well as automatically performing change propagation and sensitivity analysis on the DVs. In this section, we discuss the functionality of the tool, and we explain what has already been done and what remains to be done. The analyst is the primary user of the tool. The domain experts provide the input necessary for developing the DVs. They view the DVs that the analyst is presenting, typically in a group meeting setting. When
5.3 The approach to modeling in PREDIQT

A DV structure is developed and the prior estimates are provided, the QCFs on the non-leaf nodes are automatically propagated. The domain experts then validate the propagation results and the structure of the DV as a whole. The propagated values are also, by the analyst, validated based on measurement-based input which is statistically analyzed on additional Excel sheets. A realistic size of a DV structure is shown by Figure 31 in Paper 1. A graph from sensitivity analysis is shown by Figure 9 in Paper 1.

Figure 5.5 shows a UML use case diagram that specifies the main functionality of the DV tool. These use cases are:

(UC1) **Create a sub-tree** This involves generating a new sub-tree with one single root node and (if any) children nodes, which again may be root nodes of other sub-trees. Creation includes assigning a name to each node, and relating the
dependent nodes with arcs, according to the rules prescribed by the approach to modeling in PREDIQT.

(UC2) **Delete a sub-tree** This involves deleting an existing sub-tree with one single root node and its underlying sub-trees.

(UC3) **Update a sub-tree** Update may involve changing value of an EI or changing value of a QCF (in case of a leaf node). Structural changes (such as changing the name of a node, adding a new node or deleting a node) are covered by (UC1) and (UC2).

(UC4) **Propagate a parameter value** This involves inferring QCF to the non-leaf nodes, according to the DV propagation model exemplified by Equation 1.

(UC5) **Check validity of a parameter value** This involves validating that each QCF is between 0 and 1 and that each EI is between 0 and 1.

(UC6) **Check completeness of a sub-tree** This involves validating that sum of EI values on the arcs connecting a node to its immediate children, is equal to 1.

(UC7) **Run sensitivity analysis** This involves graphically displaying the relative impact of a set of parameters, on the root node QCF value.

(UC8) **Perform statistical analysis** This involves deducing statistical operator values from a set of data based on empirical input. Such values are used to deduce the prior estimates, or to validate the (inferred values on the) models.

The use cases (UC1)-(UC4) and (UC6)-(UC8) have all to varying degree been implemented in the prototype tool. (UC1) and (UC2) however involve manual effort and the tool does not optimize space utilization in order to offer a user-friendly visualization. (UC3) and (UC4) are implemented in the tool. (UC5) is not included in the tool. (UC6) is partially included by visualizing the sums of EI values on each sub-tree. (UC7) and (UC8) are included in the tool, although manual effort by the analyst is necessary for customizing the functionality, that is, relating the underlying data to the functions and graphs for the analysis.

### 5.4 The approach to uncertainty handling in PREDIQT

The values assigned to the DV parameters originate from two types of empirical input: domain expert judgments and measurement-based data acquisition (measurements, logs, monitoring, historical data, or similar). However fine grained, the DVs contain a certain degree of uncertainty due to lack and inaccuracy of empirical input. Therefore, the representation of the uncertain input should be intuitive, as exact as possible and provide a well defined (complete and sound) inferring mechanism. The approach to uncertainty handling in PREDIQT addresses representation, propagation and analysis of uncertainties in DVs. The purpose of the approach is to facilitate model fitting, identify the kinds of architectural design changes which can be handled by the prediction models, and indicate the value of added information. Moreover, based on experiences from PREDIQT-based analyses obtained through industrial case studies on real-life systems, we also provide guidelines for use of the approach in practice. The guidelines
5.4 The approach to uncertainty handling in PREDIQT

address the ways of obtaining the empirical estimates as well as the means and measures for reducing uncertainty of the estimates. The approach is presented in Paper 2.

As explained in Section 4.4, uncertainty is generally categorized into two different types: aleatory (due to inherent randomness of the system or variability of the usage profile) and epistemic (due to lack of knowledge or information about the system) [70]. The aleatory uncertainty is irreducible even by additional measurements. Aleatory uncertainty is typically represented by continuous probability distributions and forecasting is based on stochastic models. Epistemic uncertainty, on the other hand, is reducible, non-stochastic and of discrete nature. The epistemic uncertainty is therefore best suited for possibilistic uncertainty representations. Prediction models are characterized by rather discrete, sudden, non-stochastic and less frequent changes. In majority of the system quality prediction models, aleatory uncertainty is negligible in terms of magnitude and impact, while the epistemic one is crucial. It is therefore the epistemic uncertainty we focus on when dealing with the parameters on the DVs.

Our approach to uncertainty handling in PREDIQT is based on intervals with associated confidence level, and allows representation, propagation, and analysis of all the parameters associated with uncertainty. Input acquisition is in this context concerned with how the DV estimates and their uncertainty measures are obtained in practice. An overview of the practical means and measures for 1) acquiring the input and 2) achieving a specified minimum level of uncertainty, is clearly a prerequisite for applicability of the uncertainty handling approach. Therefore, we also provide guidelines for practical use of our solution, covering both the issues of estimate acquisition and uncertainty handling. The guidelines are driven by the experiences from the empirical evaluations of the PREDIQT method. In the following, we present and exemplify the parts of the approach regarding uncertainty representation and uncertainty propagation. For uncertainty analysis and guidelines for estimate acquisition, we refer to Paper 2.

5.4.1 Uncertainty representation

All prior estimates are expressed in terms of intervals within which the correct parameter values should lie. The width of the interval is proportional to the uncertainty of the domain experts or deduced from the standard deviation of the measurement-based input represented with probabilistic notions. In the latter case, the standard deviation indicates the accuracy of the measurements associated with each initially estimated parameter. Thus, the interval width may vary between the individual parameters.

In addition to the quantifiable uncertainty associated with each initially estimated parameter, there may exist sources of uncertainty which are general for the context or the system itself, but to a lesser degree expressive or measurable. Examples include the presence of the aleatory uncertainty, the competence of the domain experts, data quality, statistical significance, etc. Such factors contribute to the overall uncertainty, but are (due to their weak expressiveness or generality) not explicitly taken into account within the initially estimated EIs and the leaf node QCFs. Another reason for not accounting them within the intervals is because they are unavailable or may be biased at the individual parameter level. The domain experts may for example be subjective with respect to the above exemplified factors, or the tools for data acquisition may be incapable of providing the values regarding data quality, statistical significance, etc.
Achievements: the overall picture

Figure 5.6: Excerpt of a DV with intervals and confidence level

Therefore, the context related uncertainty should, from an unpartial perspective (for example, by a monitoring system or a panel, and based on a pre-defined rating), be expressed generally for all prior estimates.

Hence, we introduce the “confidence level” as a measure of the expected probability that the correct value lies within the interval assigned to a prior estimate. The confidence level is consistent and expresses the overall, uniform, context or system relevant certainty, in terms of a percentage. The confidence level regards the prior estimates only. The confidence level dictates the width of the intervals of the prior estimates, that is, the certainty with which the exact value is within the interval assigned to a prior estimate. For example, a confidence level of 100% guarantees that the exact values lie within the intervals assigned to the prior estimates. Obviously, a requirement for increased confidence level will result in wider intervals of the prior estimates.

Let QCFs and EIs be represented by intervals of type $x$:

$$x = [x; \overline{x}] = \{X \in [0; 1] : x \leq X \leq \overline{x}\} \quad (2)$$

where $x$ is the minimum estimated parameter value above which the exact value should (the term “should” is intentionally used in order to account for the confidence level of the prior estimates which is below 100%) lie, while $\overline{x}$ is the maximum parameter value below which the exact value should lie. Both $x$ and $\overline{x}$ are represented by real numbers. The interval $x$ of a prior estimate is assigned with the confidence level specified. Due to model completeness, EIs on the arcs pointing to the nodes with a common parent must satisfy:
\[ (\Sigma_{i=1}^I x_i) \leq 1 \land (\Sigma_{i=1}^I \bar{x}_i) \geq 1 \]  

(3)

where \( i \) denotes index of an arc, \( I \) denotes the total number of the arcs with outspring from a common parent, and \( x_i \) denotes the interval estimate for the EI on arc \( i \). That is, there must exist at least one subset of scalars from within each one of the intervals (representing EIs on the arcs to nodes with a common parent), whose sum is equal to 1.

The representation of the estimates and their uncertainty is exemplified through an excerpt of a DV (with fictitious values) shown in Figure 5.6. The DV models architectural aspects with respect to target system security. The top node represents the target system security. Prior estimates are assigned to all EIs and leaf node QCFs. All prior estimates are expressed in terms of intervals with a confidence level of 90% and equations 2 and 3 are satisfied.

The measurement-based input may come in terms of statistical operators. For instructions regarding how to transform the input represented by statistical operators, to interval, see Section 4 of Paper 2.

### 5.4.2 Uncertainty propagation

The initial estimates are provided in the form of intervals with respect to a confidence level, as specified above. The propagation of the initially estimated intervals on the non-leaf node QCFs is given by the existing DV propagation model (exemplified by Equation 1), the interval arithmetics \([68, 75]\), and the algorithms for non-linear optimization \([88, 103]\). The result of the propagation is in the form of intervals of QCF values on the non-leaf nodes.

The confidence level itself is not propagated but only used in the context of the assignment of the initial estimates. Therefore, the confidence level is only associated with the initial estimates and not the inferred ones (non-leaf node QCFs). The confidence level does however affect the width of the inferred parameters through the width of the initial estimates. That is, since a requirement for a higher confidence level implies wider intervals of the initial estimates, the propagation will, as specified below, result in wider intervals on the non-leaf node parameters.

The only two interval arithmetic operations needed for propagation in a DV are addition and multiplication. In case of two intervals denoted by \( x \) and \( y \) (of the form given by Equation 2), addition and multiplication are defined as:

\[ x \circ y = [x \circ y; \bar{x} \circ \bar{y}] \]  

(4)

Where \( \circ \) denotes the operation symbol.

The optimization is necessary for obtaining the boundary values (the maximum and the minimum) of the interval of a parent node in the cases when several combinations (within the propagated intervals) give a sum of the EIs (on the arcs pointing to the immediate children) equal to 1. The scalar points (from within the intervals involved), which provide the boundary values, are identified by the non-linear optimization algorithms and then inferred to the parent node QCF in the form of an interval, according to the general DV propagation algorithm.

For a set of EI intervals whose total sum of the upper interval values is more than 1, there may be infinitely many combinations (the number of the combinations depends on the number of decimal digits, which the scalars from the intervals are represented with)
Achievements: the overall picture

of scalar points from within all the intervals, which together sum up to 1. Regardless of how many EIs (or nodes) there are, finding the min and the max values of the interval resulting from the propagation (sum of products of QCF and EI values associated with respectively the immediate children nodes and the arcs pointing to them) is a feasible optimization problem [68,103]. Since the number of unknowns is equal to the number of equations involved, the only condition for the feasibility of the algorithm is the one expressed by Equation 3.

Let $qcf_i, qcf_i \in [0;1]$ denote the interval limits of the QCFs on the immediate children and let $e_i, e_i \in [0;1]$ denote the EIs on their respective interconnecting arcs. We propose the utility functions for the inferred min and max for the intervals of the parent node QCFs, which are given by respectively:

$$QCF \overset{\text{def}}{=} \min \left\{ \sum_{i=1}^{I} qcf_i \cdot e_i \mid \forall i \in I : e_i \leq e_i \leq e_i \land \sum_{i=1}^{I} e_i = 1 \right\}$$  \hspace{1cm} (5)

$$QCF \overset{\text{def}}{=} \max \left\{ \sum_{i=1}^{I} qcf_i \cdot e_i \mid \forall i \in I : e_i \leq e_i \leq e_i \land \sum_{i=1}^{I} e_i = 1 \right\}$$  \hspace{1cm} (6)

$I$ and $i$ denote the same notions as in Equation 3. The inference starts from the lowest internal nodes, and proceeds recursively upwards the tree. Figure 5.6 shows the result of applying this utility function to obtain the QCF values on Target interfaces and Target systems nodes. Due to the requirement specified by equation 3 on EIs of a sub-tree, the propagation is feasible. Equation 2 is satisfied by the QCF values on the Target interfaces and Target systems nodes due to the condition specified by the right hand side of the utility function.

The uncertainty handling approach has been tried out in practice by applying it on a DV structure originating from the first case study (presented in Paper 1). The results are reported in Section 4 of Paper 2. For the optimization necessary for evaluating the boundary values (that is, maximum and minimum points) of the utility function during the propagation of QCF values, Matlab [5] has been used. Paper 2 contains detailed instructions for how to use Matlab in this context. We have also evaluated the feasibility of other uncertainty handling approaches in the PREDIQT context. Based on our literature review, we have in Paper 2 argued why the alternative uncertainty approaches do not perform sufficiently on one or more success criteria identified for uncertainty handling in PREDIQT. Thus, the approach to uncertainty handling has been evaluated through applying it on an example, as well as through a literature review. For the details regarding the approach to uncertainty handling in PREDIQT, its evaluation, and the above mentioned guidelines for input acquisition, we refer to Paper 2.

### 5.5 The approach to traceability handling in PREDIQT

Important preconditions for model-based prediction are correctness and proper usage of the prediction models. The process of the PREDIQT method guides the development and use of the prediction models, but the correctness of the prediction models and the way they are applied are also highly dependent on the creative effort of the analyst and his/her helpers. In order to provide additional help and guidance to the analyst, we have developed a traceability approach for recording and retrieving:

- The relations between traceable elements of the prediction models;
5.5 The approach to traceability handling in PREDIQT

- Structure or contents of traceable elements documented through 1) external documents, or 2) a specification of the rationale and assumptions made.

The approach includes a traceability scheme for PREDIQT, and an implementation of it in the form of a prototype tool which can be used to define, document, search for and represent the trace-links needed. The traceability scheme defines the tracing needs for PREDIQT. The traceability scheme is based on the traceability needs deduced from detailed guidelines for application of prediction models. The guidelines are presented in Appendix 2 of Paper 5 and instruct the analyst on how to correctly apply the prediction models, supported by a traceability solution. Thus, based on the needs for the final phase of PREDIQT, we accommodate recording of the trace information during the model development. Hence, the guidelines serve for both deducing the traceability scheme, but also for instructing the analyst for how to correctly use the prediction models, with the traceability support.

The traceability solution is applied on prediction models from an earlier PREDIQT-based analysis of a real-life system. We have also evaluated the feasibility of other traceability approaches in the PREDIQT context. Based on our literature review and the results of the evaluation by Galvao and Goknil [52], we have in Section 7 of Paper 5 argued why the alternative traceability approaches do not perform sufficiently on one or more success criteria identified for traceability handling in PREDIQT. Thus, the approach to traceability handling has been evaluated through applying it on an example, as well as through a literature review. Below we present the traceability scheme and the prototype traceability tool. For the details regarding the approach to traceability handling in PREDIQT, its evaluation, and the above mentioned guidelines for application of the prediction models, we refer to Paper 5.

5.5.1 Traceability scheme

The above mentioned guidelines suggested the need for the following kinds of trace-links:

- Links between the Design Model elements;
- Links from the Design Model elements to DV elements;
- Links from DV elements to Quality Model elements (that is, traces to the relevant quality indicators and rationale for the prior estimates);
- Links to external information sources (documents, measurement, domain experts) used during the development of DV structure and estimation of the parameters;
- Links to rationale and assumptions for: Design Model elements, the semantics of the DV elements, as well as structure and prior parameter estimates of the DVs.

Another need identified based on the guidelines is that the traceability approach should have facilities for both searching with model types and model elements as input parameters, as well as for reporting linked elements and the link properties from the search results.

The traceability scheme consists of a meta-model for trace-link information and a feature diagram for capabilities of the solution. The types of the trace-links needed to
be documented and retrievable as well as the types of the traceable elements needed to be documented and retrievable, are represented through a meta-model shown by Figure 5.7. The *Element* abstract class represents a generalization of a traceable element. The *Element* abstract class is specialized into the five kinds of traceable elements: *Design Model Element*, *DV Element*, *Quality Model Element*, *External Information Source*, and *Rationale and Assumptions*. Similarly, the *Trace Link* abstract class represents a generalization of a trace-link and may be assigned a rationale for the trace-link. The *Trace Link* abstract class is specialized into the six kinds of trace-links.

Pairs of certain kinds of traceable elements form binary relations in the form of unidirectional trace-links. Such relations are represented by the UML-specific notations called association classes (a class connected by a dotted line to a link which connects

*Figure 5.7: A meta model for trace-link information, expressed as a UML class diagram*
5.5 The approach to traceability handling in PREDIQT

Figure 5.8: Main capabilities of the traceability approach, expressed as a feature diagram

two classes). For example, trace-links of type Design Model Element to Design Model Element may be formed from a Design Model Element to a Dependency View Element. The direction of the link is annotated by the origin (the traceable element that the trace-link goes from) and the target (the traceable element that the trace-link goes to). Since only distinct pairs (single instances) of the traceable elements (of the kinds involved in the respective trace-links defined on Figure 5.7) can be involved in the associated specific kinds of trace-links, uniqueness (property of UML association classes) is present in the defined trace-links. Due to the binary relations (arity of value 2) in the defined trace-links between the traceable elements, only two elements can be involved in any trace-link. Furthermore, multiplicity of all the traceable elements involved in the trace-links defined is of type “many”, since an element can participate in multiple associations (given they are defined by the meta-model and unique).

The main capabilities needed are represented through a feature diagram [124] shown by Figure 5.8. Storage of trace-links may be internal or external, relative to the prediction models. A traceable element may be of type prediction model element (see Figure 5.3) or non-model element. Reporting and searching functionality has to be supported. Trace-link info has to include link direction, link meta-data (for example, date, creator, strength) and cardinality (note that all links are binary, but a single element can be origin or target for more than one trace-link). Typing at the origin and the target ends of a trace-link as well as documenting rationale for trace-link, are optional.

5.5.2 Tool support for traceability handling

Tool support for traceability handling in PREDIQT aims to facilitate efficient documentation and retrieval of trace-link information. Tool-supported trace-link handling facilitates correctness and reuse of the trace-link info. The tool also hides the unnecessary details and facilitates the process by guiding the analyst in conducting the traceability handling activities correctly. Moreover, the tool automates parts of the traceability handling process. This is expected to save time during the analysis, which should contribute to the part of the overall objective regarding possibility of conducting a PREDIQT-based analysis within acceptable resources.

We have developed a prototype tool in the form of a database application with user interfaces, on the top of MS Access [2]. The purpose of the prototype traceability tool
Figure 5.9: Use cases for the traceability tool

is to evaluate feasibility of supporting the traceability scheme for PREDIQT, by a tool. The tool support also serves as a proof-of-concept for the traceability scheme. The tool also aims to facilitate the traceability related process activities (by guiding the user in conducting the correct sequence of actions and providing the expected contents). The prototype tool is a part of the traceability solution in PREDIQT. The prototype tool includes a relational database for organizing the trace information, queries for retrieval of the trace info, a menu for managing work flow, forms for populating trace-link information, and facilities for reporting trace-links. In this section, we discuss the functionality of the tool, and we explain what has already been done and what remains to be done. The analyst is the primary user of the tool. The domain experts provide the input necessary for recording the traceable elements and the trace-links.

Figure 5.9 shows a UML use case diagram that specifies the main functionality of
5.5 The approach to traceability handling in PREDIQT

The use cases (UC1)-(UC6) have all to varying degree been implemented in the prototype tool. (UC1)-(UC4) are implemented in the form of user interfaces and underlying database support. Although considerable manual effort is required for inserting the items and their properties, a complete functionality is provided in the tool. (UC5) is fully supported, by using the MS Access queries. (UC6) is fully included, by using the MS Access reporting functionality which facilitates designing customized reports. (UC7) is not implemented in the prototype tool, since we consider the challenge to be on the side of the external models which should export the data on the traceability-relevant items, based on a standardized format which is compatible with the traceability tool. (UC8) is not implemented in the prototype tool, since MS Access has limited facilities for such a visualization on the reports. Additional programming of the tool is needed for such an extension.
Ordering of use cases (UC1)-(UC6) represents a typical sequence of actions of an end-user (the analyst), in the context of defining, documenting and use of the trace-links. The basic definition of the types of the traceable elements and the trace-links are provided first. Then, concrete traceable elements are documented, before defining specific instances of the trace-links and their associated specific origin and target elements, involved in the binary trace-link relations. Finally, reports can be obtained, based on search results which are based on parameters such as for example model types, model elements, or trace-link types. An extract of screen shot of a trace-link report (obtained from the prototype tool) is shown by Figure 5.10. The report presents the trace-links associated with the prediction models related to “Split signature verification component into two redundant components, with load balancing”, corresponding to Change 1 in Paper 1. For further details on the prototype traceability tool, we refer to Paper 5.

We have chosen to store the trace-link information externally, as the traceability tool needs to handle model and non-model artifacts from a variety of external sources: diagrams stored in different tools, external documents and domain expert judgments. The drawback of the external storage is the manual trace recording, while the advantage is: 1) the flexibility of what can be recorded and to what degree of detail, as well as 2) avoiding of pollution in the source models. Advantages of the external storage are argued for by Anquetil et al. [17].

Figure 5.10: A screen shot of an extract of a trace-link report from the prototype traceability tool
Chapter 6

Overview of research papers

The main results of the work presented in this thesis are documented in the papers in Part II. In the following we give an overview of these research papers, by describing the topics of each paper and indicating how much of the results are credited the author of this thesis.

6.1 Paper 1: A Feasibility Study in Model-based Prediction of Impact of Changes on System Quality

Authors: Aida Omerovic, Anette Andresen, Håvard Grindheim, Per Myrseth, Atle Refsdal, Ketil Stølen and Jon Ølnes.


Contribution: Aida Omerovic was the main author, responsible for about 90% of the work.

Main topics: The paper reports on results and experiences from using the PREDIQT method in a comprehensive industrial case study targeting a so-called “Validation Authority” (VA) system [89]. VA is used for managing evaluation of electronic identifiers (certificates and signatures) worldwide. The paper first provides a general overview of the PREDIQT method in its initial form. Thereafter, an evaluation of the method in terms of a feasibility study and a thought experiment (which complemented the feasibility study), is presented. The process undergone in applying PREDIQT on VA, the experiences obtained, the prediction models developed and the evaluation results, are reported. Security is one of the three major quality attributes that the VA quality is decomposed into. The evaluation focuses on security and its trade-offs with the overall quality attributes identified as relevant and defined with respect to the VA system, namely scalability and availability. The results indicate that PREDIQT is feasible in practice, in the sense that it can be carried out on a realistic industrial case and
with limited effort and resources. Results of the evaluation of PREDIQT based on the thought experiment suggest that useful predictions can be obtained.

6.2 Paper 2: A Practical Approach to Uncertainty Handling and Estimate Acquisition in Model-based Prediction of System Quality

Authors: Aida Omerovic and Ketil Stølen.

Publication status: Accepted for publication in the International Journal on Advances in Systems and Measurements (2011, vol 4, no 1&2). This paper is a revised and expanded version of the paper originally published in the proceedings of the Second International Conference on Advances in System Simulation (SIMUL’2010) [95]. The latter paper received a best paper award at SIMUL’2010.

Contribution: Aida Omerovic was the main author, responsible for about 90% of the work.

Main topics: The paper proposes an approach to handling of uncertainties in PREDIQT. Uncertainty is associated with quantitative empirical input which is provided in terms of prior estimates to the Dependency Views (DV). The approach is based on intervals with associated confidence level, and allows representation, propagation and analysis of the uncertainty. Based on a set of criteria, we argue analytically and empirically, that our uncertainty handling approach is comprehensible, sound and practically useful in the PREDIQT context. We also evaluate the feasibility of the alternative uncertainty handling approaches, relative to the pre-defined success criteria. Moreover, based on experiences gained from PREDIQT-based analyses which were conducted in relation to two industrial case studies on real-life systems, we provide guidelines for use of the uncertainty handling approach in practice. The guidelines address the ways of obtaining the empirical estimates as well as the means and measures for reducing uncertainty of the estimates.

6.3 Paper 3: Uncertainty Handling in Weighted Dependency Trees – A Systematic Literature Review

Authors: Aida Omerovic, Amela Karahasanovic and Ketil Stølen.


Contribution: Aida Omerovic was the main author, responsible for about 90% of the work.
Main topics: The paper identifies and evaluates the existing and the potentially useful approaches for handling the uncertainty in the context of the Weighted Dependency Trees (WDTs). Any acyclic graph can be transformed to a WDT, and uncertainty handling is essential for correctness of models and credibility of a WDT-based analysis. Due to the main source of uncertainty being lack or inaccuracy of input (rather than variability of the target system and its usage), we only focus on uncertainty handling in systems whose behavior is deterministic. The existing and the potentially useful approaches are identified through a systematic literature review. The approaches are then outlined and evaluated at a high-level, before a restricted set undergoes a more detailed evaluation based on a set of pre-defined evaluation criteria. In addition to the literature review and the evaluation, this paper contributes with a proposal of a set of evaluation criteria regarding selection of uncertainty handling approaches in a WDT context. The results indicate that precision, expressiveness, predictive accuracy, scalability on real-life systems and comprehensibility are among the properties which differentiate the approaches. Based on our results, we recommend the directions for further development of the uncertainty handling approaches. The paper is intended to serve as a roadmap for examining the uncertainty handling approaches, or as a resource for identifying the adequate one.

6.4 Paper 4: Evaluation of Experiences from Applying the PREDIQT Method in an Industrial Case Study

Authors: Aida Omerovic, Bjørnar Solhaug and Ketil Stølen.

Publication status: Technical report A17562, SINTEF ICT, 2011. The report presented in this thesis is the full version of the paper published in proceedings of the 5th IEEE International Conference on Secure Software Integration and Reliability Improvement (SSIRI’2011) [93].

Contribution: Aida Omerovic was the main author, responsible for about 90% of the work.

Main topics: The paper reports on the experiences from applying the PREDIQT method in the second industrial case study – on an ICT system from a different domain and with different system architecture and quality characteristics, compared with the first case study (reported in Paper 1). The objective of the case study reported has been to perform an additional and more structured evaluation of the PREDIQT method and assess its performance with respect to a set of success criteria. The paper includes an overview of the PREDIQT method, details regarding the research method and success criteria deduced, an outline of setup and data collection during the PREDIQT-based analysis, a presentation of the process undergone during the PREDIQT-based analysis, a summary of the artifacts developed during the process, and the results of assessment in terms of: 1) evaluation of predictions 2) written feedback obtained during a postmortem review, 3) verbal feedback collected during the analysis, and 4) observations made during the analysis. The results are evaluated with respect
to the pre-defined success criteria. The evaluation suggests feasibility and usefulness of the PREDIQT-based analysis. Moreover, the study has provided useful insights into the weaknesses of the method and suggested directions for future research and improvements.

6.5 Paper 5: Traceability Handling in Model-based Prediction of System Quality

**Authors:** Aida Omerovic and Ketil Stølen.

**Publication status:** Technical report A19348, SINTEF ICT, 2011. The report presented in this thesis is the full version of the paper accepted for publication in proceedings of the 3rd International Conference on Advances in System Simulation (SIMUL’2011) [96].

**Contribution:** Aida Omerovic was the main author, responsible for about 90% of the work.

**Main topics:** The paper proposes an approach to traceability handling in PREDIQT. Traceability aims to facilitate correctness and proper usage of the prediction models. The traceability approach addresses documenting and retrieving the rationale and assumptions made during the model development, as well as the dependencies between the elements of the prediction models. The approach is defined by a traceability scheme, which is basically a feature diagram specifying capabilities of the solution and a meta-model for the trace-link information. The traceability scheme is deduced from the traceability needs for PREDIQT, which are identified from detailed guidelines for application of prediction models. The paper includes an overview of the PREDIQT method, guidelines for application of prediction models and an outline of the success criteria. Then, the traceability scheme for PREDIQT and an implementation of it in the form of a prototype tool which can be used to define, document, search for and represent the trace-links needed, are presented. Subsequently, we report on the results and experiences from applying the solution on a subset of prediction models from an earlier PREDIQT-based analysis of a real-life system. Based on the pre-defined set of success criteria, we argue that our traceability approach is useful and practically scalable in the PREDIQT context. We also evaluate the feasibility of the alternative traceability handling approaches, relative to the success criteria.
Chapter 7

Discussion

In this chapter we discuss and evaluate the contributions of this thesis. We have used cases studies, examples originating from the case studies, thought experiments and literature reviews, for the evaluation of the success criteria. In order to verify the feasibility of our artifacts and their influence on system quality and system adaptability, further empirical evaluations are necessary. In Section 7.1 we evaluate the contributions against the success criteria formulated in Chapter 2, and in Section 7.2 we discuss how our contributions relate to state of the art.

7.1 Fulfillment of the success criteria

In Chapter 1 we formulated our objective: to develop a method for model-based prediction of the effects of architectural design changes on the quality characteristics of a system. Since the development of a complete quality prediction method is too extensive for a PhD thesis, we identified four sub-tasks as a first step towards a model-based quality prediction method, namely to develop: (1) a process for conducting an analysis based on the method; (2) a modeling approach with support for modeling dependencies between system architecture and system quality characteristics; (3) an approach for handling uncertainties related to the empirical input in the context of the prediction models; and (4) an approach for handling traceability of model and non-model artifacts developed throughout the process of the method. The overall objective was refined into a set of success criteria that each contribution should fulfill. In the following sub-sections we discuss to what extent the success criteria have been fulfilled for each contribution.

7.1.1 The process of the PREDIQT method

Success criterion 1 The process of the PREDIQT method provides sufficient and comprehensible guidance for an analyst to conduct a PREDIQT-based analysis.

A description of steps and actions to be conducted by the analyst during a PREDIQT-based analysis, is provided and thoroughly exemplified through a real-life case presented in Paper 1 (which reports on the first industrial case study). The resulting prediction models are presented. They may in future PREDIQT-based analyses serve as examples of expected outcomes of the process. A second industrial case study is presented in Paper 4. Moreover, Paper 2 and Paper 5 contain overviews of the PREDIQT method with
process description and model structure. Paper 2 provides guidelines for the analyst regarding how to use the uncertainty handling approach and how to acquire the empirical input and its uncertainty estimates. Paper 5 provides guidelines for the analyst regarding how to correctly use the prediction models with support of the traceability approach, during the “Application of the prediction models” phase of PREDIQT. Overall, Papers 1, 2, 4 and 5 complement each other and provide instructions for the analyst regarding modeling, interaction with the overall stakeholders, and correct usage of the prediction models. Thus, sufficiency of the process guidelines is indicated by the fact that a consistent process specification has been used in conducting the two case studies with promising results.

Although it is assumed that an analyst is trained in conducting a PREDIQT-based analysis (as stated in Chapter 2), comprehensibility of the process from the point of view of an analyst needs further evaluation. In the two industrial case studies (reported in Paper 1 and Paper 4), the analyst was the author of these reports and designer of the PREDIQT method. Comprehensibility of the process to an analyst in general has therefore not been sufficiently evaluated. Hence, further empirical evaluation is needed to verify that the existing process specification is sufficient and comprehensible for analysts in general (assuming that they have received some basic training).

Success criterion 2 The process of the PREDIQT method results in predictions that are sufficiently correct.

All the architectural design changes, that were suggested by the representatives of the customer organizations in the two case studies, have been possible to apply and predict the effects of. Both the extent of the changes applied and the accuracy of the prediction obtained have varied. Although the evaluation of the model-based predictions has been based on thought experiments only, the predictions from the two case studies have (by the evaluators affiliated to the customer organizations) been judged to be useful. The evaluators did not express any threshold for acceptable deviation of the model-based predictions relative to the thought-experiment based ones. No significant pattern in deviation of model-based predictions relative to the thought-experiment based ones has been observed. Variation in the average deviations between predictions and thought experiment was larger in the first case study, than in the second one. In the second case study, there was a tendency towards pessimism of the predictions, while more variation was present in the average deviation associated with validation. The changes applied during validation (of the prediction models in the second case study) were smaller than the ones applied during model application. In both case studies, however, the number of changes applied is so small, that a discussion of patterns in deviation is not really meaningful. Still, correctness and usefulness of the predictions is subject to many factors, such as: kind and extent of the changes applied on the models, the purpose and complexity of the target system, the acceptable deviation in accuracy of the predictions, and quality of the prediction models. Although the two case-studies have been comprehensive, independent and conducted on two different domains, further empirical evaluations of correctness of the prediction models are needed. The approaches to uncertainty and traceability in PREDIQT aim to make the correctness and certainty of the models more explicit, as well as to facilitate correct development and usage of the prediction models.

Success criterion 3 The process of the PREDIQT method is applicable in an industrial context and within acceptable effort.
The evaluation results of the two case studies indicate that a PREDIQT-based analysis is feasible within the allocated resources, while still providing useful predictions in a real-life industrial setting. Realistic architectural design changes have been applied on the prediction models obtained and all the phases of the process of the PREDIQT method have been possible to conduct within the resources allocates in the case of both case-studies.

### 7.1.2 The approach to modeling in PREDIQT

**Success criterion 4** The approach to modeling in PREDIQT provides a sufficiently expressive and self-contained set of models and instructions needed for predicting effects of architectural design changes with respect to system quality.

The instructions for development of prediction models and presentations of the DVs (syntax, semantics and the development steps) are included in Papers 1, 2, 4 and 5. Paper 1 contains examples of real prediction models and the steps undergone in relation to their development. A prototype of tool support for DV-based modeling and analysis (presented in Section 5.3.3) is developed and used in both case studies. The approach to modeling in PREDIQT and its tool support have been used in both case studies, with promising results. The target systems have been modeled at the detail level needed and within the characterized scope. The evaluations have indicated usefulness of the predictions obtained. Since predictions depend on correctness of the prediction models, fulfillment of the criterion is indicated by the evaluations of the predictions obtained from the two case studies. The fact that all proposed and realistic changes could be applied in both case studies, suggests sufficient expressiveness of the prediction models.

**Success criterion 5** The approach to modeling in PREDIQT offers the sufficient support for analyzing effects of architectural design changes on system quality.

PREDIQT provides a self-contained set of the prediction models and guidance for how to apply and analyze the effects of architectural design changes on them. Moreover, PREDIQT provides guidance for obtaining correct DV structure, and an inference model which is applicable provided the conditions for completeness of the sub-trees and orthogonality of the immediate children (with respect to the quality characteristic modeled by the DV). The presentations of the approach to modeling in PREDIQT and the DV examples presented in the Papers 1, 2, 4 and 5, all explain and exemplify how to apply changes on models and analyze their effects on system quality. Sufficiency of the modeling support is evaluated through the thought experiment-based evaluation of the predictions in relation to the case studies. The predictions obtained (propagation paths and quality characteristic fulfillment estimates) from both case studies have been judged useful. Although additional empirical evaluations are needed to verify reliability of the predictions, the current results indicate that the success criterion is fulfilled.

**Success criterion 6** The approach to modeling in PREDIQT is suitable for handling empirical input.

Empirical input in the form of expert judgments and measurements is the basis for development of prediction models. Expert judgments come in the form of: 1) estimates of EI and QCF values, 2) statements for rationale and assumptions for model structure,
and 3) statements for rationale and assumptions for the estimates provided. Measurements may come in the form of statistical operators or be deduced to such operators from raw data. Both kinds of empirical input (expert judgments and measurements) have been possible to represent by the prediction models. The domain experts have been actively involved in development and use of prediction models during both case studies, which indicates comprehensibility of the models. The values of the statistical operators, which are derived from the measurements, have been transformed to the DV parameter values (based on the semantics of the DV operators on the nodes or the arcs they are related to). Moreover, since the statistical operators can be transformed to the DV parameter values, the DV propagation model is applicable for both kinds of empirical input. Feasibility of both case studies, active involvement of the domain experts and indicated usefulness of the predictions, suggest fulfillment of this success criterion.

**Success criterion 7** *The complexity of the prediction models is limited and scalable in a practical setting.*

In the case of the DVs, the number of estimates needed in a DV is linearly proportional to the number of nodes in the model. The number of QCFs is equal to the number of nodes, and number of EIs is equal to the number of nodes, minus one. Since prior estimates are assigned to all leaf nodes and all EIs, the number of prior estimates is equal to the number of arcs (that is, number of nodes minus one) plus the number of the leaf nodes. The number of nodes in a single DV may potentially become excessive but this has not been the case in the two case studies, since needs for decomposition of the DV structure have been limited.

The complexity of the Design Models and the Quality Models is subject to what is characterized as target and scope of the analysis. Design Models and Quality Models may consist of many diagrams and are modular. Moreover, the models existing prior to the analysis can serve the purpose of the Design Models and the Quality Models, while the DVs are developed during the analysis.

The experiences from the two case studies suggest that the complexity of the prediction models is scalable in a realistic setting. In both case studies, all prediction models have been developed during the analyses, the changes applied on them have been realistic and the predictions obtained have been judged to be useful.

**Success criterion 8** *The approach to modeling in PREDIQT is comprehensible by the domain experts and the analyst.*

The fact that the domain experts, during both case studies, actively participated and continuously made progress according to the schedule of the analysis, managed to perform thought experiments and apply the models, indicates comprehensibility of the approach to modeling in PREDIQT. During both case studies, the Design Models and the Quality Models were proposed by the analyst, based on documentation and many interactions with the domain experts. The process was iterative and ended when a common understanding of the models was reached and their approval by the domain experts obtained. During the first case study, the analyst played an active role in development of the DV structure, while the domain experts estimated the parameters and validated the models. Development of the DVs was, during the second case study, entirely performed by the domain experts and only facilitated by the analyst. The
involvement indicates that the models were in general comprehensible for the participants. Furthermore, the postmortem reviews of both case studies suggest that the models served well as an aid in establishing a common understanding of the target. Still, comprehensibility of the models may vary among the participants and between the models depending on the knowledge of the system and the modeling notation. Thanks to the structured guidelines for application of prediction models, the simple propagation model, and the DV tool support, it has been relatively easy for the analyst to manage analysis of the effects of architectural changes on quality, based on the DVs. Thus, the preliminary evaluation suggests that this success criterion is fulfilled.

However, as argued in relation to Criterion 1, we have not performed detailed studies of comprehensibility for analyst or domain experts in general. Particularly, since the analyst was actively involved in development of the method and report writing, comprehensibility for an analyst in general needs further evaluation.

### 7.1.3 The approach to uncertainty handling in PREDIQT

**Success criterion 9** The approach to uncertainty handling in PREDIQT is applicable for representing the uncertainty estimates related to both expert-judgment-based and measurement-based input.

The domain expert judgments come directly in terms of intervals with a confidence level. The interval-based approach extends the DV parameters with the notions of interval widths and confidence level. Both interval width and confidence level are based on fairly intuitive and simple definitions. Hence, the approach should be relatively easy for the domain experts to use and understand, regardless of the degree of their formal background.

The measurement-based input may come in terms of statistical notions. Section 4 of Paper 2 specifies how to map the measurement based input expressed in terms of mean and standard deviation, to the intervals.

Moreover, the interval width can be selected at the individual prior estimate level, thus allowing adjustment of precision of the uncertainty representation. The number of the decimal digits used in estimation and propagation is unlimited. We did not identify need for representing uncertainty variation within the intervals. Thus, this success criterion is fulfilled.

**Success criterion 10** The approach to uncertainty handling in PREDIQT offers sufficient and comprehensible guidelines to the analyst for obtaining the input along with its uncertainty, based on domain expert judgments and measurements.

Section 6 of paper 2 provides the guidelines necessary for an analyst to acquire input (based on domain expert judgments and measurements) along with its uncertainty. The guidelines address the means and measures for acquiring the input and expressing it in terms of interval-based estimates with a confidence level. The guidelines are based on the lessons learned from the two case studies. Section 4 of Paper 2 specifies how to map the measurement based input expressed in terms of mean and standard deviation, to the intervals. The evaluation which has been based on a subset of models from the first case study, indicated feasibility of the approach to uncertainty handling.

As for Success Criterion 1, comprehensibility of the approach from the point of view of an analyst needs further evaluation. Since the above mentioned guidelines have been
designed and written by the analyst, who also evaluated the approach to uncertainty handling, comprehensibility to an analyst in general has not been sufficiently evaluated.

**Success criterion 11** The number of uncertainty estimates needed is linearly proportional to the number of nodes in the model.

This success criterion is fulfilled, as the number of QCFs, each expressed in terms of an interval, is equal to the number of nodes, and number of EIs, each expressed in terms of an interval, is equal to the number of nodes, minus one. The prior estimates are associated with one shared confidence level. Since prior estimates are assigned to all leaf nodes and all EIs, the number of prior estimates is equal to the number of arcs (that is, number of nodes minus one) plus the number of the leaf nodes.

**Success criterion 12** The approach to uncertainty handling in PREDIQT supports propagation of the uncertainty estimates.

The propagation model is presented and exemplified in Section 5.4.2 and Section 3 of paper 2. A consequence of the inequality and equality constraints is that all the inferred values will lie within the interval $[0;1]$. In addition, the normalized quality characteristic metric is defined so that all possible values always must lie within this interval. Moreover, the propagation algorithm calculates both the upper and the lower extreme values. As a result, the inferred prediction is an interval within which the exact (factual) value should lie. Two aspects are hindering from guaranteeing that the factual value lies within the inferred interval:

- The confidence level with which the prior estimates are provided;
- The aleatory uncertainty, which unless accounted for in the confidence level, is not quantified within the intervals.

It should be pointed out that the interval propagation based on the boundary values suffers from the so-called overestimation effect, also known as dependency problem. A possible consequence of this is that the inferred interval is wider than necessary, resulting in a possibly more pessimistic uncertainty propagation. The dependency problem is due to the memoryless nature of interval arithmetic in cases when a parameter occurs multiple times in an arithmetic expression, since each occurrence of an interval variable in an expression is treated independently. Since multiple occurrence of interval parameters cannot always be avoided, the dependency problem may cause crucial overestimation of the actual range of an evaluated function. A way to approach this issue is to use interval splitting [80], where the input parameter intervals are subdivided and the arithmetics are preformed on the subintervals. The final results are then obtained by computing the minimum of all lower bounds and the maximum of all upper bounds of the intermediate results. Skelboe [112] has shown that the results obtained from the interval splitting converge to the actual range if the width of the subintervals approaches zero. Our solution does not use interval splitting, as it would significantly increase complexity of the entire approach and therefore degrade comprehensibility. Thus, the claim that the inferred prediction is an interval within which the exact (factual) value should lie, still applies.

**Success criterion 13** The approach to uncertainty handling in PREDIQT is comprehensible for the domain experts.
7.1 Fulfillment of the success criteria

The interval-based approach represents uncertainty of the DV parameters through interval widths and confidence level. As argued above, both interval width and confidence level are based on fairly intuitive and simple definitions. The input is, by the analyst, acquired by precise questions specified in the guidelines for input acquisition (Section 6 of Paper 2). Hence, the approach should be relatively easy for the domain experts to understand, regardless of the degree of their formal background. Due to their simplicity, the guidelines, the uncertainty representation and the propagation should also be comprehensible for the analyst. The simplicity of the representation also makes it less prone to unstable over-fitting, as well as bias or inaccuracy of the estimations. We have however not performed detailed evaluation of comprehensibility of the approach to uncertainty handling in PREDIQT from the point of view of the domain experts.

Success criterion 14 *The approach to uncertainty handling in PREDIQT supports statistical analysis of the uncertainty estimates.*

The analysis of the central tendency measures of the interval-based estimates relies on the existing fully defined interval arithmetics and interval statistics [50]. Both can, in their existing well-established form, be directly applied in our context. For arithmetic mean, geometric mean, harmonic mean, weighted mean, median, standard deviation and variance, see Ferson et al. [50]. In addition, Ferson et al. [50] provide guidance regarding identification of outliers, trade-off between sample size and precision, handling of measurement uncertainty, handling of dependencies among the sources of uncertainty (correlation and covariance) and accounting for incertitude. Uncertainty analysis support for interval-based representation of the estimates is also addressed in Section 3 of Paper 2. Thus, this success criterion is fulfilled.

7.1.4 The approach to traceability handling in PREDIQT

Success criterion 15 *The approach to traceability handling in PREDIQT offers the support necessary for an analyst to handle the needed kinds of the trace-links.*

The kinds of the trace-links needed are defined by the traceability scheme (presented in Section 4 of paper 5) and deduced from the guidelines for application of the prediction models (presented in Appendix 2 of Paper 5). The types of trace-links and their properties, are specified in dedicated tables in the database of the prototype traceability tool. This allows constraining the types of the trace-links to only the ones defined, or extending their number or definitions, if needed. The trace-links in the prototype traceability tool are binary and unidirectional, as required by the traceability scheme. Macros and constraints can be added in the tool, to implement any additional logic regarding trace-links, traceable elements, or their respective type definitions and relations. The data properties (for example, date, hyperlink or creator) required by the user interface, allow full traceability of the data registered in the database of the prototype tool. Handling of the trace-links implies defining, recording and using them. The usage implies searching and reporting of the trace-links recorded. These facilities are defined by the traceability scheme and supported by the prototype traceability tool. The application of the solution on example prediction models (originating from the first case study) is reported in Section 5 of Paper 5 and indicates that the trace-links
Discussion

needed and functionalities for their handling, are supported by the solution proposed. Thus, the preliminary evaluation suggests that this success criterion is fulfilled.

As for Success Criterion 1, comprehensibility of the approach to traceability handling in PREDIQT from the point of view of an analyst needs further evaluation. Since the above mentioned traceability scheme and guidelines have been designed and written by the analyst, who also evaluated the traceability handling approach, comprehensibility of the approach to an analyst in general has not been sufficiently evaluated.

**Success criterion 16** The approach to traceability handling in PREDIQT offers searching facilities with model types and model elements as input parameters.

Searching based on: user input, selectable values from a list of pre-defined parameters, or comparison of one or more database fields, are relatively simple and fully supported by queries in MS Access [2] that the prototype tool is implemented in. Thus, this success criterion is fulfilled.

**Success criterion 17** The approach to traceability handling in PREDIQT offers reporting facilities which show the linked elements and the trace-link properties.

Customized reports can be produced with results of any query and show any information registered in the database. The report, an extract of which is presented in Section 5 of Paper 5, is based on a query of all documented trace-links and the related elements. Thus, this success criterion is fulfilled.

**Success criterion 18** The approach to traceability handling in PREDIQT is flexible with respect to granularity of trace information.

The text-based fields for documenting the concrete instances of the traceable elements and the trace-links, allow level of detail selectable by the user. Only a subset of fields is mandatory for providing the necessary trace-link data. The optional fields in the tables can be used for providing additional information such as for example rationale, comments, links to external information sources, attachments, strength or dependency. There are no restrictions as to what can be considered as a traceable element, as long at it belongs to one of the element types defined by traceability scheme. Similarly, there are no restrictions as to what can be considered as a trace-link, as long at it belongs to one of the trace-link types defined by traceability scheme. The amount of information provided regarding the naming and the meta-data, are selectable by the end-user. Thus, this success criterion is fulfilled.

**Success criterion 19** The approach to traceability handling in PREDIQT supports real-life applications of PREDIQT.

Traceability scheme is specified by a meta-model and a feature diagram (both presented in Section 5.5.1), which is an established practice as presented by Winkler and Pilgrim [124]. The kinds of traceable elements and the kinds of trace-links are specified by the UML-based [108] meta-model. The capabilities of the solution are specified by the feature diagram. The design decisions made aim to balance coverage of the needs identified, with simplicity of the solution, in order to provide an approach which is useful and practically applicable within acceptable resources and as a part of a PREDIQT-based analysis.
7.2 How our contributions relate to state of the art

The approach to traceability handling in PREDIQT is supported by a tool for recording and retrieving trace-links. The tool hides the unnecessary details and facilitates the process by guiding the analyst in conducting the traceability handling activities correctly. The tool facilitates automating the activities performed and reuse of the trace information, in order to limit the time spent on traceability handling.

The approach has been applied on example prediction models (originating from the first case study). The experiences from and the results of applying the approach are reported in Section 5 of Paper 5. Given the realism of the prediction models involved in the example, the size and complexity of the target system they address, the representativeness of the change applied on them, as well as the time spent on documenting the trace-links and using them, the results indicate the applicability of our solution on real-life applications of PREDIQT. Thus, the preliminary evaluation suggests that this success criterion is fulfilled.

7.2 How our contributions relate to state of the art

In this section, we discuss how our work relates to state of the art. The discussion below addresses the method in general by comparing the most relevant parts of state of the art with PREDIQT. The reason why this is not already conducted in Chapter 4 is that this discussion refers to aspects of PREDIQT which are presented after state of the art, that is, in Chapter 5. For further comparison of our artifacts to state of the art, the reader is referred to the related work sections of the papers presented in Part II of the thesis.

We have striven to use the established terminology and adopt most of the existing tools and notations, in order to facilitate comprehensibility of the process and allow reuse of the existing models and measurements. The objective is to facilitate conducting a useful PREDIQT-based analysis within acceptable resources. Therefore, the Design Models may be based on any existing notation as long as the target characterized is modeled to a sufficient degree. Similarly, Quality Models may use the existing ISO 9126 [61] standard or goal/question/metric [20,21] approach when deducing and specifying the quality notions.

The Architecture Tradeoff Analysis Method (ATAM) [31,66,67] is most similar to PREDIQT, in terms of its objective. The purpose of ATAM is to assess the consequences of architectural decision alternatives in light of quality attribute requirements. The purpose of an ATAM is not to provide precise analyses, but to discover risks created by architectural decisions. ATAM asks guided questions in order to reveal tradeoffs, risks and trends. Instead of asking questions, PREDIQT develops explicit and more detailed models which document the system and its quality characteristics. Moreover, the models developed by PREDIQT project the architectural changes by making their propagation and effects on quality explicit. Compared to ATAM, PREDIQT allows a more fine grained analysis of several quality characteristics and their trade-offs simultaneously. Although shorter in time than the process of PREDIQT, ATAM process also takes place in the form of facilitated interaction between multiple stakeholders, leading to the identification of risks, sensitivities, and tradeoffs. In contrast to PREDIQT, ATAM does not base its input on measurements but mainly on interactions with the stakeholders. Within one of its steps, ATAM generates “quality attribute utility tree”. Its purpose is to document tradeoffs, rather than modeling dependencies in terms of
quantitative values and analyzing effects of changes in terms of parameter propagations (which is the case in PREDIQT). The utility trees focus on quality deduced from business goals and leaf nodes represent the relevant scenarios that impact the quality attributes. Instead of model-based analysis of effects of changes, the analyses of ATAM are based on brainstorming. As one of its outputs, ATAM identifies the architectural approaches impacted by the scenarios. As such, the results of ATAM are closely related to the architecture that the analysis is based on, so that the risks and trade-off points only apply to the pre-assumed architecture. In PREDIQT, however, changes in architecture are supported. Such changes are projected to the DVs and analyzed in terms of propagation paths as well as updated quality attribute values. Similarly to PREDIQT, ATAM also defines scope of analysis and quality attributes (through the above mentioned quality attribute utility tree) as frames of the analysis. Thus, ATAM is primarily a risk identification method and not a predictor of quality achievement.

In many existing comparable methods which aim to improve system quality, the development process is the independent variable through which system quality is controlled. Rather than through changes in architectural design (which is the case in PREDIQT), system quality is controlled through processes improvement measures. In Statistical Process Control (SPC) [122], product quality is monitored in order to adjust the processes to pre-defined quality goals. Capability Maturity Model Integration (CMMI) [34] focuses on model-based assessment and improvement of system development processes. Also in the case of CMMI, system quality is controlled through process improvement. Moreover, the CMMI is considerably extensive and detailed, thus requiring more effort compared to the process of PREDIQT. Similarly, Six Sigma [100] aims at quality improvement, but the process is far more extensive than the one of PREDIQT. Prediction is not an explicit goal of Six Sigma, but quality analysis is one of the phases. Similarly to CMMI and SPC, goal/question/metric (GQM) [20, 21] approach focuses on using metrics, (primarily) in the context of process improvement (and improvement of its resulting software systems). GQM addresses specification of the quality goals and their metrics. It provides guidance and notation for decomposition of the quality notions. The GQM is also suitable for deriving target system specific Quality Models. GQM can be used for specifying the quality notions and their metrics. Guidance for both notation and decomposition of the quality notions is provided, but no support for prediction is offered.

Pattern-based quantitative security assessment is presented by Yautsiukhin et al. [126], where threat coverage metrics are associated with security patterns and aggregated to an overall security indicator. This is a solely security oriented approach for quantitative security assessment (and not prediction) with limited traceability to the underlying design and quality notions. Similarly, Heyman et al. [55] introduce an approach for decomposing security objectives into security patterns, which again are decomposed into security measures, which serve as indicators. The resulting dependency graphs are only developed for the security quality attribute and with limited traceability to the system design and its overall quality characteristics.

Fault trees [58], event trees [59] and attack trees [83] focus on modeling attacks or unwanted incidents in the context of a security analysis. PREDIQT however aims to address several quality characteristics and their trade-offs. Therefore, the approach to modeling in PREDIQT needs a notation for modeling dependencies with respect to quality characteristics beyond security. Moreover, the existing tree-based notations do not offer sufficient inference mechanisms for change propagation.
7.2 How our contributions relate to state of the art

Paper 3 presents an evaluation of the existing approaches which could serve as alternatives to the DVs (with support for uncertainty handling). The results suggest that Weighted Dependency Trees (WDTs) with interval-based representation of estimates and WDTs with fuzzy logic based representation of estimates, are most applicable in the PREDIQT context. It is argued that Bayesian Belief Networks (BBNs) [54,87] demand considerable number of estimates to be provided into a realistic model, and therefore suffer from low scalability.

Moreover, BBNs are demanding to parametrize and interpret the parameters of. This issue has been addressed by Omerovic and Stølen [94] where an analytical method for transforming the DVs to BBNs is presented. It also shows that DVs are, although easier to relate to in practice, compatible with BBNs. It is possible to extend this method so that our interval-based approach is transformed to a BBN before a further BBN-based analysis can proceed. Such an extension would introduce several states on the BBN nodes, and assign probabilities to them. In that manner, the extension resembles to the Dempster-Shafer structures. Such an extension is exemplified in Paper 3. The uncertainty representation is contained in both the granularity (the number of states on the nodes) and their respective probabilities. Then, Bayes rule is applied for the propagation. The prior estimates consist of the probabilities associated with each state on the leaf nodes, as well as the dependencies estimated through the values in the conditional probability table (CPT). Due to the number of the estimates needed (which is exponential with respect to the number of nodes), the BBN-based approach does not scale in practice. Furthermore, since the uncertainty representation of several states for each node requires extensive CPTs and a demanding propagation, the BBN-based approach suffers from lower comprehensibility. The precision depends on the number of states introduced for each node, which means that e.g. new evidence of higher precision than the one defined through the existing states, may be difficult to represent. Hence, combining evidence of different uncertainty (for example, expert judgments and measurements) on a node may be problematic.

Our approach to uncertainty handling (presented in Paper 2) is a special case of the Dempster-Shafer [50] approach. In our approach, the intervals of the prior estimates have a general confidence level, and the structure of the DV allows for a linear propagation. Implementing the Dempster-Shafer theory in the context of DVs would involve solving two issues: 1) sorting the uncertainties in the empirical input into a priori independent items of evidence, and 2) carrying out Dempster’s rule computationally. The former leads to a structure involving input elements that bear on different but related concerns. This structure can be used to make computations based on Dempster’s rule, feasible. However, the additional expressiveness that the Dempster-Shafer structures offer is not needed in our context, since the certainty is highly unlikely to vary across the fractions of the intervals. In fact, such a mechanism would, due to its complex representation on subsets of the state space, in the PREDIQT context only compromise the comprehensibility of the uncertainty representation.

The interval-based approach is also a special case of the fuzzy approach [115,121,129], where only the crisp intervals are used as the membership function. The additional expressiveness that the overall types of the membership functions offer is in fact not needed in the PREDIQT context, since the increased complexity of the estimate representation would not contribute to the accuracy of the parameter values, but rather introduce misinterpretations and incorrectnesses in the input provision. The interpretation of the membership distributions and their correspondence to the practical
settings in the PREDIQT context would be demanding. The measurement-based input may however benefit from the different membership distributions. Since the membership functions can be selected and combined, the preferred one can be used in each case, thus giving a higher precision. A possible way of using the Fuzzy logic based approach is illustrated in Paper 3.
Chapter 8

Conclusion

This chapter concludes Part I of the thesis by summarizing the results and pointing out directions for future work.

8.1 Developed artifacts

The increasing criticality of the ICT systems and the steady growth of their mutual collaboration impose new challenges with respect to managing the adaptations of system architecture to new needs while preserving the required system quality. Quality compliance of collaborating systems is a condition for their interoperability. We assert that predictable system quality is a necessary condition for trustworthiness of a system. When adapting a system to changes, there is a need for decision making support which facilitates understanding of the effects of architectural design changes on system quality. There are often trade-offs between quality attributes and one goal may be possible to achieve through several different architectural designs. Analysis of system architecture and system quality based on abstract models is an alternative to for example testing or simulation. By using model-based analysis, needs for testing environments or systems replica, as well as downtime resulting from premature change deployment, may be reduced. Models can be used to efficiently communicate and analyze structure and properties of complex systems, before decisions regarding the changes on the system are made.

Such a support would have a preventive effect in the sense that models could be used to efficiently make informed decisions about possible system changes and their effects, rather than implementing them and then observing their effects on quality on a running system. Moreover, a predictive approach could serve in selecting among the alternative architectural design changes in cases when several options lead to a desired goal regarding a system adaption and the resulting quality level. A model-based approach to system quality prediction could potentially also serve in prioritizing test cases or planning maintenance of a system.

This thesis contributes to the domain of model-based system quality prediction. Our overall objective has been to develop a method for model-based prediction of the effects of architectural design changes on the quality characteristics of a system, that is:

1. useful in the sense that the predictions are sufficiently correct;

2. applicable in an industrial context and within acceptable effort;
3. comprehensible for the stakeholders involved; and
4. cost-effective.

To this end, we have put forward a method, called PREDIQT. The PREDIQT method aims at establishing the right balance between practical applicability of the process and usefulness of the predictions. The PREDIQT method makes use of models that capture: 1) the architectural design of the target system, 2) the quality notions of the target system, and 3) the interplay between architectural design and quality characteristics of the target system. The predictions result in propagation paths and the modified values of the parameters which express the quality characteristic fulfillment at the different abstraction levels. We are not aware of other approaches that combine these elements in such a way. However, the issues of metrics estimation, system quality and the various notations for modeling system architecture, have received much attention in the literature. Similarly, there exist numerous frameworks for software system evaluation, methodologies for process improvement, as well as approaches for handling traceability and uncertainty. We have argued that some of the existing modeling notations as well as quality standards, are applicable for fulfilling specific partial needs of PREDIQT. We have also argued the need for novel solutions. To this end, we have put forward four new artifacts, each of which is necessary within such a method:

1. The process of the PREDIQT method.
2. The approach to modeling in PREDIQT.
3. The approach to uncertainty handling in PREDIQT.
4. The approach to traceability handling in PREDIQT.

### 8.2 Evaluation of artifacts

As part of the evaluation of the process of the PREDIQT method and the approach to modeling in PREDIQT, we have conducted two industrial case studies (presented in Paper 1 and Paper 4) performed on real-life systems. Thought experiments have in both cases been used to evaluate predictions. Alternatives to the approach to modeling in PREDIQT with support for uncertainty handling have been evaluated a systematic literature review (presented in Paper 3).

Both case studies have given strong indications of feasibility of PREDIQT, reported similar benefits and undergone the same stages of the PREDIQT process. The experiences and results obtained indicate that the PREDIQT method can be carried out within acceptable resources, on a real-life system, and result in useful predictions. Furthermore, the observations indicate that the method, particularly its process, facilitates understanding of the system architecture and its quality characteristics, and contributes to structured knowledge management through system modeling.

Moreover, the involvement of the participants of the analyses and the feedback received during the postmortem reviews of the case studies, indicate that the analyses have been comprehensible for the stakeholders involved. Given the evaluation of the predictions, the resources necessary for conducting the two analyses and the feedback received, the preliminary evaluation indicates cost-effectiveness of the method.
The approach to uncertainty handling (presented in Paper 2) has been evaluated based on an example and a literature review. The preliminary evaluation indicates usefulness and practical applicability of the approach, in the PREDIQT context. The literature review has indicated need for an uncertainty handling approach which is adapted to PREDIQT. The literature review has furthermore argued that, given the pre-defined success criteria for uncertainty handling in PREDIQT, our approach is more applicable than the existing ones.

The approach to traceability handling (presented in Paper 5) has also been evaluated based on an example and a literature review. The preliminary evaluation indicates usefulness and practical applicability of the approach, in the PREDIQT context. The literature review has indicated need for a traceability handling approach which is adapted to PREDIQT. The literature review has furthermore argued that, given the pre-defined success criteria for traceability handling in PREDIQT, our approach is more applicable than the existing ones.

8.3 Directions for future work

There are a number of possible directions for future work. Most importantly, more empirical evaluations of PREDIQT in various domains are needed for assessing its performance in practical settings, with respect to the overall objective. Moreover, further evaluations can provide insights into the weaknesses of the method and suggest directions for future research and improvements. Our approaches to uncertainty and traceability handling should also be included in future industrial case studies, in order to evaluate their practical applicability and efficiency in a real-life setting.

PREDIQT has only architectural design as the independent variable – the Quality Model itself is, once developed, assumed to remain unchanged. This is of course a simplification, since system quality prediction is subject to more factors than architectural design. Usage profile, quality definitions and system development process, are examples of the factors whose variation PREDIQT does not address. These factors would be interesting to integrate into PREDIQT.

It would also be interesting to include into PREDIQT a cost-benefit analysis of architectural design changes. Such an extension could potentially ease selection among a set of architectural designs, according to their cost-effectiveness. A possible way of approaching this is by including the notions of cost and utility into the approach to modeling in PREDIQT, and explicitly representing such parameters on the DVs. There are already approaches and tools which include utility functions into BBNs, and it may be possible to gain some insights from those approaches.

Development of an experience factory, that is, a repository of the non-confidential and generalizable experiences and models from earlier analyses, is another direction for future work. An experience factory from similar domains and contexts would allow reuse of parts of the prediction models and potentially increase model quality as well as reduce the resources needed for a PREDIQT-based analysis.

Moreover, PREDIQT does not take into account the timing aspects. That is, the estimates to the prediction models do not express the variation of the values over time but only the values related to the current state of the system. Similarly, the values of the quality predictions are not related to the time elapsed after the change has been implemented. The variation over time would be interesting to include into the
predictions, particularly if the notions of cost and value are included in the method. Including the timing aspects and a value-based approach would facilitate balancing the capabilities (particularly complexity and flexibility) of the system architecture with the system lifetime.

A PREDIQT manual for the analyst with: detailed guidelines, description of procedures, specification of prerequisites for sub-phases, specification of (required and optional) analysis participants and their roles, templates for input acquisition, templates for reporting, reusable prediction model patterns, and examples from earlier analyses, would make the PREDIQT process more streamlined and ease training of a new analyst. It could be interesting to evaluate if tool-supported process guidance would be beneficial. A tool-supported PREDIQT process guidance would make the steps performed more schematic, parts of the process could be automated, and quality assurance of the process would be more manageable.

With respect to uncertainty handling, further work should address analysis of the prediction accuracy, that is the deviation between the predicted and the actual quality characteristic values. The notions of magnitude of average deviation [91], balanced relative error (BRE) [18] and hit rate (i.e., the percentage of the correct values lying within the predicted intervals) can be used as measures of prediction accuracy. For an accurate prediction model, the hit rate should be consistent with the confidence level. The BRE allows analysis of bias and precision of the predictions. Thus, systematic and random variance of the prediction accuracy can be distinguished in a meta analysis of our uncertainty handling approach. Moreover, identifying and categorizing the variables that impact the uncertainty of the estimates, is important for improving uncertainty management.

With respect to traceability handling, performing an analysis of factors such as cost, risk, and benefit and following the paradigm of value-based software-engineering, would be relevant in order to stress the traceability handling effort on the important trace-links. As argued by Winkler and Pilgrim [124], if the value-based paradigm is applied to traceability, cost, benefit, and risk will have to be determined separately for each trace according to if, when, and to what level of detail it will be needed later. This leads to more important artifacts having higher-quality traceability. There is a trade-off between the semantically accurate techniques and cost-efficient but less detailed approaches. Finding an optimal compromise is still a research challenge. Our solution proposes a feasible approach, while finding the optimal one is subject to further research. Future work should also include standard interfaces and procedures for updating the traceable elements from the prediction models into our prototype traceability tool. As model application phase of PREDIQT dictates which trace-link information is needed and how it should be used, the current PREDIQT guidelines focus on the application of the prediction models. However, since the group of recorders and the group of users of traces may be distinct, structured guidelines for recording the traces during the model development should also be developed as a part of the future work.

With respect to approach to modeling in PREDIQT, improved tool support is needed. The current tool does not offer sufficient support for efficient visualization of the DV structure. Particularly, structural changes require manual work related to visualization, as well as adjustment of the embedded propagation function and validation of estimates on the sub-tree affected. Moreover, the DV tool does not include uncertainty handling, because extensions of the tool are needed for enabling uncertainty
propagation as specified by our uncertainty handling approach. We have demonstrated feasibility of the uncertainty propagation by calculating the inferred boundary values using optimization functions in MATLAB [5], but the functionality should be implemented in a DV tool. Additionally, the DV tool should support the DV-related part of traceability handling. Whether the two tools should be tightly or loosely integrated, is subject to further research.

Moreover, the needs for and possible benefits of integration of the experience factory, the above mentioned process support tool, the DV tool and the traceability tool, would be worthwhile examining. The integration needs are subject to the degree to which the parts of the process that the tools support, are related. Technical properties of the tools that support the different parts of the PREDIQT process, dictate the possibilities of integrating them. In case external tools are used for development of Design Models and Quality Models, interfaces for export of traceable element information into the traceability tool are relevant for reducing the manual effort in relation to trace recording.
Bibliography


Part II

Research Papers
Chapter 9

Paper 1: A Feasibility Study in Model-based Prediction of Impact of Changes on System Quality
A Feasibility Study in Model Based Prediction of Impact of Changes on System Quality

AUTHOR(S)
Aida Omerovic, Anette Andresen, Håvard Grindheim, Per Myrseth, Atle Refsdal, Ketil Stølen and Jon Ølnes

CLIENT(S)
Research Council of Norway

ABSTRACT
We propose a method, called PREDIQT, for model based prediction of impact of architecture design changes on system quality attributes. PREDIQT supports simultaneous analysis of several quality attributes and their trade-offs. This paper argues for the feasibility of the PREDIQT method based on a comprehensive industrial case study targeting a system for managing validation of electronic certificates and signatures worldwide. We first give an overview of the PREDIQT method, and then present an evaluation of the method in terms of a feasibility study and a thought experiment. The evaluation focuses on security and its trade-offs with the overall quality attributes of the target system.

Key words: Quality, Prediction, System architecture, Modeling, Simulation, Impact analysis, Design
# TABLE OF CONTENTS

1 Introduction ................................................................. 1

2 Overview of the PREDIQT Method ................................................. 3

  2.1 Target modeling ................................................................. 3

  2.1.1 Characterize the target and the objectives ............................... 4
  2.1.2 Create quality models ...................................................... 4
  2.1.3 Map design models ......................................................... 4
  2.1.4 Create dependency views ............................................... 4
  2.2 Verification of prediction models ........................................... 6

  2.2.1 Evaluation of models ...................................................... 6
  2.2.2 Fitting of prediction models ............................................. 7
  2.2.3 Approval of the final prediction models .............................. 7

  2.3 Application of prediction models .......................................... 7

  2.3.1 Specify a change ............................................................ 7
  2.3.2 Apply the change on prediction models .............................. 8
  2.3.3 Quality prediction .......................................................... 8

3 Application of PREDIQT in the Industrial Case ............................ 9

  3.1 Target modeling ................................................................. 9

  3.1.1 Characterize the target and the objectives ............................... 9
  3.1.2 Create quality models ...................................................... 9
  3.1.3 Map design models ......................................................... 11
  3.1.4 Create dependency views ............................................... 11

  3.2 Verification of prediction models ........................................... 13

  3.2.1 Evaluation of models ...................................................... 13
  3.2.2 Fitting of the prediction models ........................................ 14
  3.2.3 Approval of the final prediction models .............................. 14

  3.3 Application of prediction models .......................................... 15

  3.3.1 Specify a change ............................................................ 15
  3.3.2 Apply the change on prediction models .............................. 15
  3.3.3 Quality prediction .......................................................... 17

4 Evaluation Based on a Thought Experiment .................................. 18

  4.1 Set-up ................................................................. 18
  4.2 Research method ........................................................... 19
  4.3 Results ................................................................. 19

5 Discussion .............................................................................. 19

  5.1 Feasibility study .............................................................. 19
  5.2 Thought experiment ........................................................ 20

6 Threats to Validity and Reliability ............................................. 21

7 Related Work .......................................................................... 21

8 Conclusions and Future Work .................................................. 23

Appendix 1: Organization of the PREDIQT trial on VA .......................... 26
Appendix 2: A selection of the design models of the VA ......................... 28
Appendix 3: The quality models of the VA ........................................ 45
Appendix 4: The conceptual model of the VA ....................................... 52
Appendix 5: Structure of the DVs .................................................. 54
Appendix 6: Schema for documentation of results of the change simulation 57
Appendix 7: The measurement plan for VA ........................................ 68
Appendix 8: Schema for documentation of results of the thought experiment 72
Appendix 9: The process diagram for evaluation of the PREDIQT method 74
A Feasibility Study in Model Based Prediction of Impact of Changes on System Quality

Aida Omerovic\(^1,2\), Anette Andresen\(^3\), Håvard Grindheim\(^3\), Per Myrseth\(^3\), Atle Refsdal\(^1\), Ketil Stølen\(^{1,2}\) and Jon Ølnes\(^3\)

1: SINTEF ICT, Oslo, Norway; 2: Department of Informatics, University of Oslo, Norway; 3: DNV, Høvik, Norway

Abstract. We propose a method, called PREDIQT, for model based prediction of impact of architecture design changes on system quality. PREDIQT supports simultaneous analysis of several quality attributes and their trade-offs. This paper argues for the feasibility of the PREDIQT method based on a comprehensive industrial case study targeting a system for managing validation of electronic certificates and signatures worldwide. We first give an overview of the PREDIQT method, and then present an evaluation of the method in terms of a feasibility study and a thought experiment. The evaluation focuses on security and its trade-offs with the overall quality attributes of the target system.

Key words: Quality prediction, system architecture design, change impact analysis, modeling, simulation

1 Introduction

Quality is defined as “The totality of features and characteristics of a software product that bear on its ability to satisfy stated and implied needs” [3]. Examples of quality attributes include availability, scalability, security and reliability. When adapting a system to new usage patterns or technologies, it is necessary to foresee what such adaptions of architectural design imply in terms of system quality. Predictability with respect to non-functional requirements is one of the necessary conditions for the trustworthiness of a system. Examination of quality outcomes through implementation of the different architecture design alternatives is often unfeasible. A model based approach is then the only alternative.

We have developed the PREDIQT method with the aim to facilitate prediction of impacts of architecture design changes on system quality. The PREDIQT method produces and applies a multi-layer model structure, called prediction models, which represent system design, system quality and the interrelationship between the two. Our overall hypothesis is that the PREDIQT method

can, within practical settings and with needed accuracy, be used to predict the
effect of specified architecture design changes on the quality of a system. Qual-
ity is decomposed through a set of quality attributes, relevant for the target
system. PREDIQT supports simultaneous analysis of all the identified quality
attributes and their trade-offs. The PREDIQT approach of merging quality con-
cepts and design models into multiple, quality attribute oriented “Dependency
Views” (DVs), and thereafter simulating impacts of architecture design changes
on quality, is novel.

The prediction models represent system relevant quality concepts (through
“Quality Models”) and architecture design (through “Design Models”). In addi-
tion, the prediction models comprise “Dependency Views” (DVs), which repre-
sent the dependencies between architecture design and quality in form of multiple
weighted and directed trees. The aim has been to develop a method that provides
the necessary instructions and means for creating and maintaining prediction
models of sufficient quality. That is, prediction models which provide the neces-
sary means for an analyst to simulate the impacts of architecture design-related
changes on system quality at both quality attributes and overall quality level. Archi-
tectural adaptions are the main objective of PREDIQT, but the aim is also
to support prediction of effects of low-level design changes. Our intention is that
PREDIQT, combined with a cost-benefit analysis of system changes, should ease
the task of prioritizing between a set of architecture designs, according to their
cost-effectiveness.

This paper reports on experiences from using the PREDIQT method in a
major industrial case study focusing on a so-called “Validation Authority” (VA)
system [19] for evaluation of electronic identifiers (certificates and signatures)
worldwide. We first give an overview of the PREDIQT method, and then present
an evaluation of the method in terms of a feasibility study and a thought exper-
iment (which complemented the feasibility study). Security is one of the three
major quality attributes that the VA quality is decomposed into. The evaluation
focuses on security and its trade-offs with the overall quality attributes identified
and defined with respect to the VA system, namely scalability and availability.
The results indicate that PREDIQT is feasible in practice, in the sense that it
can be carried out on a realistic industrial case and with limited effort and re-
sources. Moreover, results of the evaluation of PREDIQT based on the thought
experiment, are promising.

The paper is organized as follows: An overview of the PREDIQT method is
provided in Section 2. Section 3 presents the feasibility study. Section 4 presents
the setup and results from the thought-experiment. Results from the evaluation
based on the feasibility study and the thought-experiment are discussed in Sec-
tion 5. A discussion of the threats to validity and reliability is undertaken in
Section 6. Section 7 provides an overview of related research. Concluding re-
marks and future work are summarized in Section 8.

The appendices appear in the following order: Appendix 1 provides an overview
of the workshops performed during the trial, with the related activities and the
deliverables. Appendix 2 includes a selection of the design models of the VA,
with their brief explanations. Appendix 3 contains the quality models of the VA, with their full structure and definitions. Appendix 4 includes the conceptual model of the VA. Appendix 5 provides the structure of an attribute specific DV, and the structure of the total quality DV of the VA system (the parameters are omitted due to confidentiality). Schema for documentation of results of the change simulation (the values are omitted due to confidentiality) is provided in Appendix 6. Appendix 7 includes the measurement plan for the VA. Schema for documentation of results of the thought experiment is included in Appendix 8. The process diagram for evaluation of the PREDIQT method is included in Appendix 9.

2 Overview of the PREDIQT Method

This section outlines the PREDIQT method, including its structure, process and outcomes. The process of the PREDIQT method consists of the three overall phases illustrated in Fig. 1. Each of these phases is decomposed into sub-phases. Sections 2.1 and 2.2 present the “Target modeling” and “Verification of prediction models” phases, respectively. The “Application of prediction models” phase consists of the three sub-phases presented in Section 2.3.

2.1 Target modeling

The sub-phases within the “Target modeling” phase are depicted in Fig. 2. The requirement specifications and system design models are assumed to be made available to the analysis team, along with the intended usage, operational environment constraints (if available) and expected nature and rate of changes. An overview of the potential changes helps to:

– limit the scope of the prediction models to the needs of the system quality prediction, and
– focus on the relevant aspects of system quality and design during the target modeling and verification of the prediction models.
The possible architecture design changes (or their causes) include: new technology, modified usage, different dataflow, structural design changes, new components, different usage profile, changed load, changed quality characteristics of system parts, etc.

**Characterize the target and the objectives:** Based on the initial input, the stakeholders involved deduce a high level characterization of the target system, its scope and the objectives of the prediction analysis, by formulating the system boundaries, system context (including the operational profile), system life time and the extent (nature and rate) of design changes expected.

**Create quality models:** Quality models are created in the form of a tree, by decomposing total quality into the system specific quality attributes. A qualitative interpretation and a formal rating of each quality attribute are provided. Each quality attribute relates to a number of indicators, called sub-characteristics. Each sub-characteristic is assigned a qualitative interpretation and a formal measure. The purpose of having both formal and informal parts of the definitions is two-fold:

- putting the attribute and the sub-characteristic definitions into the context of the system under consideration, and
- providing a system-customized and unambiguous basis for incorporating both subjective and objective input (during the later sub-phases).

The quality models are customized for the system under analysis, but the individual definitions and the notation may be reused from established standards (such as [3] and [5]). For completeness purpose, each set of attributes and sub-characteristics having a common parent is supplemented by a node “Other”. All nodes having a common parent are decomposed and defined so that they are orthogonal. The quality models represent a taxonomy with interpretations and formal definitions of system quality notions, but no value assignments to the notions are defined. The output of this sub-phase are “quality models”.

**Map design models:** The initially obtained design models are customized so that:

1. only their relevant parts are selected for use in further analysis, and
2. a mapping within and across high-level design and low-level design models (if available), is made.

The mapped models result in a class diagram which includes the relevant elements and their relations only. The output of this sub-phase are “design models”.

**Create dependency views:** In order to support traceability to (and between) the underlying quality models and the mapped design model, a conceptual model
Model Based Prediction of Impact of Changes on System Quality

(a tree formed class diagram) where classes represent elements from the underlying design and quality models, relations show the ownership, and the class attributes indicate the dependencies, interactions and the properties, is created. The conceptual model is then transformed into a generic DV – a directed tree (without any parameter values assigned) representing relationships among quality aspects and the design of the system. For each quality attribute defined in the quality model, a quality attribute specific DV is created, by a new instantiation of the generic DV. Each set of nodes having a common parent is supplemented with an additional node called “Other”, for completeness purpose. In addition, a total quality DV is deduced from the quality models. The parametrization of the quality attribute specific DVs involves assigning a “degree of Quality attribute or Characteristic Fulfillment” (QCF) value to each leaf node, followed by assigning “Estimated degree of Impact” (EI) values to arcs. A QCF expresses the value of the rating of the attribute in question (the degree of its fulfillment), within the domain of the node. Due to the rating definitions, the values of QCFs are constrained between 0 (minimum) and 1 (maximum). The EI value on an arc expresses the degree of impact of a child node (which the arc is directed to) on the parent node, or to what degree the parent node depends on a child node. An EI value is assigned with respect to the sub-characteristics of the quality attribute under analysis (defined in the quality models) and their respective impact on the relation in question. Once the EIs are evaluated for all arcs directed to nodes having a common parent, their values are normalized so that they sum up to 1, due to model completeness.

The total quality DV is assigned weights on the arcs, which, based on the attribute definitions in the quality models, express the impact of each attribute (in terms of the chosen stakeholder’s gain or business criticality) on the total system quality. The weights are system general objectives. The weights are normalized and sum up to 1, since also this DV is complete. The leaf node QCFs of the total quality DV correspond to the root node QCFs of the respective quality attribute specific DVs. The output of this sub-phase is one DV per quality attribute defined in the quality models and a total quality DV, with assigned estimates. Evaluation of the DV parameters may be based on automated data acquisition or judgments of domain experts.

The set of the preliminary prediction models developed during the “Target modeling” phase, consists of design models, quality models and the DVs, depicted in Fig. 3. The conceptual model is derived from the design models and the quality models. The conceptual model is transformed into a generic DV model, which is then instantiated into as many quality attribute specific DVs as there are quality attributes defined in the quality model (the total number of the quality attributes, and therefore the quality attribute specific DVs, corresponds to “n” on Fig. 3). The total quality model is an instantiation of the top two levels of the quality model. The relationship between the quality attribute DVs and the total quality DV is due to the inheritance of the root node QCF value of each quality attribute DV into the respective leaf node of the total quality DV.
2.2 Verification of prediction models

The “Verification of prediction models” phase (depicted in Fig. 4) aims to validate the prediction models (in terms of the structure and the individual parameters), before they are applied. The prediction models are revised and fitted until an acceptable threshold of deviation between estimates and measurements is reached. Empirical evaluations are made on the influential and uncertain parts of the prediction models. Such a threshold is expressed in terms of the tolerable degree of uncertainty of the predictions, and may be quality attribute specific, or general.

Evaluation of models: A measurement plan with the necessary statistical power is developed, describing what should be evaluated, when, how and by whom. The measurement plan is developed with respect to two major aspects:

1. parts of the DVs having dominating or uncertain values of $QCF \cdot EI$, and
2. feasibility of measurement.

Both internal and leaf nodes of DVs are evaluated, in order to test the correctness of the overall model (with respect to both structure and the inferred QCF values) and the individual estimates. The source of the input has to be more valid and reliable than the one used for estimation. Once the input is ready, it is compared with the estimates (obtained during development of the DVs) and the correspondence is evaluated. In case of uncertainty, additional or repeated measurements are undertaken. The candidates for evaluation are ranked starting from the children of the root node of each DV, and moving down the tree.
Fitting of prediction models: Model fitting is conducted in order to adjust the DV parameters to the evaluation results. The modification candidate parameters (and to some extent the degree of modification) are identified by relating the deviations uncovered during the evaluation to the relevant parts of the conceptual model. Based on this and with reference to the definitions from the quality models, the identified parameters (EIs and leaf node QCFs) are modified by the domain experts and the analyst.

Approval of the final prediction models: The objective of this sub-phase is to evaluate the prediction models as a whole and to validate that they are complete, correct and mutually consistent after the fitting. A deviation threshold expressing the acceptable deviation uncertainty at quality attribute level, is set by the stakeholders in charge. The overall deviation between estimates and measurements is compared to a pre-defined threshold and the structure is examined. If a revision of the design or quality models is needed, the target modeling phase above has to be re-initiated. In case only the DV parameters are above the deviation threshold while the model structure is approved, the verification phase is re-initialized with the fitted prediction models as input. Approval of the prediction models is a prerequisite for proceeding to the next phase. Once the prediction models are approved, they can, in the next phase, be exploited to simulate alternative ways of reaching desired quality levels by modifying system design, in a “what if” manner. Otherwise, the verification sub-phase is re-initialized with the fitted prediction models as input.

2.3 Application of prediction models

During the “Application of prediction models” phase (depicted in Fig. 5), a specified change is applied and simulated on the approved prediction models. The “Application of prediction models” phase is depicted in Fig. 5. During this phase, a proposed change is applied to the prediction models, which allow analysis of its propagation. The simulation reveals which design parts and aspects are affected by the change and the degree of impact (in terms of the quality notions defined by the quality models).

Specify a change: The change specification should clearly state all deployment relevant facts, necessary for applying the change on the prediction models. The specification should include the current and the new state and characteristics of the design elements/properties being changed, the rationale and the assumptions made.

Fig. 5. Application of models – phase
Apply the change on prediction models: This phase involves applying the specified change on the prediction models. The change is applied on the design models first, reflected on the conceptual model and finally on the DVs. In case design model elements are replaced, added or moved, both structure and attributes of the conceptual model may be affected, thus causing the corresponding modifications in structure and parameters of the quality attribute DVs. If only properties or interactions of the design elements are modified, the attributes of the corresponding classes in the conceptual model are affected, and thus the corresponding parameters (EIs and leaf node QCFs) of the DVs are modified (which of the identified parameters are modified on which DVs depends on the relevant definitions provided in the quality model). If the specified change can be fully applied, it is within the scope of the prediction models, which is a prerequisite for proceeding to the next sub-phase. Otherwise, the modifications are canceled and the change deemed not predictable by the models as such. Upon applying a change on the prediction models, we can determine whether it is within the scope of the prediction models, which is a prerequisite for proceeding to the next sub-phase.

Quality prediction: The propagation of the change throughout the rest of each one of the modified DVs, as well as the total quality DV, is performed based on the general DV propagation model, according to which the QCF value of each parent node is recursively calculated by first multiplying the QCF and EI value for each closest child and then summing these products. Such a model is legitimate since each quality attribute DV is complete, the EIs are normalized and the nodes having a common parent are orthogonal due to the structure. The root node QCF values (which represent the system-level rating value of the quality attribute that the DV is dedicated to) on the attribute specific DVs are reflected to the corresponding node of the total quality DV and propagated throughout the total quality DV, according to the above mentioned general DV propagation model. The general DV propagation model can be applied since the total quality DV is complete too, the weights are normalized and the nodes are orthogonal due to their definitions in the quality model. We have developed a tool on the top of Microsoft Excel. It enables automated change propagation within and among DVs, according to the general DV propagation model. Deploying a change which is within the scope of the models does not require new target modeling or verification of the prediction models, but only adoption of the updates made during the simulation. Model erosion due to deployment of the changes within the scope is handled by re-initiating the verification phase. When it should be undertaken depends on the required prediction accuracy, which is beyond the scope of this paper.

Each DV contains two corresponding sensitivity charts, visualizing the degree of impact (on quality attribute represented by the DV) of the individual nodes (QCF) and arcs (EI), respectively. Actual deployment of a change on the system leads to adoption of the new version of the prediction models, which has resulted from the related simulation. However, if the deployed change is outside the scope
of the prediction models, the development of prediction models phase has to be re-initiated in order to re-calibrate the prediction models. Otherwise, impacts of different or additional changes may be simulated by once again initializing the “Application of prediction models” phase.

3 Application of PREDIQT in the Industrial Case

The PREDIQT method was tried out in a major industrial case study focusing on a Validation Authority (VA) [18] system. The VA offers an independent service for digital signatures and electronic ID validation, used worldwide [19]. The case study was held during six workshops, and included four domain experts and one analyst. We focus on illustrating the part of the analysis relevant to the security quality attribute and its trade-offs with the overall attributes due to its central role in the analysis, and importance for the system in question. The VA system was at the moment in a piloting phase.

3.1 Target modeling

Based on the initially obtained documentation (requirement specifications, operational environment specifications, etc.) and several interactions with the domain experts, UML based design models of the VA were developed. These are the design models that PREDIQT presupposes being in place prior to the analysis.

Characterize the target and the objectives: Our target was the VA, the purpose of which is the validation of certificates and signatures in the context of any kind of workflow involving authentication. An overview of the functional properties of the VA, the workflows it is involved in and the potential changes, was provided, in order to determine the needed level of detail and the scope of our prediction models. Among the assumed changes were increased number of requests, more simultaneous users and additional workflows that the VA will be a part of. The target stakeholder group is the system owner.

Create quality models: Requirements specifications, use cases and domain expert judgments were among the influential input in order to define and decompose the quality in a system specific, generic manner. An extract of the quality model of the VA is shown by Fig. 6. Total quality of the VA system is first decomposed into the four quality attributes: availability, scalability, security and “Other Attr.” (this node covers the possibly unspecified attributes, for model completeness purpose). The $X$, $Y$, $Z$ and $V$ represent system quality attribute ratings, while $u$, $v$, $g$ and $j$ represent the normalized weights expressing each attribute’s contribution to the total quality of VA. The attributes and their ratings are defined with respect to the VA system, in such a manner that they are orthogonal and together represent the total quality of the VA system. Thereafter, the sub-characteristics of each quality attribute are defined with respect to the
VA system, so that they are orthogonal to each other and represent a complete decomposition of the VA attribute in question. The sub-characteristics act as indicators, thus the dependency UML notation in Fig. 6. Values and importance of the sub-characteristics are DV node specific. Data encryption is for example more important when estimating QCF and EI of some nodes within a security DV, then others. Both the attribute ratings and the sub-characteristic metrics are defined so that their value lies between 0 (minimum) and 1 (maximum fulfillment). Qualitative and quantitative (VA specific) definitions of security and each one of its sub-characteristics are provided in the UML-based quality models.

Consider the attribute “Security” in Fig. 6. Its weight is $v$ (the coefficient of $Y$ in the “total quality” class). The security attribute definition was based on [4]: “The capability of the software product to protect information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them.”. The security rating was based on [20]. The dimension-notation shown in the form of the security class attributes in Fig. 6, originates from [5]. The dimensions represent the variables constituting the security rating for the VA system. Given the attribute and rating definition, the appropriate (for the attribute and the VA system) sub-characteristics were retrieved from [3], where they are fully defined. $Q$, $W$, $E$, $R$ and $M$ represent the measure definitions of each sub-characteristic. Security depends on the five sub-characteristics displayed: access auditability, access controllability, data encryption, data corruption prevention and “other sub-characteristics” (for model completeness purpose). The access auditability sub-characteristic, for example, expresses how complete the implementation of access login is, con-

Fig. 6. An extract of the quality model
sidering the auditability requirements. Its measure is: Access auditability = \( r \), where \( r = \text{Nr. of information recording access log confirmed in review} \); \( a = \text{Nr. of information requiring access log} \) [3]. Development of quality models involved designing the structure and assigning the VA specific definitions to it. The quality models serve as a reference in the remainder of the analysis.

**Map design models:** The originally developed design models of the VA covered both high-level and low-level design aspects through use cases, class diagrams, composite structure diagrams, activity diagrams and sequence diagrams (SDs). A selection of the relevant models was made, and a mapping between them was undertaken. The model mapping resulted in a class diagram containing the selected elements (lifelines, classes, interfaces etc.) as classes, ownership as relations and interactions/dependencies/properties as class attributes. Due to only ownership representing a relation, this resulted in a tree-formed class diagram, so that the abstraction levels which need to be estimated are refined into the observable ones.

**Create dependency views:** A conceptual model (a tree-formed class diagram) with classes representing the selected elements and relations denoting ownership (with selected dependencies, interactions quality properties, association relationships or their equivalents represented as the class attributes), is deduced from:

1. the quality models, and
2. the above mentioned class diagram.

See Fig. 7 for an extract. It provided an overall model of the dependencies between quality and design aspects, traceable to the respective underlying quality and design models. The deduction of the relations was performed by following a systematic procedure: selecting the relevant parts of the underlying quality models and the design models mapping class diagram, and structuring them so that:

1. the already established relationships within quality and design models are not contradicted, and
2. the nodes having a common parent are orthogonal.

The dependent nodes were thus placed at different levels of this tree structure. The ownership relations in the conceptual model were mainly deduced from the selected generalization, realization, aggregation, composition in the (UML based) design and quality models. A generic DV – a directed tree representing relationships among quality aspects and design of the system was obtained by transforming the conceptual model. Each set of nodes having a common parent was supplemented with an additional node called “other”, for completeness purpose. Quality attribute specific DVs were then obtained by instantiating the generic DV and selecting the relevant subset of its structure. In addition, a total quality DV was instantiated from the quality models. The parametrization of
the quality attribute specific DVs involved assigning a QCF value to each leaf node, followed by assigning EI values on arcs.

Totally four DVs were developed: one total quality DV and three quality attribute specific DVs, that is one for each quality attribute. The node “Other attributes” was deemed irrelevant. The DVs were then inserted into the tool we have built on top of Microsoft Excel for enabling automatic simulations within and across the DVs. DVs were, based on estimates provided by an expert panel, assigned parameter values. QCF values of attribute specific DVs were estimated by assigning a value of the quality attribute (which the DV under consideration is dedicated to) to each leaf node of the quality attribute specific DVs. The QCF value quantification involved revisiting quality models, and providing a quality attribute rating value to each node. The rating value expresses to what degree the quality attribute (given its system specific formal definition) is fulfilled within the scope of the node in question. This was followed by estimation of EI values on each arc of the quality attribute specific DVs, in a bottom-up manner. The EIs expressed the degree of influence of a child node on the parent node. The EI of an arc was assigned by evaluating the impact of the child node on its parent node, with respect to the sub-characteristics (defined in the quality models) of the attribute under consideration. Once a sum of the contributions of the sub-characteristics was obtained on each arc pointing to children nodes with a common parent, the EI values were normalized so that they sum up to 1 (due to model completeness).

Finally, the total quality DV (consisting of a root node for total quality and four children nodes, each for one of the quality attributes) was assigned weights (denoted by \( u \), \( v \), \( g \) and \( j \) in Fig. 6) on arcs. The weights were deduced by first evaluating the contribution (in monetary units for business goals of the
system owner) of each attribute (given its definitions) on the total quality of the system, and then normalizing the values. The leaf node QCFs correspond to the root node QCFs of the respective quality attribute DVs and are automatically reflected on the total quality DV.

Fig. 8 shows an extract of the Security attribute specific DV of the VA. The extract of the DV shown by Fig. 8 is assigned fictitious values on nodes as well as on arcs, for confidentiality reasons. In the case of the “Error detection” node of Fig. 8, the QCF value expresses the effectiveness of error detection with respect to security. The EI value on an arc expresses the degree of impact of the child node (which the arc is directed to) on the parent node, that is to what degree the error handling depends on the error detection. Thus, QCFs as well as EIs of this particular DV are estimated with reference to the definition of Security attribute and its sub-characteristics, respectively. The definitions are provided by the quality models exemplified in Fig. 6. Once estimates of leaf nodes’ QCFs and all EIs are provided, the QCF values on the non-leaf nodes of all the DVs are automatically inferred based on the general DV propagation model.

![Quality attribute: Security](image)

**Fig. 8.** A part of the Security DV.

### 3.2 Verification of prediction models

Verification of the prediction models for the VA relied on empirical input consisting of measurements and expert judgments.

**Evaluation of models:** Once the target modeling was completed, a measurement plan for the VA was developed. The uncertain and dominant parameters
were selected\(^1\), and the appropriate measurements on the VA were precisely defined for each one. The expected values of the measurements were calculated by the analyst, but kept unrevealed for the participants involved in the measurement. Among the measurements performed on the VA were:

1. time to handle 30 simultaneous transactions of varying size and random kind (signature validation or certificate verification), and
2. time taken to download 60 CRLs (Certificate Revocation Lists) simultaneously.

The first one was repeated with different numbers of users. The second one distinguished the time taken by network, server and database. The measurement process was revisited until the empirical values of the parameters selected were obtained. The parameters and their empirical counterparts were then compared, when evaluating their magnitude of average deviation. The measurements were provided in Microsoft Excel sheets, where statistical analysis was performed in order to determine the impact of the file size, number of users, network, server, database and type of request. The mapping between the measurements and the parameters was made according to the quality model and the conceptual model. Microsoft Excel allowed relating the measurements to the parameters and expected values, and automatically analyzing their deviations in terms of measures of tendency and charts. The importance of the usage profile became evident, particularly for scheduling and performance, which are the main factors of the response time. In addition, a walkthrough of the selected internal nodes was made in order to test the automatically deduced values and to increase the statistical power. The measurements were collected and domain experts’ estimates obtained, before the inferred parameter values and statistical analysis were revealed for the domain experts.

**Fitting of the prediction models:** The analysis of the deviations between the models and the measurements were presented in terms of statistical operators and charts. The walkthrough of the analysis resulted in 3-5 slightly fitted parameters per quality attribute specific DV. Specification of values is omitted for confidentiality reasons.

**Approval of the final prediction models:** The fitted parameters on the quality attribute specific DVs were automatically propagated upwards in the four DVs. The inferred values were then compared with the empirical or expected input and judged by the domain experts. Since the QCFs on the root nodes of the attribute specific DVs denote the rating values at the VA system level, their inferred values were compared with the known ones for availability, security and scalability, resulting in a negligible magnitude of average deviation. The approval

\(^1\) The existing estimates of downtime, quality of logging, degree of security breaches, quality of encryption, access controllability, access auditability, security of the database, communication protocol security, etc., were known to be reliable.
of the fitted prediction models was obtained when the domain experts agreed that an acceptable model quality (in terms of the deviation threshold, contents and scope of the models) was reached.

3.3 Application of prediction models

The approved prediction models were applied for simulation of impacts of 14 specified architecture design changes on the VA quality. Each specified architecture design change was first applied on the affected design models, followed by the conceptual model and finally the DVs.

Specify a change: Totally 14 changes were specified, as shown by Table 1. Some of the changes (e.g. change 1) addressed specific architecture design aspects, others referred to the system in general (e.g. change 5), while the overall changes (e.g. changes 6 through 14) addressed parameter specifications of the DVs. The specification suggested each change being independently applied on the approved prediction models.

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Change specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Split signature validation (SV) component into two redundant components, with load balancing.</td>
</tr>
<tr>
<td>2</td>
<td>Merge certificate verification (CV) and signature validation (SV) interfaces (not components).</td>
</tr>
<tr>
<td>3</td>
<td>Deploy redundant VA with external workload balancing.</td>
</tr>
<tr>
<td>4</td>
<td>Average size (nr. of data elements and data amount) of SV requests increases by 100%.</td>
</tr>
<tr>
<td>5</td>
<td>Use of gateway made mandatory.</td>
</tr>
<tr>
<td>6</td>
<td>Decrease error detection QCF for availability by 50%.</td>
</tr>
<tr>
<td>7</td>
<td>Decrease coupling QCF for availability by 50%.</td>
</tr>
<tr>
<td>8</td>
<td>Decrease upgrading QCF for availability by 50%.</td>
</tr>
<tr>
<td>9</td>
<td>Increase scalability QCF under modularity by 66%.</td>
</tr>
<tr>
<td>10</td>
<td>Increase scalability QCF upgrading by 56%.</td>
</tr>
<tr>
<td>11</td>
<td>Increase scalability QCF “Measures for protection of operational environment” by 37%.</td>
</tr>
<tr>
<td>12</td>
<td>Increase security QCF “Logging” by 16%.</td>
</tr>
<tr>
<td>13</td>
<td>Increase security QCF “Monitoring” by 16%.</td>
</tr>
<tr>
<td>14</td>
<td>Increase security QCF “Measures for protection of operational environment” by 5%.</td>
</tr>
</tbody>
</table>

Table 1. Change specification table

Apply the change on prediction models: All the 14 changes were successfully applied on the prediction models, thus showing to be within the scope of the models. Due to the space limitation, this subsection focuses on change nr.
3. We assume that the production environment of the VA is currently not redundant, and there is thus no load balancing. The change implies introducing VA redundancy (that is a duplicate installation of the entire VA system: client, server, database, etc.) and a separate load balancing mechanism. The latter involves distribution and scheduling of all requests by delegating them to one of the two VA installations. The change involved modification of:

1. a composite diagram of the VA environment
2. a class diagram with the VA interfaces, and
3. a sequence diagram (SD) showing the interaction between VA, CA and a relying party.

The last one referenced three additional SDs modeling detailed interactions in relation to certificate validation and signature verification. These modifications of the design models were sufficient for reflecting the specified change on the design diagrams. Since no structural adjustments of the design models were necessary, no overall prediction models were affected in terms of the structure. Next, the affected attributes of the conceptual model were identified and mapped to the corresponding parameters of the quality attribute specific DVs. The parameters were modified with reference to their definitions in the quality model and the changes were documented. Which DVs were affected and the extent of modification of the identified parameters, was dictated by the respective definitions in the quality models. In the case of change 3 from Table 1, all DVs were affected.

<table>
<thead>
<tr>
<th>Change number: 3</th>
<th>Availability (Initial QCF=X)</th>
<th>Security (Initial QCF=Y)</th>
<th>Scalability (Initial QCF=Z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>QCF changed on leaf nodes:</td>
<td>SW recovery a → a'</td>
<td>Redundancy m → m'</td>
<td>VA server g → g'</td>
</tr>
<tr>
<td></td>
<td>Modularity b → b'</td>
<td>Modularity n → n'</td>
<td>Message routing h → h'</td>
</tr>
<tr>
<td></td>
<td>Monitoring c → c'</td>
<td>Modularity i → i'</td>
<td>Modularity o → o'</td>
</tr>
<tr>
<td></td>
<td>Middleware d → d'</td>
<td>Updates j → j'</td>
<td>Upgrading f → f'</td>
</tr>
<tr>
<td>EI changed on arcs to nodes:</td>
<td>Hardware e → e'</td>
<td>Hardware o → o'</td>
<td>Hardware k → k'</td>
</tr>
<tr>
<td></td>
<td>Upgrading p → p'</td>
<td>Monitoring c → c'</td>
<td>Upgrading l → l'</td>
</tr>
<tr>
<td>New root node QCF:</td>
<td>X'</td>
<td>Y'</td>
<td>Z'</td>
</tr>
</tbody>
</table>

Table 2. An outline of the change simulation table

The result of the parameter modifications was a table of the form illustrated by Table 2 with modified EIs and leaf node QCFs on each DV. In the case of the security DV for change 3, two leaf node QCFs were increased due to the duplication of VA: Redundancy and Modularity. These two aspects fulfill (within their respective domains) the security attribute better, given this change. Both redundancy and modularity are now improved, with respect to security (given its VA specific definition from the quality models). Specification of the initial values (denoted by a-p and X-Z) and the values deduced from the above presented steps (denoted by a'-p' and X'-Z') are omitted due to their confidentiality.

Generally, for each quality attribute specific DV:
1. The DV structure was modified in order to maintain consistency with the modified conceptual model (tree formed class diagram mapping the design and quality models) which the DVs are instantiated from.

2. For those leaf nodes that were directly traceable to the affected attributes (which represent properties, interactions and dependencies in the design models) of the conceptual model illustrated by Fig. 7, the leaf node QCFs were modified by the analyst (based on the quality attribute rating definition) if appropriate for the DV in question (recall the quality model).

3. The affected arcs were identified, based on the affected attributes of the conceptual model (illustrated by Fig. 7). The EI values on the affected arcs were changed by the analyst, by re-evaluating the impact of the sub-characteristics of the attribute that the DV is dedicated to and normalizing them on the arcs pointing to the nodes having a common parent. Which DVs were affected and the extent of modification of the identified EIs on them, was determined by the quality models.

4. The modifications, their rationale and the results were documented.

**Quality prediction:** The tool support enabled automated change propagation within and among DVs. The propagation within the DVs involved automatic propagation of the affected QCF values and sensitivity charts, based on the general DV propagation model, while propagation among DVs involved reflecting modifications of attribute specific DVs, on the total quality DV. The impact of a change with respect to quality could be observed from the inferred parameters of the respective DVs. For each change, its predicted effects were documented by a snapshot of the initial prediction models, change application steps, and a change simulation table (an extract is illustrated by Table 2). $X'$, $Y'$ and $Z'$ represent the new values (due to change 3) for availability, security and scalability, respectively. They are equivalent to the predicted root node QCF values on the respective quality attribute specific DVs, given that change 3 is deployed. The new total quality value was then automatically obtained from the total quality DV. Since our specification treated the 14 changes independently from each other, the prediction models were initialized (to the approved ones) prior to starting the simulation of each change.

Each DV contained two corresponding sensitivity charts, visualizing the degree of impact (on the root node QCF) of the individual nodes (QCF) and arcs (EI), respectively. One of the sensitivity charts, showing the impact of the QCF values from some of the second top level nodes of the security attribute DV, is illustrated by Fig. 9. The sensitivity charts show the possible intersection points (and thus the corresponding values of the overall variables as well as the top node value) assuming a changed value of a parameter. A value update on a DV resulted in an automatic update of the sensitivity charts. Effects of each change with respect to quality at all levels above the modified ones, were automatically

---

2 The input values the sensitivity chart is based on, are imaginary due to confidentiality of the DV parameter values.
propagated. The impact of a change with respect to quality could be observed from:

1. the sensitivity charts showing slopes and intersections of the various QCFs and EIs at their respective abstraction levels, or
2. the inferred parameters of the respective DVs.

4 Evaluation Based on a Thought Experiment

As mentioned earlier, we basically did two evaluations. Firstly, we did a feasibility study, as described above, with the objective of evaluating whether PREDIQT could be used in a practical setting and deliver useful results. Secondly, we conducted a thought experiment. This section concentrates on the latter.

Set-up: Due to the involvement of several participants, a certain presence of the human factors was inevitable. Therefore, we briefly present the setup\(^3\). The analysis leader had more than six years of relevant experience in architecture design and software engineering. She served as the method developer and the analyst. She developed the VA prediction models, while interacting with the domain experts and then performed the quality simulations presented in Section 3.3. The domain expert panel consisted of four professionals from DNV [1], with many years of relevant experience within software engineering. Two of the domain expert panel members were involved in the development and verification of the VA prediction models.

\(^3\) As of the initialization of the trial in February 2008
Research method: None of the changes specified had been deployed on this particular system previously, although some freely available experience factories from changes of this kind on similar systems could be recalled by the analysis participants. The simulation (presented in Section 3.3) of impacts of the 14 specified changes on quality of the VA system was performed by the analyst. Both the simulation approach and all the resulting predictions were documented, stored by an additional analysis participant, and kept unrevealed for the domain expert panel until their completion of the thought experiment. The domain expert panel was, by the analyst, given a brief presentation and handouts of the approved prediction models (which they had been involved in the development and verification of) for the VA. The changes were considered independently of each other. For each one of the 14 specified changes we went through the following three steps:

1. The change was presented by the analyst, by only providing the change facts.
2. The preliminary design model modifications (if any) for the change under consideration were proposed by the analyst and then further revised by the domain expert panel.
3. The domain experts were asked to estimate a new root node QCF value (represented by X', Y' and Z' in Table 2) for each quality attribute specific DV, assuming the change is deployed (the specific values obtained are confidential).

Results: The results expressing the magnitude of average deviation $AD = \frac{|E-S|}{E}$ between the PREDIQT based simulations $S$ and the empirical counterparts (thought experiment) $E$, are presented by Table 3. The blank fields indicate that the quality attribute was unaffected by the change and does not influence the statistical values provided at the bottom. The specific $E$ and $S$ values obtained for the attributes are confidential and therefore not revealed here.

5 Discussion

This section discusses the results from both the feasibility study and the thought experiment, presented by Sections 3 and 4, respectively.

Feasibility study: A lightweight post-mortem analysis of the case study was conducted. The domain experts (who are the main stakeholders: service providers, quality engineers and system architects) have expressed that the development (structuring and parametrization) of DVs was relatively simple, thanks to the comprehensible DVs (including the structure of the DVs and the general DV propagation model) as well as the confined underlying quality and design models. One of the main points of the feedback was that the reasoning around DVs, particularly their parametrization, facilitated understanding the system design and reasoning about alternatives for potential improvements, as well as existing and potential weaknesses of architectural design, with respect to quality. We
Table 3. Results of evaluation

<table>
<thead>
<tr>
<th>Change</th>
<th>Availability</th>
<th>Scalability</th>
<th>Security/Tot. quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.005</td>
<td>0.024</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>0.013</td>
<td>0.059</td>
<td>0.014</td>
</tr>
<tr>
<td>3</td>
<td>0.009</td>
<td>0.077</td>
<td>0.025</td>
</tr>
<tr>
<td>4</td>
<td>0.122</td>
<td>0.188</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>0.026</td>
<td>0.033</td>
<td>0.197</td>
</tr>
<tr>
<td>6</td>
<td>0.104</td>
<td></td>
<td>0.053</td>
</tr>
<tr>
<td>7</td>
<td>0.082</td>
<td></td>
<td>0.043</td>
</tr>
<tr>
<td>8</td>
<td>0.053</td>
<td></td>
<td>0.027</td>
</tr>
<tr>
<td>9</td>
<td>0.100</td>
<td></td>
<td>0.023</td>
</tr>
<tr>
<td>10</td>
<td>0.032</td>
<td></td>
<td>0.007</td>
</tr>
<tr>
<td>11</td>
<td>0.031</td>
<td>0.000</td>
<td>0.006</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>0.003</td>
<td>0.001</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>0.008</td>
<td>0.002</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>0.015</td>
<td>0.004</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>0.052</td>
<td>0.068</td>
<td>0.029</td>
</tr>
<tr>
<td><strong>Stdev</strong></td>
<td>0.046</td>
<td>0.055</td>
<td>0.064</td>
</tr>
<tr>
<td><strong>Max</strong></td>
<td>0.122</td>
<td>0.188</td>
<td>0.197</td>
</tr>
<tr>
<td><strong>Min</strong></td>
<td>0.005</td>
<td>0.024</td>
<td>0.000</td>
</tr>
</tbody>
</table>

managed to conduct all the steps of the PREDIQT method, with limited resources and within the planned time period (six half-a-day workshops with up to one man-labour week before each). The changes specified were deduced with the objective of addressing the most relevant parts of the prediction models, being diverse (with some intended overlaps – in order to check for consistency) and realistic. The fact that all the changes were within the scope of the prediction models and could be simulated, indicates feasibility of developing the prediction models with intended scope and quality. Overall, applying PREDIQT was feasible within the practical settings of this case. The models were relatively straightforward to develop, and judged to be fairly easy to use.

**Thought experiment:** The $AD$ values and their standard deviations in Table 3 are quite low, although not negligible. Table 3 indicates a strong consistency between $S$ and $E$ values. $AD$ was, rather than $|E - S|$ used as a measure of deviation, in order to compensate for the quality attribute ratings being limited to a value between 0 and 1. Being defined as the fraction between $|E - S|$ and $E$, $AD$ is necessarily higher than $|E - S|$, thus taking into account the size of the change relative to the initial value of the quality attribute rating. We do not have hard evidence that the predictions were correct, but the results of the thought experiment are promising.
6 Threats to Validity and Reliability

The validity of the findings with respect to (1) the feasibility of the PREDIQT method; and (2) the results of the evaluation of the predictions based on the thought experiment, depends to a large extent on how well the threats have been handled. In addition to reliability threats, four types of validity threats, presented in [9], are addressed: construct validity, conclusion validity, internal validity and external validity. Reliability is concerned with demonstrating that the operations of the case study can be repeated with the same results. Construct validity concerns whether we measure what we believe we measure. Conclusion validity concerns the composition of participants and the statistical analysis. Internal validity concerns matters that may affect the causality of an independent variable, without the knowledge of the researcher. External validity concerns the generalization of findings of this case study to other contexts and environments.

The VA is representative for the systems intended to be within the scope of the PREDIQT method. No particular customizations of the method were needed for this trial. Thus, it should be possible to reapply PREDIQT in another context. The largest threat may be related to the conclusion validity since both the model development and the thought experiment relied on the subjective estimates provided by domain experts. There was turnover among the domain experts, but two of them participated throughout the case study. The simulations themselves were conducted by the method developer before and independently from the thought experiment. The change specification included diverse, non-overlapping changes covering major parts of the prediction models. The quality attribute specific DVs were relatively complex, which minimizes the possibility that the domain experts were able to remember the DVs and thus quickly calculate propagation of the changes 6-14 during the thought experiment. Still, the standard deviations of $AD$ in Table 3 were quite small. All this considered, the risk that the prediction models, the change impact simulations and the thought experiment based estimates were consistently wrong, should be relatively small. Hard evidence in the form of measurements to validate the correctness of the predictions would have been desirable, but this was unfortunately impossible within the frame of this case study. Statistical power was low, due to low number of participants. The careful selection of experienced participants and the variety of the changes might compensate for some of this.

7 Related Work

Research on quantifiable system quality improvement is extensive. Most traditional system quality research has concentrated on defects modeling, and relied on statistical techniques such as regression analysis. [21] presents risk identification techniques like principal component analysis, discriminant analysis, tree-based reliability models and optimal set reduction. Common for all these techniques is that they analyze and, to some degree, predict defects, based on raw low-level data. [5] provides a UML notation for QoS modeling, which has also
been applied in our quality models. PREDIQT is compatible with the established software quality standard [3], which is applied in this case study. The goal/question/metric paradigm [7], [6] is a significant contribution to quality control which also is compatible with PREDIQT and can be used for development of quality models and design of a measurement plan [13], [11].

[14] introduces an approach for decomposing security objectives into security patterns, which again are decomposed into security measures, which serve as indicators. The resulting dependency graphs are only developed for the security quality attribute and with limited traceability to the system design and its quality notions. Still, the idea of pattern reuse would be useful in the PREDIQT context. Pattern based quantitative security assessment is also presented in [22], where threat coverage metrics are associated with security patterns and aggregated to an overall security indicator. Again, this is a solely security oriented approach for quantitative security assessment (and not prediction) with limited traceability to the underlying design and quality notions.

[10] and [17] provide surveys of the software architecture analysis methods (SAAM, ATAM, ALPSM, SAEM etc.). Compared to PREDIQT, they are more extensive and provide a more high-level based architecture assessment of mainly single quality attributes (maintainability or flexibility). Furthermore, they are not predictive, do not incorporate measurement, and quality is defined and quantified differently. ATAM [16], [15], [8] is, for example, more coarse-grained than PREDIQT in terms of both quality definitions and measurement. PREDIQT allows a more fine-grained analysis of several quality attributes and their trade-offs simultaneously and with limited effort. Hence, an integration of the two may be worthwhile examining.

According to [12], most prediction models use size and complexity metrics to predict defects. Others are based on testing data, the quality of the development process, or take a multivariate approach. In many cases, there are fundamental statistical and data quality problems that undermine model validity. In fact, many prediction models tend to model only part of the underlying problem and seriously misspecify it.

[12] argues that traditional statistical approaches are inadequate and recommends causal, holistic models for software defect prediction, using Bayesian Belief Networks (BBNs). However, a drawback both statistical and BBN based models suffer, is their scalability. Providing sufficient data for statistical models requires tedious data acquisition effort and knowledge of statistical techniques. PREDIQT resolves such scalability issues, by relying on a generic process which results in comprehensible and feasible prediction models. No particular expert knowledge regarding theories or tools is necessary. We operate with multi-level prediction models, allowing a wide range of architecture design changes to be simulated and their impacts on quality predicted at multiple abstraction levels. The method can be carried out with limited effort, and still offer a sufficient level of detail. The incorporation of the established notations, techniques and standards in PREDIQT allows for reuse of the known (and possibly already adopted by the system or the organization involved in the analysis) processes and tools.
8 Conclusions and Future Work

This paper has presented PREDIQT – a method for model based prediction of impacts of architecture design changes on system quality. We have also reported on results from applying PREDIQT in a major industrial case study, as well as on a thought experiment. The case study focused on security and its trade-offs with the overall quality attributes of the target system. The results indicate that PREDIQT is feasible in practice, in the sense that it can be carried out on a realistic industrial case and with limited effort and resources. Moreover, the evaluation of PREDIQT based on the thought experiment is promising. We expect PREDIQT to be applicable in several domains of distributed systems with high quality and adaptability demands. Handling of inaccuracies in the prediction models, improving traceability of the models and design of an experience factory, are among the planned and partially initiated future developments.

Acknowledgments. This work has been conducted within the DIGIT (180052/S10) project, funded by the Research Council of Norway. The authors acknowledge the feedback from Olav Ligaarden, Birger Møller-Pedersen and Tormod Vaksvik Håvardsrud.
Bibliography

In Second International Conference on the Quality of Software Architectures, Västerås, Sweden, June 2006.


Appendix 1: Organization of the PREDIQT trial on VA

The trial of PREDIQT on the VA system was organized in totally six workshops, which enabled the covering all the parts of the PREDIQT process. The workshops with the respective objectives, deliverables, prerequisites and required participants are summarized in Table 4.

Each workshop lasted approximately three hours and the average time between each workshop was about a month. In case the goals of the workshops or the prerequisites were not fulfilled according to the schedule or if clarifications were needed before proceeding, additional efforts and interactions took place between the planned workshops. At the beginning of each workshop, an update on the overall schedule for PREDIQT was given, the goals for the current meeting were presented, and information on the work done since the previous meeting was given. Each workshop ended by planning the forthcoming activities/tasks and by setting a date for the next workshop.
<table>
<thead>
<tr>
<th>Workshop</th>
<th>Objective</th>
<th>Prerequisites</th>
<th>Participants</th>
</tr>
</thead>
</table>
| 1        | - The customers presentation of the system.  
- Identify objective and scope of the analysis.  
- Identify goals of the system.  
- Characterize main quality characteristics, operational profile, expected lifetime and expected extensions of the system.  
- Provide the system documentation.  
- Appoint the communication points and establish communication routines.  
- Appoint the head of the customer group. | - A brief introduction of the PREDIQT method with process, objectives and capabilities, is given.  
- Commitment of the management and the domain experts.  
- The necessary resources made available.  
- No interest conflicts. | Analyst  
- All domain experts who will participate throughout the trial (staff responsible for design architecture, testing, project management, system operation and integration), |
| 2        | - Presentation of the target system: scope, design models and initial quality models, initial dependency view structure - by the analyst.  
- Customers review of the models, followed by a revision, if needed.  
- Views decomposition up to the measurable and needed detail. | - Sufficient feedback from the domain experts for final update of the models (without estimates). | Analyst  
- All domain experts. |
| 3        | - Approval of the initial prediction models (without estimates) of the system.  
- Estimation of the parameter values on the dependency views. | - The prediction models are updated according to the feedback from the previous workshop. | Analyst  
- All domain experts. |
| 4        | - Presentation of the estimated prediction models of the system.  
- Preparation of a measurement plan regarding the current and the changed system. | - Estimation by several appointed domain experts is finalized. | Analyst  
- Head of the domain expert group.  
- Domain experts responsible for data acquisition (related to evaluation). |
| 5        | - Model evaluation and fitting. | - Measurement of the actual QoS values, completed.  
- Statistical analysis completed. | Analyst  
- All domain experts. |
| 6        | - Applying the method for simulation of change impacts on quality. | - The fitted prediction models are approved. | Analyst  
- All domain experts. |

Table 4. Organization of the PREDIQT trial on the VA
Appendix 2: A selection of the design models of the VA

This section contains a selection of the main design models of the VA system, developed in “Rational Software Modeller 6.0.1” tool:

– The use case shown in Fig. 10 models the scope of the analysis with respect to the stakeholders and processes involved.
– The class diagram shown in Fig. 11 models the classification of the quality notions for certificates and signatures handled by the VA system.
– The class diagram shown in Fig. 12 models the gateway of the VA system and its environment, with the respective operations available.
– The class diagram shown in Fig. 13 models the VA and the CA (certificate authority) with the respective environments.
– The class diagram shown in Fig. 14 models the interfaces of the VA.
– The class diagram shown in Fig. 15 models the database of the VA.
– The class diagram shown in Fig. 16 models the interfaces of the gateway.
– The composite diagram shown in Fig. 17 models the VA environment with the interactions.
– The composite diagram shown in Fig. 18 models the VA Gateway environment with the interactions.
– The sequence diagram shown in Fig. 19 models the request types.
– The sequence diagram shown in Fig. 20 models the certificate verification request.
– The sequence diagram shown in Fig. 21 models the certificate verification response.
– The sequence diagram shown in Fig. 22 models the signature validation process.
– The activity diagram shown in Fig. 23 models the certificate verification process.
– The activity diagram shown in Fig. 24 models the signature validation process.
– The class diagram shown in Fig. 25 models the data format of the signature validation.

These models were developed in the beginning of the trial, and actively used during development of the conceptual model and the dependency views, as well as during simulation of the changes. When applying changes on the prediction models, a modification of the design models was the first step, in order to identify the affected design elements and interactions, before reflecting them on the overall prediction models.
Fig. 10. A use case defining the scope, relative to stakeholders and processes
Fig. 11. Signature and certificate quality classification
Fig. 12. Gateway with the environment.

For purpose of confidentiality and response time.
Fig. 13. VA and CA with environments
Fig. 15. VA database
Fig. 16. Gateway interfaces
Fig. 17. VA environment composite diagram
Fig. 18. VA Gateway environment composite diagram.

Request signing is optional. Response signing is mandatory.
Fig. 19. Request types sequence diagram
Fig. 20. Certificate verification request sequence diagram
Fig. 21. Certificate verification response sequence diagram
Fig. 22. Signature validation sequence diagram
Fig. 23. Certificate verification activity diagram
Fig. 24. Signature validation activity diagram
Fig. 25. Signature validation data format.
Appendix 3: The quality models of the VA

The structure and definitions of the quality models for the VA system are provided below. The models are developed in “Rational Software Modeller 6.0.1.” tool. The total quality is decomposed into the three quality attributes, as shown by Fig. 26. The total quality is defined as “The totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs” [2].

The “Security” quality attribute of the VA system is decomposed as shown by Fig. 27. The security attribute definition was based on [4]: “The capability of the software product to protect information and data so that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them.”. The security rating was:

$$\sum_{i}^{I} \sum_{j}^{J} \frac{\text{Size}(i) \cdot \text{Op}(j)}{\text{Size}(i) \cdot \text{Op}(j) + \forall(i(SBL_i, W))}$$

where:
- SBL is Security Breach Level
- W is weight
- i is the index for the component size
- j is the index for the operation
- I is the total number of the components
- J is the total number of the operations.

Most of the security sub-characteristics were assigned an internal and an external measure, based on [3].

For “Access auditability”:
- Internal: “How complete is the implementation of access login instances considering the auditability requirements?” (nr of information recording access log confirmed in review)/(nr of information requiring access log) [3, Part 3, p. 7]
- External: “How complete is the audit trail concerning user access to the system and data” (nr of user accesses recorded)/(nr of user accesses done) [3, Part 2, p. 22-23]

For “Access controllability”:
- Internal: “How complete is the detection of user access to the system” (nr of incorrect/illegal operations detected)/(nr of incorrect/illegal operations to be detected) [3, Part 3, p. 7]
- External: “How complete is the detection of user access to the system” (nr of incorrect/illegal operations detected)/(nr of incorrect/illegal operations anticipated in the specification) [3, Part 2, p. 22-23]

For “Data corruption prevention”: 
Fig. 26. Decomposition of the total quality of the VA into attributes
Model Based Prediction of Impact of Changes on System Quality

– Internal: “The ratio of implemented data corruption prevention in operations to the total number of operations capable of corrupting data” [3, Part 3, p. 7]
– External: “How often do fatal data corruptions occur?” Frequency of data corruption events = 1-(nr of major data corruption events)/(nr of test cases tried to occur data corruption events)) [3, Part 2, p. 22-23]

For data encryption: Internal only: “What is the data encryption ratio?” (nr of data items implementing data encryption/detection facility)/(nr of data items requiring data encryption/decryption facility) [3, Part 2 and Part 3, p. 7]

The “Availability” quality attribute of the VA system is decomposed as shown by Fig. 28. The availability attribute definition was “The degree to which the VA system and its parts are operational and accessible when required for use”. Uptime, as well as service continuity had to be taken into account, and downtime was defined as “incorrect operation time (planned or unplanned), or lasting of a failure”. The availability rating was:

\[
\text{Availability} = \frac{\text{uptime}}{\text{uptime} + \text{downtime}}
\]

The measures of the sub-characteristics under availability were as follows:

– Mean down time: “How long is usually system down?” (Total down time)/(nr. of observed breakdowns) [3, Part 2, p. 34]
– Recovery: “How long will it take to recover if system went down?” Mean time to recover = (total sum of time to recover)/(number of recovery attempts) [3, Part 2, p. 35]
– Restorability: “How is the product capable to restore in defined cases?” (nr. of cases capable to restore)/ (required nr. of cases capable to restore) [3, Part 3, p. 13]
– Restore effectiveness: “How effective will the restoration process be?” (nr. of restore cases meeting the target time)/(number of restore cases which shall meet target restore time) [3, Part 3, p. 13]

The “Scalability” quality attribute of the VA system is decomposed as shown by Fig. 29. The scalability attribute definition was “The ability of the system to support rapid and extensive variations in the number of users, without requiring any changes” [20]. The scalability rating was two-fold:

– “The maximum number of simultaneous inquiries (Certificate verification + signature validation requests) supported by the system (without unacceptable degradation of quality of service level), before any changes need to be made (distinguishing between http and ssl related transaction loads and taking into account number of internal transactions).”
– “The maximum number of certificate authorities supportable by the system (without unacceptable degradation of quality of service level), before any changes need to be made.”

The measures of the sub-characteristics under scalability were as follows:
Fig. 27. Decomposition of the security quality attribute for the VA, into sub-characteristics
Fig. 28. Decomposition of the availability quality attribute for the VA into sub-characteristics.
Response time: “The estimated time to complete a specified task.” [3, Part 3, p. 23]
Throughput time: “The estimated number of tasks that can be performed over a unit of time.” [3, Part 3, p. 23]
Turnaround time: “The estimated time to complete a group of related tasks as a job lot.” [3, Part 3, p. 23]
Fig. 29. Decomposition of the scalability quality attribute for the VA, into sub-characteristics
Appendix 4: The conceptual model of the VA

The conceptual model of the VA is displayed in Fig. 30. The conceptual model merges the quality and the design models, prior to its transformation to the generic DV, which is then instantiated into the respective quality attribute specific DVs. The conceptual model was actively used in relation to model fitting and particularly when applying the prediction models, in order to identify relationships between the different parts of the prediction models of the VA system.
Fig. 30. The conceptual model of the VA
Appendix 5: Structure of the DVs

Since the parameter values are confidential, we only provide the structure of the quality attribute specific DVs (same structure for all the three quality attribute specific DV of the VA, as shown in Fig. 31) and the total quality DV (shown in Fig. 32). The former is an instantiation of the conceptual model, while the latter is an instantiation of the top two levels of the quality model. On both DVs shown in Figures 31 and 32, the parameter values are initialized due to their confidentiality.
Fig. 31. Structure of a quality attribute specific DV of the VA
Fig. 32. Structure of the total quality DV of the VA
Appendix 6: Schema for documentation of results of the change simulation

The results of simulation of the 14 specified changes were documented in a table, shown by Figures 33, 34 and 35. All the updated parameters are displayed here. The simulation took place prior to the thought experiment, was performed by the analyst (who did not perform the thought experiment), stored by an additional (independent) participant, and kept unrevealed until completion of the thought experiment. Each change was simulated independently. The actual values are not shown due to their confidentiality. The letters are displayed instead and their (re)use is coincidental. As a result, it is here invisible whether the updates involved increase or decrease of the original (unmarked) parameter values, and to what degree. The marked letters represent the updated parameter values.

In the case of change 1, three design models were modified, as shown by Figures 36, 37 and 38, respectively.

In the case of change 2, two design models were modified, as shown by Figures 39 and 40, respectively.

In the case of change 3, one sequence diagram was modified, as shown by Fig. 41. Change 3 also involved a modification of the diagrams shown by Figures 14 and 17, in form of a duplication of all the elements included in these two figures.

In the case of change 4, one design model was modified, as shown by Fig. 42.

The overall changes (5-14) did not require modifications of the design models.
<table>
<thead>
<tr>
<th>Change number</th>
<th>Change description.</th>
<th>Availability (Current QCF=X)</th>
<th>Scalability (Current QCF=Z)</th>
<th>Security (Current QCF=Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Split signature verif. component into two redundant components, with load balancing</td>
<td>QCF changed on nodes: Modularity a-&gt;a' Redundancy b-&gt;b' Middleware support servers c-&gt;c' SW recovery d-&gt;d' EL changed on nodes: =&gt; Avail QCF = X'</td>
<td>QCF changed on nodes: Updates e-&gt;e' Modularity a-&gt;a' message routing f-&gt;f' EL changed on nodes: Measures for prot of op env g-&gt;g' Message routing h-&gt;h' =&gt; Scal QCF = Z'</td>
<td>Unchanged.</td>
</tr>
<tr>
<td>2</td>
<td>Merge CV and SV interfaces (not components)</td>
<td>QCF changed on nodes: Modularity a-&gt;a' Scheduling i-&gt;i' Middleware supp services c-&gt;c' Message routing f-&gt;f' Updates j-&gt;j' EL changed on nodes: none =&gt; Avail QCF = X'</td>
<td>QCF changed on nodes: Updates j-&gt;j' Modularity k-&gt;k' Coupling l-&gt;l' Cohesion m-&gt;m' Scheduling n-&gt;n' Middleware supp services p-&gt;p' EL changed on nodes: none =&gt; Scal QCF = Z'</td>
<td>QCF changed on nodes: Access rights a-&gt;a' Data encryption b-&gt;b' Message routing c-&gt;c' Error detection d-&gt;d' Error messaging e-&gt;e' Modularity f-&gt;f' EL changed on nodes: none =&gt; Sec QCF = Y'</td>
</tr>
<tr>
<td>3</td>
<td>Redundant VAS with external workload balancing</td>
<td>QCF changed on nodes: SW recovery a-&gt;a' Modularity b-&gt;b' Redundancy c-&gt;c' Front end server d-&gt;d'</td>
<td>QCF changed on nodes: Redundancy (meas for prot of op env) a-&gt;a' VA server b-&gt;b' Message routing c-&gt;c'</td>
<td>QCF changed on nodes: Measures for prot of op env (redundancy) a-&gt;a' Modularity b-&gt;b'</td>
</tr>
<tr>
<td>Change number</td>
<td>Change description</td>
<td>Availability: Current QCF=X</td>
<td>Scalability: Current QCF=Z</td>
<td>Security: Current QCF=Y</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------</td>
<td>----------------------------</td>
<td>---------------------------</td>
<td>------------------------</td>
</tr>
</tbody>
</table>
| 4             | Average size (nr. of data elements and data amount) of SV requests increases by 100%. | Monitoring e->e'  
Middleware support services f->f'  
EI changed on nodes:  
Hardware g->g'  
Middleware support serv. h->h'  
Upgrading i->i'  
=> Avail QCF = X | Scheduling d->d'  
Datab. redundancy e->e'  
SW recovery f->f'  
Modularity g->g'  
Coupling h->h'  
Cohesion i->i'  
Updates j->j'  
EI changed on nodes:  
Hardware k->k'  
Middleware support services l->l'  
Upgrading m->m'  
=> Scal QCF = Z' | EI changed on nodes:  
Hardware e->e'  
Upgrading d->d'  
Measures for prot. of env e->e'  
=> See QCF = Y' |
| 5             | Use of gateway made mandatory | QCF changed on nodes:  
Network e->e' | QCF changed on nodes:  
Scheduling b->b'  
Message routing c->c'  
Network d->d'  
EI changed on nodes:  
Network e->e'  
Measures for prot. of env f->f'  
Message routing g->g'  
=> Scal QCF = Z' | QCF changed on nodes:  
Network e->e'  
Measures for prot. of env f->f'  
Message routing g->g'  
=> Scal QCF = Z' | QCF changed on nodes:  
Network e->e'  
Measures for prot. of env f->f'  
Message routing g->g'  
=> Scal QCF = Z' |

Fig. 34: Change simulation table - Part 2
<table>
<thead>
<tr>
<th>Change number</th>
<th>Change description.</th>
<th>Availability (Current QCF=X)</th>
<th>Scalability (Current QCF=Z)</th>
<th>Security (Current QCF=Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Decrease error detection QCF for availability by 50%</td>
<td>X'</td>
<td>Message routing a-&gt;a'</td>
<td>Message routing a-&gt;a'</td>
</tr>
<tr>
<td>7</td>
<td>Decrease coupling QCF for availability by 50%</td>
<td>X'</td>
<td>Coupling b-&gt;b'</td>
<td>EI changed on nodes:</td>
</tr>
<tr>
<td>8</td>
<td>Decrease upgrading QCF for availability by 50%</td>
<td>X'</td>
<td>Cohesion c-&gt;c'</td>
<td>Measure for prot. of</td>
</tr>
<tr>
<td>9</td>
<td>Increase scalability QCF modularity by 56%</td>
<td>X</td>
<td>Gateway e-&gt;e'</td>
<td>op.env. b-&gt;b'</td>
</tr>
<tr>
<td>10</td>
<td>Increase scalability QCF upgrading by 56%</td>
<td>X</td>
<td>EI changed on nodes:</td>
<td>Upgrading c-&gt;b'</td>
</tr>
<tr>
<td>11</td>
<td>Increase scalability QCF &quot;measures for protection of operational environment&quot; by 37%</td>
<td>X'</td>
<td>Gateway d-&gt;d'</td>
<td>Gateway d-&gt;d'</td>
</tr>
<tr>
<td>12</td>
<td>Increase security QCF &quot;logging&quot; by 16%</td>
<td>Y</td>
<td>=&gt; Sec QCF = Y'</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Increase security QCF &quot;monitoring&quot; by 16%</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Increase security QCF &quot;measures for protection of operational environment&quot; by 8%</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Fig. 36. Design modification due to change 1 - Part 1
Fig. 37. Design modification due to change 1 - Part 2
Fig. 38. Design modification due to change 1 - Part 3
Fig. 39. Design modification due to change 2 - Part 1
Fig. 40. Design modification due to change 2 - Part 2
Fig. 41. Design modification due to change 3
Average size (nr. of data elements and data amount) of SV requests increases by 100%.

Fig. 42. Design modification due to change 4
Appendix 7: The measurement plan for VA

The measurement plan for the VA was based on the needs for evaluation, and feasibility of the measurement. The latter caused several revisions of the plan, the final version of which is shown by Figures 43, 44 and 45, respectively.
<table>
<thead>
<tr>
<th>Node/ DV</th>
<th>Availability</th>
<th>Scalability</th>
<th>Security</th>
</tr>
</thead>
</table>
| Overall | Downtime over a period of time. May use a scale: | Transaction times of certificate validations and signature verification (dependent on how large document (SV) and which RespondWith items that are included) | Nr. of security breaches over the past x period:  
- Level 1  
- Level 2  
- Level 3  
A breach: ineligible system or data access, system or data corruption etc. |
| Updates | Measure the time the system is unavailable due to updates over a period of time | Downloading time of CRLs  
When heavy download: many concurrent CRL downloadings | Number of missing or incorrect updates of users, messages CRLs etc. over period x |
<p>| Upgrading | The time the system is unavailable due to upgrading – this can be found in ADSS log files. It might be difficult to differentiate between the different nodes cohesion, coupling, and modularity. | The degree (in terms of level and duration), to which upgrading has been a hinder to the needed scalability | Number (per degree) of security breaches due to upgrading |
| Coupling | | Number of security breaches due to unsatisfactory or unnecessary cross-component interactions? |
| Monitoring | To what degree is the system unavailable due to weaknesses of the monitoring mechanism (logging and error handling)? | To what degree is the system unable to scale due to weaknesses of the monitoring mechanism (logging and error handling)? | To what degree is the system unable to handle security breaches due to weaknesses of the monitoring mechanism (logging and error handling)? |
| Repository (Logging) | Percentage of repository availability. | | |
| Statistical data (Logging) | Percentage of availability relevant statistics | | |
| Db recovery (Recovery mechanisms) | Loss of availability due to insufficient db recovery | | |
| SW recovery | Loss of availability due to | | |</p>
<table>
<thead>
<tr>
<th>Node/ DV</th>
<th>Availability</th>
<th>Scalability</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Recovery mechanisms)</td>
<td>insufficient sw recovery.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error detection (Error handling)</td>
<td>Nr. of availability relevant errors detected, in relation to the number of the actual ones.</td>
<td>Nr. of scalability relevant errors detected, in relation to the number of the actual ones.</td>
<td>Nr. of security relevant errors detected, in relation to the number of the actual ones.</td>
</tr>
<tr>
<td>Error messaging (error handling)</td>
<td>Nr. of availability relevant errors which are timely reported, in relation to the number of the detected ones.</td>
<td>Nr. of scalability relevant errors which are timely reported, in relation to the number of the detected ones.</td>
<td>Nr. of security relevant errors which are timely reported, in relation to the number of the detected ones.</td>
</tr>
<tr>
<td>Database efficiency mechanisms</td>
<td></td>
<td>To what degree (percentage in duration per scale) is the system unable to scale due to limitations in database?</td>
<td>Number of security breaches (per security level) due to database related weaknesses.</td>
</tr>
<tr>
<td>Scheduling (Database efficiency mechanisms)</td>
<td>Percentage of detected availability relevant errors caused by weak scheduling.</td>
<td>Percentage of detected performance relevant errors caused by weak scheduling.</td>
<td>Nr. of message or data corruption events due to scheduling.</td>
</tr>
<tr>
<td>Search mechanisms (Database efficiency mechanisms)</td>
<td>Time to look up in the database? – Can be taken from log files</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message routing</td>
<td>Transaction time?</td>
<td>Messages delayed due to routing problems?</td>
<td>Messages lost or incorrectly delivered due to routing problems?</td>
</tr>
<tr>
<td>OS support services</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Middleware support services</td>
<td>Percentage of availability relevant errors due to middleware weaknesses.</td>
<td>Percentage of scalability relevant errors due to middleware weaknesses.</td>
<td>Percentage of security relevant errors due to middleware weaknesses.</td>
</tr>
<tr>
<td>Hardware security module (hardware)</td>
<td>Percentage of downtime due to HSM weaknesses.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Front end server (hardware)</td>
<td>Percentage of downtime due to front and server weaknesses</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VA server (hardware)</td>
<td>Percentage of downtime due to VA server weaknesses</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>Networks availability – any downtime is reported in the monthly service report</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Measures for protection of op. env.</td>
<td></td>
<td></td>
<td>Percentage of the security breaches due to operational env. weaknesses, or that could have</td>
</tr>
<tr>
<td>Node/ DV</td>
<td>Availability</td>
<td>Scalability</td>
<td>Security</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>Firewalls (VAS: Measures for protection of op. env.)</td>
<td>Percentage of the downtime caused by the firewall.</td>
<td>How soon a new firewall opening can be in place? (or is that update?)</td>
<td>Percentage of the security breaches due to firewall weaknesses, or that could have been prevented by firewall.</td>
</tr>
<tr>
<td>Physical protection (VAS: Measures for protection of op. env.)</td>
<td>Percentage of availability degradation due to weak physical security.</td>
<td>Percentage of scalability degradation due to weak physical security.</td>
<td>Percentage of security breaches due to weak physical security.</td>
</tr>
<tr>
<td>Data encryption (VAS: Measures for protection of op. env.)</td>
<td>Percentage of availability degradation due to encryption weaknesses.</td>
<td>How many concurrent SSL connections to the service (stress testing)?</td>
<td>Percentage of security breaches due to weak data encryption.</td>
</tr>
<tr>
<td>Redundancy (VAS: Measures for protection of op. env.)</td>
<td>Percentage of availability degradation due to weak redundancy.</td>
<td>- Time to add new users</td>
<td>Percentage of the total number (scaled) security breaches caused by weaknesses in user management (provisioning, user activation and access right management).</td>
</tr>
<tr>
<td>User management</td>
<td>Percentage of availability degradation due to general user management weaknesses.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provisioning (User management)</td>
<td>Percentage of availability degradation due to provisioning weaknesses.</td>
<td></td>
<td>Fraction of the cases of incorrect provisioning, in relation to the total number of provisioning jobs.</td>
</tr>
<tr>
<td>User activation (User management)</td>
<td>How fast a user can be added an activated?</td>
<td></td>
<td>Fraction of the cases of incorrect (or missing) user activation, in relation to the total number of user activations.</td>
</tr>
<tr>
<td>VAS: Access rights management (User management)</td>
<td>Percentage of availability degradation due to access rights weaknesses.</td>
<td></td>
<td>Fraction of the cases of incorrect (or missing) access rights activities, in relation to the total number of access rights activities.</td>
</tr>
<tr>
<td>Gateway</td>
<td>Percentage of availability degradation due to gateway weaknesses.</td>
<td></td>
<td>Percentage of gateway failures, in relation to the nr of successfully performed tasks.</td>
</tr>
</tbody>
</table>
Appendix 8: Schema for documentation of results of the thought experiment

The schema for documentation of results of the thought experiment is shown in Fig. 46. The domain expert group was asked to consider each specified change independently, agree upon the design model modifications (if any), and finally to agree upon an estimated new value of the root node QCF (represented by the question marks on Fig. 46) on the respective DVs, which could have possibly been affected. Some of the changes explicitly implied modifications of only a single specified DV, which is why some of the fields on Fig. 46 do not contain question marks. The approved prediction models were available to each domain expert. In addition, the relevant current root node QCF was informed about by the analyst. The discussion about each estimate took approx. 2-5 minutes.
<table>
<thead>
<tr>
<th>Change number</th>
<th>Change description</th>
<th>Availability (Current QCF=X)</th>
<th>Scalability (Current QCF=Z)</th>
<th>Security (Current QCF=Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Split signature verification component into two redundant components, with load balancing.</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>2</td>
<td>Merge CV and SV interfaces (not components)</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>3</td>
<td>Introduce redundant VAS with external workload balancing</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>Average size (nr. of data elements and data amount) of SV requests increases by 100%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>5</td>
<td>Use of gateway made mandatory</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>6</td>
<td>Decrease error detection QCF for availability by 50%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>7</td>
<td>Decrease coupling QCF for availability by 50%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>8</td>
<td>Decrease upgrading QCF for availability by 50%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>9</td>
<td>Increase scalability QCF under modularity by 66%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>10</td>
<td>Increase scalability QCF upgrading by 56%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>11</td>
<td>Increase scalability QCF &quot;measures for protection of operational environment&quot; by 37%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>12</td>
<td>Increase security QCF &quot;logging&quot; by 16%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>13</td>
<td>Increase security QCF &quot;monitoring&quot; by 16%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>Increase security QCF &quot;measures for protection of operational environment&quot; by 5%</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>
Appendix 9: The process diagram for evaluation of the PREDIQT method

Fig. 47 shows an activity diagram for the overall plan for evaluation of PREDIQT. The method evaluation model shown assumes that the prediction models are approved.

The thought experiment based evaluation was, for each change in the change specification, carried out according to Fig. 47. The results of PREDIQT simulations were only presented when all the planned thought experiments and measurements had taken place.

Two aspects are crucial when performing a taught experiment:

- The participants’ insecurity should be kept within an acceptable threshold.
- The simulation results from prediction models should not be revealed for the participants prior to the taught experiment execution.

In general, the acceptable number of repetitions can be calculated from statistical power analysis. The number obtained is in most cases much higher than what is feasible or realistic. Therefore, the selection of changes and the number of repetitions should be result of a thorough planning.
Fig. 47. The PREDIQT method evaluation plan
Chapter 10

Paper 2: A Practical Approach to Uncertainty Handling and Estimate Acquisition in Model-based Prediction of System Quality
A Practical Approach to Uncertainty Handling and Estimate Acquisition in Model-based Prediction of System Quality

Aida Omerovic∗† and Ketil Stølen∗†
∗SINTEF ICT, P.O. Box 124, 0314 Oslo, Norway
†University of Oslo, Department of Informatics, P.O. Box 1080, 0316 Oslo, Norway
Email: {aida.omerovic,ketil.stolen}@sintef.no

Abstract—Our earlier research indicated the feasibility of applying the PREDIQT method for model-based prediction of impacts of architectural design changes on system quality. The PREDIQT method develops and makes use of so-called prediction models, a central part of which are the “Dependency Views” (DVs) – weighted trees representing the relationships between architectural design and the quality characteristics of a target system. The values assigned to the DV parameters originate from domain expert judgements and measurements on the system. However, fine-grained, the DVs contain a certain degree of uncertainty due to lack and inaccuracy of empirical input. This paper proposes an approach to the representation, propagation and analysis of uncertainties in DVs. Such an approach is essential to facilitate model fitting (that is, adjustment of models during verification), identify the kinds of architectural design changes which can be handled by the prediction models, and indicate the value of added information. Based on a set of criteria, we argue analytically and empirically, that our uncertainty handling approach is comprehensible, sound, practically useful and better than any other approach we are aware of. Moreover, based on experiences from PREDIQT-based analyses through industrial case studies on real-life systems, we also provide guidelines for use of the approach in practice. The guidelines address the ways of obtaining empirical estimates as well as the means and measures for reducing uncertainty of the estimates.

Keywords—uncertainty, system quality prediction, modeling, architectural design, change impact analysis, simulation.

I. INTRODUCTION

An important aspect of quantitative prediction of system quality lies in the appropriate representation, propagation and interpretation of uncertainty. Our earlier work has addressed this issue by proposing an interval-based approach to uncertainty handling in model-based prediction of system quality [1]. This paper extends the interval-based approach to uncertainty handling with two major tightly related issues:

- uncertainty analysis, and
- practical guidelines for use of the interval-based approach, addressing both the uncertainty handling and the estimate acquisition.

We have developed and tried out the PREDIQT method [2], [3] for model-based prediction of impacts of architectural design changes on system quality characteristics and their trade-offs. Examples of quality characteristics include availability, scalability, security and reliability. Among the main artifacts of the PREDIQT method are the Dependency Views (DVs). The DVs currently rely on sharp parameter values which are based on empirical input. As such, the parameters assigned to the DVs are not very reliable, thus providing predictions of unspecified certainty.

Since the input to the DVs is based on both measurement-based data acquisition (measurements, logs, monitoring, historical data, or similar) and expert judgements, the representation of the uncertain input should be intuitive, as exact as possible and provide a well-defined (complete and sound) inferring mechanism. In a real-life setting, finding the right balance between accuracy and practical feasibility is the main challenge when selecting the appropriate approach to uncertainty handling in prediction models. We propose an approach to deal with uncertainty which, as we will argue, is both formally sound and practically applicable in the PREDIQT context. Our approach is based on intervals with associated confidence level, and allows representation, propagation and analysis of all the parameters associated with uncertainty.

Input acquisition is in this context concerned with how the DV estimates and their uncertainty measures are obtained in practice. An overview of the practical means and measures for 1) acquiring the input and 2) achieving a specified minimum level of uncertainty, is clearly a prerequisite for applicability of the uncertainty handling approach. Therefore, we also provide guidelines for practical use of our solution, covering both the issues of estimate acquisition and uncertainty handling. The guidelines build on the experiences from the empirical evaluations of the PREDIQT method.

The paper is organized as follows: The challenge of uncertainty handling in the context of the PREDIQT method is characterized in Section II. We define the frame within which the approach should be applicable, by providing an overview of the PREDIQT method and in particular the DVs, introducing the notion of uncertainty, and outlining a set of success criteria. The interval-based approach to uncertainty handling is presented in Section III. Section IV argues for the usefulness and practical applicability of the approach by evaluating it with respect to the success criteria. An extensive
number of the candidate methods for uncertainty handling have been systematically reviewed prior to the proposal of our approach. Section V substantiates why our approach, given the criteria outlined in Section II, is preferred among the alternative ones. Practical guidelines for use of our solution, based on lessons learned from PREDIQT-based analyses on real-life systems, are provided in Section VI. The concluding remarks and the future work prospects are given in Section VII.

II. THE CHALLENGE

Our earlier work indicates the feasibility of applying the PREDIQT method for model-based prediction of impacts of architectural design changes, on the different quality characteristics of a system. The PREDIQT method produces and applies a multi-layer model structure, called prediction models. The PREDIQT method is outlined in the next subsection. Uncertainty and the evaluation criteria for the uncertainty handling approach are thereafter presented in dedicated subsections.

A. Overview of the PREDIQT method

The PREDIQT method defines a process and a structure of prediction models. These two perspectives are presented in the following.

1) The process of the PREDIQT method: The process of the PREDIQT method consists of three overall phases as illustrated by Figure 1. Each of these phases is decomposed into sub-phases.

The sub-phases within the “Target modeling” phase are depicted in Figure 2. Based on the initial input, the stakeholders involved deduce a high-level characterization of the target system, its scope and the objectives of the prediction analysis, by formulating the system boundaries, system context (including the usage profile), system lifetime and the extent (nature and rate) of design changes expected. Quality Model diagrams are created in the form of a tree, by decomposing total quality into the system specific quality characteristics, their respective sub-characteristics and indicators. The Quality Model diagrams represent a taxonomy with interpretations and formal definitions of system quality notions. The initially obtained Design Models are customized so that (1) only their relevant parts are selected for use in further analysis; and (2) a mapping within and across high-level design and low-level Design Models (if available), is made. The mapped models result in a class diagram, which includes the relevant elements and their relations only. A conceptual model (a tree-formed class diagram) in which classes represent elements from the underlying Design Models and Quality Models, relations represent the ownership, and the class attributes represent the dependencies or the properties, is created.

For each quality characteristic defined by the Quality Model, a quality characteristic specific DV is created via the instantiation of the conceptual model. A DV is basically a weighted dependency tree which models the relationships among quality characteristics and the design of the system. The instantiation of the conceptual model into a DV is performed by selecting the elements and relationships which are relevant to the quality characteristic being addressed by the DV. Each set of nodes having a common parent is supplemented with an additional node called “Other” for completeness purpose. The DV parameters are assigned by providing the estimates on the arcs and the leaf nodes, and propagating them according to a pre-defined inference algorithm.

The sub-phases within the “Verification of prediction models” phase are depicted in Figure 3. This phase aims to validate the prediction models, with respect to the structure and the individual parameters, before they are applied. A measurement plan with the necessary statistical power is
The PREDIQT method produces and applies a multi-layer model structure, called prediction models, which represent system relevant quality concepts (through “Quality Models”) and architectural design (through “Design Models”).

The Design Models represent the architectural design of the target system. The models include the parts and the detail level characterized (during the first sub-phase of the PREDIQT process) as a part of the objective of the analysis. Typically, Design Models include diagrams representing the process, the system structure, the dataflow and the rules for system use and operation. The Design Model diagrams are used to specify the target system and the changes whose effects on quality are to be predicted.

A Quality Model is a tree-like structure whose nodes (that is, quality notions at the different levels) are defined qualitatively and formally, with respect to the target system. The total quality of the system is decomposed into characteristics, sub-characteristics and quality indicators. Each of them is, by the Quality Model, defined in terms of a metric and an interpretation with respect to the target system. The definitions of the quality notions may for example be based on ISO 9126 product quality standard [5].

In addition, the prediction models comprise DVs, which are deduced from the Design Models and the Quality Models of the system under analysis. As explained above, the DVs model the dependencies of the architectural design with respect to the quality characteristic that the DV is dedicated to, in the form of multiple weighted and directed trees. The values and the dependencies modeled through the DVs are based on the quality characteristic definition provided by the Quality Model. A DV comprises two notions of parameters:

1) EI: Estimated degree of Impact between two nodes, and
2) QCF: estimated degree of Quality Characteristic Fulfillment.

Each arc pointing from the node being influenced is annotated by a quantitative value of EI, and each node is annotated by a quantitative value of QCF.

Figure 5 shows an excerpt of an example DV with fictitious values. In the case of the Encryption node of Figure 5, the QCF value expresses the goodness of encryption with respect to the quality characteristic in question, e.g., security. A QCF value on a DV expresses to what degree the node (representing system part, concern or similar) is realized so that it, within its own domain, fulfills the quality characteristic. The QCF value is based on the formal definition

![Application of prediction models](image_url)

Figure 4. Application of models – phase

![Excerpt of an example DV with fictitious values](image_url)

Figure 5. Excerpt of an example DV with fictitious values
of the quality characteristic (for the system under analysis), provided by the Quality Models. The EI value on an arc expresses the degree of impact of a child node (which the arc is directed to) on the parent node, or to what degree the parent node depends on the child node. The EI of an arc captures the impact of the child node on its parent node, with respect to the quality characteristic under consideration.

Input to the DV parameters may come in different forms (e.g., from domain expert judgements, experience factories, measurements, monitoring, logs, etc.), during the different phases of the PREDIQT method. Once the initial parameter values are assigned, the QCF value of each non-leaf node is recursively (starting from leaf nodes and moving upwards in the tree) propagated by multiplying the QCF and EI value for each immediate child and summing up these products for all the immediate children. This is referred to as the general DV propagation algorithm. For example, with respect to Data protection node in Figure 5 (denoting: DP: Data protection, E: Encryption, AT: Authentication, AAT: Authorization, and O:Other):

\[
QCF_{DP} = QCF_E \cdot EI_{(DP\rightarrow E)} + QCF_{AT} \cdot EI_{(DP\rightarrow AT)} + QCF_{AAT} \cdot EI_{(DP\rightarrow AAT)} + QCF_O \cdot EI_{(DP\rightarrow O)}
\]  

The DV-based approach constrains the QCF of each node to range between 0 and 1, representing minimal and maximal characteristic fulfillment (within the domain of what is represented by the node), respectively. This constraint is ensured through the normalized definition of the quality characteristic metric. The sum of EIs, each between 0 (no impact) and 1 (maximum impact), assigned to the arcs pointing to the immediate children must be 1 (for model completeness purpose). Moreover, all nodes having a common parent have to be orthogonal (independent). The dependent nodes are placed at different levels when structuring the tree, thus ensuring that the needed relations are shown at the same time as the tree structure is preserved. The overall concerns are covered by the nodes denoted Other, which are included in each set of nodes having a common parent, thus making the DV complete.

The general DV propagation algorithm, exemplified by Eq. 1, is legitimate since each quality characteristic DV is complete, the EIs are normalized and the nodes having a common parent are orthogonal due to the structure. A DV is complete if each node which is decomposed, has children nodes which are independent and which together fully represent the relevant impacts on the parent node, with respect to the quality characteristic that the DV is dedicated to.

The rationale for the orthogonality is that the resulting DV structure is tree-formed and easy for the domain experts to relate to. This significantly simplifies the parameterization and limits the number of estimates required, since the number of interactions between the nodes is minimized. Although the orthogonality requirement puts additional demands on the DV structuring, it has been shown to represent a significant advantage during the estimation.

Figure 6 provides an overview of the prediction models, expressed as a UML [6] class diagram. A prediction model is decomposed into a Design Model, a Quality Model and a DV. A Quality Model is a set of tree-like structures. Each tree is dedicated to a target system-relevant quality characteristic. Each quality characteristic may be decomposed into quality sub-characteristics, which in turn may be decomposed into a set of quality indicators. As indicated by the relationship of type aggregation, specific sub-characteristics and indicators can appear in several Quality Model trees dedicated to the different quality characteristics. Each element of a Quality Model is assigned a quantitative normalized metric and an interpretation (qualitative meaning of the element), both specific for the target system. A Design Model represents the relevant aspects of the system architecture, such as for example process, dataflow, structure and rules. A DV is a weighted dependency tree dedicated to a specific quality characteristic defined through the Quality Model. As indicated by the attributes of the Class Node, the nodes of a DV are assigned a name and a QCF (that is, value of the degree of fulfillment of the quality characteristic, with respect to what is represented by the node). As indicated by the Semantic dependency relationship, semantics of both the structure and the weights of a DV are given by the definitions of the quality characteristics, as specified in the Quality Model. A DV node may be based on a Design Model element, as indicated by the Based on dependency relationship. As indicated by the self-reference on the Class Node, one node may be decomposed into children nodes. Directed arcs express dependency with respect to quality characteristic by relating each parent node to its immediate children nodes, thus forming a tree structure. Each arc in a DV is assigned an EI, which is a normalized value of degree of dependence of a parent node, on the immediate child node. The values on the nodes and the arcs are referred to as parameter estimates. We distinguish between prior (or initial) and inferred parameter estimates. The former ones are, in the form of empirical input, provided on leaf nodes and all arcs, while the latter ones are deduced using the DV propagation model for PREDIQT exemplified above. For further details on the PREDIQT method, see [2], [3], [7], [1].

B. Uncertainty

The empirical input is always associated with a degree of uncertainty. Uncertainty is generally categorized into two different types: aleatory (due to inherent randomness of the system or variability of the usage profile) and epistemic (due to lack of knowledge or information about the system) [8]. The aleatory uncertainty is irreducible even by additional measurements. Aleatory uncertainty is typically represented by continuous probability distributions and forecasting is
based on stochastic models.

Epistemic uncertainty, on the other hand, is reducible, non-stochastic and of discrete nature. The epistemic uncertainty is therefore best suited for possibilistic uncertainty representations. For a detailed classification of the types and sources of imperfect information, along with a survey of methods for representing and reasoning with the imperfect information, see [9]. For a systematic literature review of the approaches for uncertainty handling in weighted dependency trees, see [10].

Prediction models, as opposed to for example weather forecasting models, are characterized by rather discrete, sudden, non-stochastic and less frequent changes. The weather forecasting models are of stochastic and continuous nature and the aleatory uncertainty is the dominating one (due to uncontrollable variabilities of many simultaneous factors). In majority of the system quality prediction models, aleatory uncertainty is negligible in terms of magnitude and impact, while the epistemic one is crucial. It is therefore the epistemic uncertainty we focus on when dealing with the parameters on the DVs.

C. Success criteria

Since expert judgements are a central source of input during the development of the prediction models, and also partially during the model verification, it is crucial that the formal representation of uncertainty is comprehensible to those who have in-depth system knowledge, but not necessarily a profound insight into the formal representation. The representation form of uncertainty estimates should make them easy for domain experts to provide and interpret.

Simultaneously, each individual parameter estimate should express the associated uncertainty so that it is as exact as possible. That is, the parameter and uncertainty values provided should be as fine grained as possible to provide, but without restricting comprehensibility. Thus, the right granularity of the uncertainty representation at the level of each parameter is needed.

Moreover, the input representation should facilitate combining both expert judgement-based and measurement-based input at the level of each parameter in a DV.

The DV propagation algorithm has a number of associated prerequisites (e.g., completeness, independence of the nodes which have a common parent, and ranges that the EI and QCF values can be expressed within). Together, they restrict the inference and the structure of the DVs so that the DVs become sound and comprehensible. When the parameters with the uncertainty representation are propagated within and across the DVs, the inference must still be well-defined and sound.

When applied on real-life cases, the uncertainty handling approach should propagate to practically useful predictions, in the sense that the approach can be applied on realistic DVs with limited effort and give valuable output.

Statistical and sensitivity analyses are currently performed in the DVs, during the Fitting of prediction models sub-phase and the Application of prediction models phase (of the PREDIQT process), respectively. Therefore, the uncertainty handling approach should also allow deduction of the central tendency measures such as mode, median, arithmetic mean, geometric mean, and variance.

Given the overall objective and context, the main success criteria for the uncertainty handling approach can, in a prioritized order, be summarized into:

1) The representation form of each parameter estimate and its uncertainty should be comprehensible for the domain experts involved in the development and use of the prediction models.

2) The representation form of each parameter estimate and its uncertainty should be as exact as possible, in terms of expressing both the parameter estimate and the associated uncertainty.
3) The approach should facilitate combining both expert judgement-based and measurement-based input.
4) The approach should correctly propagate the estimates and their uncertainty.
5) The approach should provide practically useful results.
6) The approach should allow statistical analysis.

III. OUR SOLUTION

This section presents an interval-based approach to representation and propagation of uncertainties on the DVs.

A. Uncertainty representation

All prior estimates (the terms “prior estimate” and “initial estimate” are used interchangeably, and regard the intervals directly assigned to the EIs and leaf node QCFs, i.e., the parameters based on the empirical input and assigned before the non-leaf node QCFs may be inferred) are expressed in terms of intervals within which the correct parameter values should lie. The width of the interval is proportional to the uncertainty of the domain experts or deduced from the standard deviation of the measurement-based input represented with probabilistic notions. In the latter case, the standard deviation indicates the accuracy of the measurements associated with each initially estimated parameter. Thus, the interval width may vary between the individual parameters. The representation of the estimates and their uncertainty is exemplified through an excerpt of a DV (with fictitious values) shown in Figure 7.

In addition to the quantifiable uncertainty associated with each initially estimated parameter, there may exist sources of uncertainty which are general for the context or the system itself, but to a lesser degree expressive or measurable. Examples include the presence of the aleatory uncertainty, the competence of the domain experts, data quality, statistical significance, etc. Such factors contribute to the overall uncertainty, but are (due to their weak expressiveness) not explicitly taken into account within the initially estimated EIs and the leaf node QCFs. Another reason for not accounting them within the intervals is because they are unavailable or may be biased at the individual parameter level. The domain experts may for example be subjective with respect to the above exemplified factors, or the tools for data acquisition may be incapable of providing the values regarding data quality, statistical significance, etc. Therefore, the context related uncertainty should, from an unpartial perspective (e.g., by a monitoring system or a panel, and based on a pre-defined rating), be expressed generally for all prior estimates.

Hence, we introduce the “confidence level” as a measure of the expected probability that the correct value lies within the interval assigned to a prior estimate. The confidence level is consistent and expresses the overall, uniform, context or system relevant certainty, in terms of a percentage. The confidence level regards the prior estimates only. The confidence level dictates the width of the intervals of the prior estimates, i.e., the certainty with which the exact value is within the interval assigned to a prior estimate. For example, a confidence level of 100% guarantees that the exact values lie within the intervals assigned to the prior estimates. Obviously, a requirement for increased confidence level will result in wider intervals of the prior estimates. In the case of Figure 7 the prior estimates are assigned with a confidence level of 90%. Let QCFs and EIs be represented by intervals of type $x$:

$$x = \{ x; \exists \} = \{ X \in [0;1] : x \leq X \leq \exists \}$$

where $x$ is the minimum estimated parameter value above which the exact value should (the term “should” is intentionally used in order to account for the confidence level of the prior estimates which is below 100%) lie, while $\exists$ is the maximum parameter value below which the exact value should lie. Both $x$ and $\exists$ are represented by real numbers. The interval $x$ of a prior estimate is assigned with the confidence level specified. Due to model completeness, EIs on the arcs pointing to the nodes with a common parent must satisfy:

$$\left( \sum_{i=1}^{I} \Sigma_{i} \right) \leq 1 \land \left( \sum_{i=1}^{I} \Pi_{i} \right) \geq 1$$

where $i$ denotes index of an arc, $I$ denotes the total number of the arcs with outspring from a common parent, and $\Sigma_{i}$ denotes the interval estimate for the EI on arc $i$. That is, there must exist at least one subset of scalars from within each one of the intervals (representing EIs on the arcs to nodes with a common parent), whose sum is equal to 1.

B. Uncertainty propagation

The initial estimates are provided in the form of intervals with respect to a confidence level, as specified above. The propagation of the initially estimated intervals on the non-leaf node QCFs is given by the existing DV propagation algorithm (exemplified by Eq. 1 in Section II), the interval arithmetics [11], [12], and the algorithms for non-linear optimization [13], [14]. The result of the propagation is in the form of intervals of QCF values on the non-leaf nodes.

The confidence level itself is not propagated but only used in the context of the assignment of the initial estimates. Therefore, the confidence level is only associated with the initial estimates and not the inferred ones (non-leaf node QCFs). The confidence level does however affect the width of the inferred parameters through the width of the initial estimates. That is, since a requirement for a higher confidence level implies wider intervals of the initial estimates, the propagation will, as specified below, result in wider intervals on the non-leaf node parameters.

The only two interval arithmetic operations needed for propagation in a DV are addition and multiplication. In case of two intervals denoted by $x$ and $y$ (of the form given by Eq. 2), addition and multiplication are defined as:
Confidence level on prior estimates: 90%

### Target system
- **QCF:**
  - min: 0.676
  - max: 0.96

### Target interfaces
- **QCF:**
  - min: 0.67
  - max: 0.92

### Authentication
- **QCF:**
  - min: ... 0.20
  - max: 0.40

### EI:
- min: 0.30
- max: 0.50

- min: 0.20
- max: 0.50

### Confidence level on prior estimates: 90%

#### Figure 7. Excerpt of a DV with intervals and confidence level

\[ x \circ y = [x \circ y, \bar{x} \circ \bar{y}] \]  

Where \( \circ \) denotes the operation symbol.

The optimization is necessary for obtaining the extreme values (the maximum and the minimum) of the interval of a parent node in the cases when several combinations (within the propagated intervals) give a sum of the EIs (on the arcs pointing to the immediate children) equal to 1. The scalar points (from within the intervals involved), which provide the extreme values, are identified by the non-linear optimization algorithms and then inferred to the parent node QCF in the form of an interval, according to the general DV propagation algorithm.

For a set of EI intervals whose total sum of the upper interval values is more than 1, there may be infinitely many combinations (the number of the combinations depends on the number of decimal digits, which the scalars from the intervals are represented with) of scalar points from within all the intervals, which together sum up to 1. Regardless of how many EIs (or nodes) there are, finding the min and the max values of the interval resulting from the propagation (sum of products of QCF and EI values associated with respectively the immediate children nodes and the arcs pointing to them) is a feasible optimization problem [14], [11]. Since the number of unknowns is equal to the number of equations involved, the only condition for the feasibility of the algorithm is the one expressed by Eq. 3.

Let \( qcf_j, qcf_i \in [0; 1] \) denote the interval limits of the QCFs on the immediate children and let \( e_i, \bar{e}_i \in [0; 1] \) denote the EIs on their respective interconnecting arcs. We propose the utility functions for the inferred min and max for the intervals of the parent node QCFs, which are given by respectively:

\[
\begin{align*}
\min \left\{ \sum_{i=1}^{\infty} qcf_j \cdot e_i \mid & e_i \leq e_i \leq \bar{e}_i \right\} & = 1 \} \quad (5) \\
\max \left\{ \sum_{i=1}^{\infty} qcf_j \cdot e_i \mid & e_i \leq e_i \leq \bar{e}_i \right\} & = 1 \} \quad (6)
\end{align*}
\]

Where \( I \) and \( i \) denote the same notions as in Eq. 3. The inference starts from the lowest internal nodes, and proceeds recursively upwards the tree.

The sensitivity of the inferred interval width of a dependent node, on the interval width of a dependee (node or arc), can be deduced by:

1. estimating the initial parameters and propagating them
2. obtaining the inferred interval width \( W \) of the selected dependee node
3. removing (or partially reducing) the interval width of the selected dependee \( D \)
4. obtaining the new inferred interval width \( W' \) of the dependee node
5. calculating the sensitivity \( S \) between the dependent node \( W \) and the dependee parameter \( D \), with respect to uncertainty.

We define the sensitivity measure \( S_{W,D} \) as:

\[
S_{W,D} = (1 - \frac{W'}{W})
\]  

In the context of predicting the quality characteristic, the natural choice of the dependent node will be the root node, which represents the quality characteristic that the DV is dedicated to, while the dependee will be a leaf node QCF or an EI. The QCF value on the root node will then represent the value of the quality characteristic of the system. The dependee is subject to the initial estimation. Therefore, the uncertainty of a dependee may be directly adjustable (for example, by reducing interval width due to added input). The sensitivity value can be obtained prior to selecting the candidate parameters for uncertainty reduction through added input. The obtained value of sensitivity (defined by Eq. 7) can in such a case be considered in relation to the effort needed for acquisition of the additional input. That is, higher sensitivity justifies putting more effort in acquiring additional input in order to decrease uncertainty of the dependee (and thus dependent) node.

### C. The uncertainty propagation in practice

Currently, we run the optimization in *Matlab*, where the utility function is, based on the DV propagation model exemplified by Eq. 1, defined as the sum of products of the QCF and EI intervals related to the immediate children nodes. The constraints of the utility function are:

- all QCF intervals involved,
- all EI intervals involved, and

...
\[ \sum_{i=1}^{\Sigma} e_i = 1 \] (where \( i \) denotes an arc, \( I \) is the total number of the arcs pointing to the nodes with the common parent under consideration, and \( e_i \) is a variable representing the EI value on the arc \( i \)). This constraint ensures the model completeness.

The minimum of the inferred interval is obtained from the utility function, while the maximum of the inferred interval is obtained by inverting the sign on the left hand side of the utility function and re-running the non-linear optimization algorithm. The Target interfaces and Target system nodes in Figure 7 are examples where such an algorithm had to be run in order to obtain the propagated intervals. In the case of Target interfaces, the utility function is specified in Matlab as:

```matlab
function f = objfun(x,y)
    f = [x(1)*x(2)+x(3)*x(4)+x(5)*x(6)]
end
```

Where \( x(1), x(3) \) and \( x(5) \) represent the EI values on the arcs pointing to the Authentication, Provisioning and Other nodes, respectively; while \( x(2), x(4) \) and \( x(6) \) represent the QCF values on the Authentication, Provisioning and Other nodes, respectively.

The related nonlinear inequality constraints representing the max and the min interval values of each respective variable specified above are defined in Matlab as:

\[
c = [-x(1) + 0.2; x(1) - 0.4; -x(2) + 0.7; x(2) - 0.9; -x(3) + 0.3; x(3) - 0.5; -x(4) + 0.6; x(4) - 0.8; -x(5) + 0.2; x(5) - 0.5; -x(6) + 0.8; x(6) - 1.0];
\]

The nonlinear equality constraint specifying that the sum of the EIs has to equal to 1, is defined in Matlab as:

\[
\text{ceq} = [x(1) + x(3) + x(5) - 1];
\]

The optimization algorithm is run by the following command in Matlab:

```matlab
x0 = [0,0,0,0,0,0]; % Make a starting guess at the solution
options = optimset('LargeScale','on');
[x, fval] = ... fmincon(@objfun,x0,[],[],[],[],options)
```

Providing the following result, where the values in the vector \( x \) specify the scalar points within the intervals \( x(1)-x(6) \), which yield the min value 0.67 of the utility function:

\[
x = 0.3000 0.7000 0.5000 0.6000 0.2000 0.8000
\]

\[
fval = 0.6700
\]

The max of the inferred interval is specified in Matlab by changing the sign of the above shown utility function to:

\[
E = -[x(1)*x(2)+x(3)*x(4)+x(5)*x(6)];
\]

and re-running the command from above. The output obtained is:

\[
x = 0.2000 0.9000 0.3000 0.8000 0.5000 1.0000
\]

\[
fval = 0.9200
\]

where the values in the vector \( x \) specify the scalar points within the intervals \( x(1)-x(6) \), which yield the max value of the utility function, namely 0.92.

The propagation results are displayed in Figure 7. We see that the scalar points of the optimization output are in accordance with the Eq. 5 and Eq. 6.

### D. Uncertainty analysis

Statistical analysis of measurements performed prior to model fitting and sensitivity analysis performed in relation to the application of prediction models, require a toolset for analysis of the data sets represented by intervals. The analysis of the central tendency measures of the interval-based estimates relies on the existing fully defined interval arithmetics and interval statistics [15]. Both can, in their existing well-established form, be directly applied in our context.

Apart from the summation and the multiplication presented by Eq. 4, the elementary interval arithmetic functions addition and multiplication (given two intervals denoted by \( x \) and \( y \), both of the form given by Eq. 2) include subtraction and division:

\[
x - y = [x - y, x - y]
\]

\[
x/y = [x/y, 1/y]/y, 1/y]
\]

As long as \( 0 \neq y \).

Arithmetic mean is given by:

\[
\left\{ \frac{1}{\Sigma} \Sigma_{i=1}^{\Sigma} x_i \right\}.
\]

For geometric mean, harmonic mean, weighted mean, and median, see [15]. Since no two data values are likely to be the same at infinite precision, mode does not generalize to a useful summary for data sets containing interval values. Instead, [15] proposes a substitute statistic, which identifies the places where most values in the data set overlap.

For problems with large sample sizes, computing variance of the interval data is an NP-hard problem. The algorithms for calculating variance presented in [15] solve the issue of infeasibility and make practical calculations of the needed interval statistics.

The standard deviation \( \sigma \) of an interval can be computed immediately from the variance \( \text{var} \) by taking its square root:

\[
\sigma = [\text{var}, \text{var}] = \sqrt{\text{var}} = \left[ \sqrt{\text{var}}, \sqrt{\text{var}} \right].
\]

Interval statistics for interquartile range, skewness, confidence intervals, regression fitting, maximum likelihood methods, as well as inferential interval statistics are thoroughly presented in [15]. In addition, [15] provides guidance regarding identification of outliers, trade-off between sample size and precision, handling of measurement uncertainty, handling of dependencies among the sources of uncertainty (correlation and covariance) and accounting for incertitude.

### IV. Why our solution is a good one

This section argues that the approach presented above fulfills the success criteria defined in Section II. Each one of the six criteria is considered in a dedicated subsection.

#### A. Criterion 1

The interval-based approach extends the DV parameters with the notions of interval widths and confidence level. Both interval width and confidence level are based on...
fairly intuitive and simple definitions. Hence, the approach should be relatively easy for the domain experts to use and understand, regardless of the degree of their formal background. The simplicity also makes it less prone to unstable over-fitting, as well as bias or inaccuracy of the estimations.

B. Criterion 2

The interval width can be selected at the individual prior estimate level, thus allowing adjustment of granularity of the uncertainty representation. The number of the decimal digits used in estimation and propagation is unlimited.

C. Criterion 3

The domain expert judgements are provided directly in terms of intervals with a confidence level. However the measurement-based input may come in terms of statistical notions.

Given that the measurement-based input is normally distributed, the interval end points can be calculated as [16]:

\[ \mu \pm t(1 - conf,n - 1)\sigma \sqrt{\frac{1}{n}} + 1 \]  

Eq. 12

where \( t(1 - conf,n - 1) \) is the two-tailed value of the Student’s t-distribution for the confidence level \( 1 - conf \) and \( n - 1 \) degrees of freedom, \( \mu \in [0;1] \) is the mean value, \( \sigma \) is the standard deviation of the measurements and \( n \) is the number of measurements. The “1” term inside the square root describes the spread of the measurement accuracy, while the “1/n” term describes the spread of the mean measurement accuracy. When \( n \) is high, there will be almost no uncertainty about the mean measurement accuracy, but the spread of the measurement accuracy may still be large. One can express both QCFs and EIs in this manner (for the relationship between the DV parameters and the measurements, see [2]), while requiring that Eq. 2 and Eq. 3 are satisfied. Alternatively, one can represent the QCF values in this manner, and the EI value of each related arc as a probability \( p \in [0;1] \), while enforcing \( \sum p = 1 \) for all nodes having a common parent. Thus, both kinds of input are transformable to intervals, which then can be propagated as defined in Section III and exemplified below.

D. Criterion 4

A consequence of the inequality and equality constraints is that all the inferred values will lie within the interval [0;1]. In addition, the normalized quality characteristic metric is defined so that all possible values always must lie within this interval. Moreover, the propagation algorithm calculates both the upper and the lower extreme values. As a result, the inferred prediction is an interval within which the exact (factual) value should lie. Two aspects are hindering from guaranteeing that the factual value lies within the inferred interval:

1) the confidence level with which the prior estimates are provided, and
2) the aleatory uncertainty, which unless accounted for in the confidence level, is not quantified within the intervals.

E. Criterion 5

The interval-based approach has also been tested by providing example values of estimates and their uncertainty on a real DV structure. The DV structure was originally used in a feasibility study of the PREDIQT method [2], performed on an extensive, real system. The uncertainty estimates were straightforward to provide by referring to the definition of the rating of the quality characteristic and expressing the estimates in terms of intervals. The interval width was mostly subject to observability of the parameter and existence of relevant historical input. The DV consisted of 38 leaf nodes, 9 internal nodes and 1 root node. The number of EIs on the arcs was 47. Thus, the number of initial (empirical input-based) estimates was 85, in this case. All initial estimates were expressed with intervals of reasonable and varying widths, within 90% confidence level. Once the initial estimates were in place, the propagation was quick and straightforward.

Table I summarizes the intervals applied. Each column lists the number of elements, the maximum interval width, the minimum interval width, the average interval width and the standard deviation of the interval width. The first two columns present the values for the initial estimates of both the leaf node QCFs and all the EIs, respectively. The third column presents the values for the initial estimates of both the leaf node QCFs and all the EIs. The last column presents the results for the propagated QCFs (on the internal nodes and the root node). The resulting interval width of the root node QCF was 0.032. Given the attempt to provide as realistic and as variable interval widths of the initial estimates as possible, the example should be an indication of the expected findings in similar settings. Note that, while the interval widths reflect the expected uncertainty, all values assigned to parameter estimates are fictitious, due to their confidentiality. The obtained root node interval width can be considered as a promising result, since the predictions are still likely to be associated with limited and acceptable uncertainty.

To test impact of uncertainty elimination on one leaf node (a child node of the root node) on the above presented DV, its QCF was changed from [0.90;0.95] to [0.925;0.925]. The
resulting interval width of the root node QCF became 0.0295 and the value of Eq. 7 became 0.081. Note that these values, too, are based on fictitious input, due to confidentiality of the actual initial estimates.

In a real-life setting, not all the estimates will be expressed with uncertainty, since some of the nodes have no impact or no uncertainty. The evaluation of the above mentioned feasibility study showed that the uncertainty of the input and the deviations between the PREDIQT-based and the empirical predictions are relatively low. The experience from the feasibility study is that the interval widths would be quite small. Most of the nodes of the DVs were placed on the second or the third level, which considerably limits the vertical propagation of uncertainties.

Reducing the confidence level and conducting further model fitting (through additional input) are the obvious counter-measures when the inferred values are too uncertain. The candidate parameters for reduction of uncertainty can be identified by using the sensitivity measure proposed in Section III in relation to the effort needed for the uncertainty reduction in question. Alternatively, a sensitivity analysis supported by charts and central tendency measures can be pursued in order to observe the impact that a reduction of uncertainty of the individual estimates would have on (the root node of) the DV.

F. Criterion 6

The analysis of the central tendency measures of the interval-based estimates relies on the existing fully defined interval arithmetics and interval statistics [15]. Both can, in their existing well-established form, be directly applied in our context. For arithmetic mean, geometric mean, harmonic mean, weighted mean, median, standard deviation and variance, see [15]. In addition, [15] provides guidance regarding identification of outliers, trade-off between sample size and precision, handling of measurement uncertainty, handling of dependencies among the sources of uncertainty (correlation and covariance) and accounting for incertitude.

V. WHY OTHER APPROACHES ARE NOT BETTER IN THIS CONTEXT

A ratio scale is a measurement scale in which a certain distance along the scale means the same thing no matter where on the scale we are, and where “0” on the scale represents the absence of the thing being measured. Statistical analysis and arithmetics are supported for the ratio scale. The ratio scale is in fact used in Section II. We may for example introduce uncertainty representation by defining fixed increments on the scale from 0 to 1, and relating their meaning to the quality characteristic rating. The input would have to be expressed in the form of the increments defined, and the uncertainty would per definition range half the way to the neighboring increments. Obviously, this is a special case of the interval approach where the increments and their granularity are frozen at the model (and not parameter) level. By using a ratio scale in the PREDIQT context, the schema of the increments would have to apply for the entire model (in order for the uncertainty propagation to be meaningful) rather than being adjustable at the parameter level. As a result, the schema of the increments may be either too coarse grained or too fine grained in the context of certain parameters. The variation of uncertainty between parameters would not be supported, thus violating criterion 2 from Section II.

The Dempster-Shafer structures [15] offer a way of representing uncertainty quantified by mass distribution functions. A mechanism for aggregation of such representation stored in distributed relational databases, is proposed by [17]. The Dempster-Shafer approach characterizes uncertainties as intervals with degrees of certainty (that is, sets of values with weights which add up to 1). It can be seen as a generalization of both interval analysis and probability theory. Weights of evidence are put on a collection of intervals and the structures may overlap. Implementing the Dempster-Shafer theory in our context would involve solving two issues: 1) sorting the uncertainties in the empirical input into a priori independent items of evidence, and 2) carrying out Dempster’s rule computationally. The former one leads to a structure involving input elements that bear on different but related concerns. This structure can be used to make computations based on Dempster’s rule feasible. Our solution is a special case of the Dempster-Shafer approach, where the intervals of the prior estimates have a general confidence level, and the structure of the DV allows for a linear propagation. The additional expressiveness that the Dempster-Shafer structures offer is not needed in our context, since the certainty is highly unlikely to vary across the fractions of the intervals. In fact, such a mechanism will, due to its complex representation on subsets of the state space, in the PREDIQT context only compromise the comprehensibility of the uncertainty representation and therefore the correctness of the input.

Bayesian networks (BNs) [18], [19] may represent both model uncertainty and parameter uncertainty. A BN is a directed acyclic graph in which each node has an associated probability distribution. Observation of known variables (nodes) allows inferring the probability of others, using probability calculus and Bayes theorem throughout the model (propagation). BNs can represent and propagate both continuous and discrete uncertainty distributions. BNs in their general form are however demanding to parameterize and interpret the parameters of, which violates our first criterion. This issue has been addressed by [20] where an analytical method for transforming the DVs to Bayesian networks is presented. It also shows that DVs, although easier to relate to in practice, are compatible with BNs. It is possible to generalize this transformation so that our interval-based approach is transformed to a BN before
a further BN-based analysis may be conducted. Such an extension would introduce several states on the BN nodes, and assign probabilities to each of them. In that manner, the extension would resemble the Dempster-Shafer structures. BNs in their general form do not score sufficiently on our criteria 1 and 5.

Fuzzy logic provides a simple way to draw definite conclusions from vague, ambiguous or imprecise information, and allows for partial membership in a set. It allows modeling complex systems using higher levels of abstraction originating from the analyst’s knowledge and experience [21]. A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership function, which assigns to each object a grade of membership ranging between zero and one [22]. Using the fuzzy membership functions, a parameter in a model can be represented as a crisp number, a crisp interval, a fuzzy number or a fuzzy interval. In the fuzzy logic approach the algebraic operations are easy and straightforward, as argued and elaborated by [23]. The interval-based approach is a special case of the fuzzy approach, where only the crisp intervals are used as membership functions. The additional expressiveness that the overall types of the membership intervals are used as membership functions. The additional expressiveness that the overall types of the membership functions offer is in fact not needed in the PREDIQT context, since the increased complexity of the estimate representation would not contribute to the accuracy of the parameter values, but rather introduce misinterpretations and incorrectnesses in the input provision. The interpretation of the membership distributions and their correspondence to the practical settings in the PREDIQT context would be demanding.

Subjective logic [24] is a framework for reasoning, which consists of a belief model called opinion and set of operations for combining opinions. A single opinion \( \pi \) is uniquely described as a point \( \{b,d,i\} \) in an “Opinion Triangle”, where \( b \), \( d \) and \( i \) designate belief, disbelief and ignorance, respectively. For each opinion, the three notions sum up to unity. The operations formally defined include: conjunction, disjunction, negation, consensus, recommendation and ordering. The subjective logic is suited for the domain expert judgements, but how the measurement-based input can be related to the concepts of the subjective logic, needs to be defined. Thus, applying the subjective logic in the PREDIQT context would increase the fulfillment of our second criterion beyond the needs, while degrading fulfillment of the third criterion.

Uncertainty representation in software development effort-estimation [25], [26] is most comparable to ours. However, they do not have as a strict criterion of propagation, and can therefore introduce different notions to the uncertainty representation.

It should be pointed out that the interval propagation based on the extreme values suffers from the so-called overestimation effect, also known as the dependency problem. The dependency problem is due to the memoryless nature of interval arithmetic in cases when a parameter occurs multiple times in an arithmetic expression, since each occurrence of an interval variable in an expression is treated independently. Since multiple occurrence of interval parameters cannot always be avoided, the dependency problem may cause crucial overestimation of the actual range of an evaluated function. A way to approach this issue is to use interval splitting [27], where the input parameter intervals are subdivided and the arithmetics are preformed on the subintervals. The final results are then obtained by computing the minimum of all lower bounds and the maximum of all upper bounds of the intermediate results. Skelboe [28] has shown that the results obtained from the interval splitting converge to the actual range if the width of the subintervals approaches zero. Our solution does not use interval splitting, as it would significantly increase complexity of the entire approach, thus compromising our first criterion.

The epistemic uncertainty is the crucial one in the context of PREDIQT and therefore given the main attention in our context. Being of a discrete nature, the epistemic uncertainty should, as argued in Section II, be handled by a purely possibilistic approach. The approaches mentioned in the remainder of this section focus to a high degree on the stochastic uncertainties, which makes them less suited in the PREDIQT context.

The ISO approach to handling measurement uncertainty [29] uses a probabilistic representation with normal distribution, and treats both aleatory and epistemic uncertainty equally. Such an approach however does not explicitly account for the notion of ignorance about the estimates, thus failing to intuitively express it.

A simulation mechanism, which takes into account both aleatory and epistemic uncertainty in an interval-based approach, is proposed by [30]. It concentrates on stochastic simulations as input for the interval estimates when significant uncertainties exist. Moreover, [15] proposes considering a hybrid approach comprising both probabilistic and interval representation, in order to account for both aleatory and epistemic uncertainty. Neither of these two approaches would in the the context of PREDIQT increase fulfillment of our success criteria. In fact, the systematic sources of uncertainty would not be represented more accurately, while comprehensibility would degrade.

A hybrid Monte Carlo and possibilistic method for representation and propagation of aleatory and epistemic uncertainty is presented by [31]. The method is applied for predicting the time to failure of a randomly degrading component, and illustrated by a case study. The hybrid representation captures the aleatory variability and epistemic imprecision of a random fuzzy interval in a parameterized way through \( \alpha \)-cuts and displays extreme pairs of the upper and lower cumulative distributions. The Monte Carlo and the possibilistic representations are jointly propagated. The gap between the upper and the lower cumulative distributions.
represents the imprecision due to epistemic variables. The possibility distributions are aggregated according to the so-called Ferson method. The interpretation of the results in the form of limiting cumulative distributions requires the introduction of a degree of confidence directly connected with the confidence on the value of epistemic parameters. Compared to this approach, our solution is more comprehensible but less suited for handling the aleatory uncertainty. However, given our criteria, the former aspect outranges the latter one.

The approaches to uncertainty handling in other domains, such as weather forecasting [32], electricity demand forecasting [33], correlations between wind power and meteorological conditions [34], power system planning [35] and supply industry [36] are mainly based on probabilistic representations and stochastic simulations. They focus mainly on the aleatory uncertainty, which in the PREDIQT context is of secondary relevance.

Hence, given the criteria presented in Section II, the interval-based approach prevails as the most appropriate one in the PREDIQT context.

VI. LESSONS LEARNED

This section provides practical guidelines for obtaining the empirical input and reducing the uncertainty of estimates. Firstly, we elaborate on how the maximum acceptable uncertainty objective, that is, an acceptable threshold for uncertainty, can be characterized. Secondly, guidelines for obtaining the prior estimates are summarized. Lastly, means and measures for reducing uncertainty are outlined. The guidelines are based on the authors’ experiences from industrial trials of PREDIQT on real-life systems [2], [3]. As such, the guidelines are not exhaustive but may serve as an aid towards a more structured process for uncertainty handling.

A. Characterizing the maximum acceptable uncertainty objective

The maximum acceptable uncertainty objective can to a certain degree be expressed through the confidence level, which is a measure of the expected probability that the correct value lies within the interval assigned to a prior estimate. However, the confidence level is merely concerned with the prior estimates although it indirectly influences the inferred DV estimates. Therefore, if the interval width of a specific non-leaf node is of major concern, it has to be specified directly as a part of the maximum acceptable uncertainty objective, by the stakeholders. Note however that there is still a correlation between the confidence level of the prior estimates and the inferred QCFs, that is, the uncertainty of an inferred QCF is expressed through both width of its interval, as well as the confidence level of the prior estimates which influence the QCF value of the non-leaf node in question.

Consequently, in the case of the prior estimates, the maximum acceptable uncertainty objective can be expressed through the confidence level, and will in that case give interval widths depending on the quality of the empirical input. In the case of the non-leaf node QCF values, the maximum acceptable uncertainty objective should be expressed in terms of both the confidence level of the prior estimates and the interval width of the parameters in question.

B. Obtaining the prior estimates

We recommend obtaining the leaf node QCFs of a subtree prior to obtaining the related EIs. The rationale for this is to fully understand the semantics of the nodes, through reasoning about their QCFs first. In estimating a QCF, two steps have to be undergone:

1) interpretation of the node in question – its contents, scope, rationale and relationship with the Design Models, and
2) identification of the relevant metrics from the Quality Model of the quality characteristic that the DV is addressing, as well as evaluation of the metrics identified.

QCF is the degree of fulfillment of a quality characteristic, with respect to the node in question. Normalization of the values of the above mentioned metrics and their degree of influence, results in a QCF value with an uncertainty interval assigned with respect to the pre-defined confidence level. Alternatively, rating of the characteristic (as formally defined by its Quality Model at the root node level) can be estimated directly with respect to the node under consideration, in order to provide its QCF value.

In estimating an EI, two steps have to be undergone:

1) interpretation of the two nodes in question, and
2) determination of the degree of impact of the child node on the parent node, with respect to the quality characteristic (defined by the Quality Model) that the DV is addressing. The value is assigned relative to the overall EIs related to the same parent node, and with a consistent unit of measure, prior to being normalized (in order to fulfill Eq. 2). The normalized EIs on the arcs from the same parent node have to fulfill Eq. 3, due to the requirement of model completeness.

Hence, EI is the dependency of the parent node on the child node. Estimation of the EI values between a parent node and its immediate children, results in intervals with respect to the pre-defined confidence level.

1) Questions to ask domain experts: The first step in the interaction between the analyst and the domain experts is to clarify the meaning of the node(s) under consideration, their respective rationales and the possible traces to the Design Models. Secondly, the analyst has to facilitate the estimation by reminding the domain experts of the quality characteristic definition – both the qualitative and the formal part of it.

When estimating a QCF the following question is posed: “To what degree is the quality characteristic fulfilled, given
the contents and the scope of the node?” The definition of the quality characteristic (interpretation and the metric) should be recalled.

When estimating an EI the following question is posed:

“To what degree does the child node impact the parent node, or how dependent is the parent node on child node, with respect to the quality characteristic that the DV is dedicated to?” The definition of the quality characteristic provided by its Quality Model, should be recalled and the estimate is provided relative to the impact of the overall children nodes of this parent. Alternatively, an impact value is assigned using the same unit of measure on all arcs of the sub-tree, and normalized thereafter.

Once one of the above specified questions is posed, depending on the kind of the DV parameter, the domain expert panel is asked to provide the estimate with an interval so that the correct value is within the interval with a probability given by the confidence level. For EIs on the nodes having a common parent, it has to be validated that Eq. 3 is fulfilled.

Furthermore, discussions among the domain experts should be encouraged and all the estimates should be requested during a limited period of time (in the form of tightly scheduled meetings), in order to ensure relative consistency of the estimates. Additionally, for the purpose of the relative consistency of the estimates, the domain expert group should be diverse and representative. There should be continuity in a fraction of the group, and limited turnover between the different meetings. The turnover may however be advantageous for the purpose of the expertise at the different stages of the process of PREDIQT.

Apart from the domain expert judgements, the estimates are also based on measurements. When obtaining measurement-based input, we rely on a measurement plan which relates the practical measurements to the DV parameters and the quality notions. The Goal/Question/Metric [37], [38], [39] approach and the ISO 9126 product quality standard [5] are particularly useful for deducing such relationships. The overall literature on software measurement is extensive [40], [41], [42], [43] and provides useful guidelines for obtaining the measurement-based input.

2) Use of Quality Models: Quality Models are used as a reference in estimation of each prior estimate. The Quality Models assist the domain experts in selecting and evaluating the relevant metrics. The metrics also provide a basis for defining the measurements. The decomposition of the Quality Models is however only based on indicators whose overlaps and degrees of impact on the characteristic may vary. The composition of the degrees of relevance of the various indicators is therefore left to the analyst or the domain experts to determine in the case of each estimate.

3) Use of Design Models: Design Models specify the target of the analysis in terms of scope and the contents. The Design Models serve as a reference for common understanding of the system, prior to and during the estimation.

In addition, the appropriate parts of the DVs are traced to the elements of the Design Models, making the contents and the scope of the DV elements more explicit.

4) Determining the uncertainty value: The uncertainty value of a prior estimate is determined through the interval width based on the pre-defined confidence level. In the case of the measurement-based input, the transformation to an interval is presented in Section IV. In that context, confidence level will reflect the data quality (that is, the validity of the measurements).

In the case of the domain expert judgements, however, the interval width is agreed upon by the domain expert panel, while the validity of the panel (that is, mainly representativeness and statistical significance of its composition) is reflected through the confidence level. This ensures consistency of the confidence level in the case of the expert judgements.

In order to also ensure a consistent confidence level in the case of the measurements (where data quality may vary among the measurements related to the different DV estimates), the confidence level can be kept consistent by compensating for the possible variations through the interval width. The relationship between the confidence level and the interval width is however not formalized beyond the fact that the confidence level denotes the probability of the correct value of the estimate lying within the interval.

C. Reducing uncertainty

Since we only consider the epistemic uncertainty, there exist means and measures that can be used to reduce it. The difficulty of reducing uncertainty lies in addressing the unknown sources of uncertainty, which are not explicitly expressed in the estimates. This is however not a major issue in the case of the epistemic uncertainty.

The rest of this section provides guidelines for uncertainty reduction from the different perspectives: process, model granularity, measurement-based input and expert judgements.

1) Process related measures: Among the process related measures are:

- access to the necessary documentation
- access to measurement facilities
- involvement and composition of the domain expert panel in all phases of the process
- common understanding of the modeling language and the terminology
- sufficient understanding of the PREDIQT method, particularly the models (by the domain experts and the analyst)
- use of known notations and modeling frameworks
- use of standards where appropriate
- user-friendly tool support with structured process guidance
- reuse of the existing models where appropriate.
The rationale for these measures is a more structured process which provides the sufficient input and leads towards a harmonized understanding of the models. For a more detailed elaboration of the process related measures, see [3].

2) **Granularity of the models:** Quality of a model-based prediction is, once the prediction models are developed, subject to the granularity of the models. Increased granularity of all prediction models will potentially decrease uncertainty.

In case of Quality Models, finer granularity can be achieved by further formalization and decomposition of the quality characteristics. In case of Design Models, the more detailed diagrams and traces among them are a means of addressing granularity.

In the case of the DVs, additional traceability of the actions and rationale, as well as increased traceability between DV model elements and the Design Models, will increase the precision and reduce uncertainty. Particularly, the following should be documented during the DV development:

- assumptions
- rationale
- relationships or traces to the Design Models
- traces to the relevant quality indicators and contributions of the relevant quality indicators
- interpretations of the prior estimates
- the supporting information sources (documents, measurement, domain experts) used during the development of DV structure and estimation of the parameters.

3) **Quality of measurement data:** Increase of validity of the measurement data will directly increase the confidence level. This may be achieved by increasing the statistical significance of the measurements in terms of relevance and amount of the measurement-based input.

4) **Quality of expert judgements:** The expert judgements are subject to understandability and granularity of the prediction models, composition of the expert panel (representativeness, number of participants, their background and interests), setup and approach to the estimate acquisition. Discussion should be facilitated and possible interest conflicts should be addressed.

VII. **CONCLUSION AND FUTURE WORK**

Our earlier research indicates the feasibility of the PREDIQT method for model-based prediction of impacts of architectural design changes on system quality. The PREDIQT method produces and applies a multi-layer model structure, called prediction models, which represent system design, system quality and the interrelationship between the two. A central part of the prediction models are the DVs, which are parameterized in terms of fulfillment of quality characteristics and impacts among the elements, with respect to the quality characteristics. The DV elements are representations of architectural design or quality, which are partially traceable to the underlying Design Models and Quality Models. Due to its empirical nature, input into the DVs is associated with uncertainty. By handling the uncertainty in the DVs, quality of the prediction models and accuracy of the predictions are made explicit, thus indicating which changes are predictable and whether further model fitting is needed.

Based on a set of criteria identified with respect to the PREDIQT method, we have proposed and evaluated an approach to uncertainty handling in the DVs. The approach relies on intervals with a confidence level, and covers representation, propagation and analysis of the DV parameters and their respective uncertainty estimates. The interval-based approach allows comprehensible representation of uncertainty on all kinds of parameters, with the needed accuracy. Estimation, propagation and analysis in the interval-based approach are scalable and efficient. The interval arithmetics, the algorithms for non-linear optimization, and the statistical analysis of intervals are already fully established and can be applied in the PREDIQT context in their existing forms. The evaluation argues for the correctness and practical usefulness of our approach, as well as its outranging appropriateness relative to the alternative uncertainty handling approaches.

The approach is entirely compliant with the existing version of the PREDIQT method. Based on empirical trials of PREDIQT, we have provided guidelines for use of the uncertainty handling approach in practice. The guidelines address the ways of obtaining the empirical estimates as well as the means and measures for reducing uncertainty of the estimates.

Further work will address analysis of the prediction accuracy, that is the deviation between the predicted and the actual quality characteristic values. The notions of magnitude of average deviation AD [2], balanced relative error BRE [44] and hit rate (i.e., the percentage of the correct values lying within the predicted intervals) can be used as measures of prediction accuracy. For an accurate prediction model, the hit rate should be consistent with the confidence level. The BRE allows analysis of bias and precision (see Figure 8) of the predictions. Thus, systematic and random variance of the prediction accuracy can be distinguished in a meta analysis of our uncertainty handling approach. The prospects of further work also include additional empirical evaluations of practical usefulness and accuracy of the approach. Moreover, identifying and categorizing the variables that impact the uncertainty of the estimates, is important for improving uncertainty management.
ACKNOWLEDGMENT
This work has been conducted as a part of the DIGIT (180052/S10) project funded by the Research Council of Norway, as well as a part of the NESSoS network of excellence funded by the European Commission within the 7th Framework Programme.

REFERENCES


Chapter 11

Paper 3: Uncertainty Handling in Weighted Dependency Trees – A Systematic Literature Review
# Table of Contents

INTRODUCTION .................................................................................................................................................. 1

WEIGHTED DEPENDENCY TREES AND UNCERTAINTY .............................................................................. 2
  Weighted dependency trees............................................................................................................................... 2
  Uncertainty........................................................................................................................................................ 3

THE RESEARCH METHOD ................................................................................................................................. 4
  Overall process .................................................................................................................................................. 4
  Literature review ............................................................................................................................................... 4
  The mapping study.......................................................................................................................................... 4
  The systematic literature review .................................................................................................................... 5

THE EVALUATION CRITERIA ............................................................................................................................ 6

THE HIGH-LEVEL EVALUATION ......................................................................................................................... 8
  Bayesian networks............................................................................................................................................ 8
  Bootstrapping.................................................................................................................................................. 9
  ISO guide to handling measurement uncertainty ........................................................................................ 9
  Delta method.................................................................................................................................................. 9
  Qualitative probabilistic networks............................................................................................................... 9
  Credal networks............................................................................................................................................. 10
  Intervals ....................................................................................................................................................... 10
  Subjective logic............................................................................................................................................ 11
  Fuzzy logic................................................................................................................................................... 11
  Dempster-Shafer structures......................................................................................................................... 12
  Evidence theory.......................................................................................................................................... 13
Uncertainty Handling in Weighted Dependency Trees
A Systematic Literature Review

Aida Omerovic\textsuperscript{A,B}, Amela Karahasanovic\textsuperscript{A,B} and Ketil Stølen\textsuperscript{A,B}

\textit{A: SINTEF ICT, Norway}
\textit{B: Department of Informatics, University of Oslo, Norway}

\textbf{ABSTRACT}
Weighted dependency trees (WDTs) are used in a multitude of approaches to system analysis, such as fault tree analysis or event tree analysis. In fact, any acyclic graph can be transformed to a WDT. Important decisions are often based on WDT analysis. Common for all WDT-based approaches is the inherent uncertainty due to lack or inaccuracy of the input data. In order to indicate credibility of such WDT analysis, uncertainty handling is essential. There is however, to our knowledge, no comprehensive evaluation of the uncertainty handling approaches in the context of the WDTs. This paper aims to rectify this. We concentrate on approaches applicable for epistemic uncertainty related to empirical input. The existing and the potentially useful approaches are identified through a systematic literature review. The approaches are then outlined and evaluated at a high-level, before a restricted set undergoes a more detailed evaluation based on a set of pre-defined evaluation criteria. We argue that the epistemic uncertainty is better suited for possibilistic uncertainty representations than the probabilistic ones. The results indicate that precision, expressiveness, predictive accuracy, scalability on real-life systems, and comprehensibility are among the properties which differentiate the approaches. The selection of a preferred approach should depend on the degree of need for certain properties relative to others, given the context. The right trade off is particularly important when the input is based on both expert judgments and measurements. The paper may serve as a roadmap for examining the uncertainty handling approaches, or as a resource for identifying the adequate one.

\textbf{INTRODUCTION}
WDTs are widely used in approaches to system analysis. WDTs are used as a means to understand some artifact, e.g. a system and to make informed decisions regarding its future behaviour. Examples include:
- fault tree analysis (IEC, 2006) – a risk analysis technique based on so-called fault trees;
- event tree analysis (IEC, 1995) – a modeling technique for representing consequences of an event and the probabilities of the respective consequences;
- attack trees (Schneier, 1999) – a notation similar to fault tree analysis for modeling potential attacks on a system with an attack goal as the top node and different ways of achieving that goal as leaf nodes; and
- dependency views in system quality prediction (Omerovic, et al., 2010).

Common for all approaches supported by WDTs is the inherent uncertainty due to lack or inaccuracy of input data. The input data originates from two kinds of sources: expert-judgments-based and measurement-based data acquisition (such as logs, monitoring, experience factories and other measurements). Uncertainty regarding input data (due to for example lack of knowledge, as well as variability or poor quality of the measurement-based data) can lead to errors in relation to both modeling and analysis.
Apart from lack or inaccuracy of input, another source of the uncertainty may be the variability of the system or its usage. However, in this paper we restrict our attention to artifacts, e.g. systems, whose behavior is deterministic, for which the former type of uncertainty is the only prevailing one. Consequently, we only focus on deterministic uncertainty handling.

Important decisions are made during and after a WDT-based analysis. The ability of explicitly expressing uncertainty in the input and its implications on the rest of the model is crucial due to the consequences of the input uncertainty in the decision making. The uncertain input may have extensive impact on the rest of the WDT-based model. In worst case, decisions involving unacceptable risk may be taken, without the awareness of the analyst. There are numerous approaches to uncertainty handling. Each of them is motivated by a special need not covered by the competing ones. Their properties such as complexity, expressiveness, propagation and generally the practical applicability vary to a high extent. It is however unclear to what degree the existing approaches are suitable for being adopted for use on WDTs. We have not been able to find a comprehensive evaluation of the uncertainty handling approaches in the context of the WDTs.

The overall objective of this paper is therefore to identify and evaluate the existing and the potentially useful approaches that could, individually or in combination, be adopted for handling the uncertainty in WDTs. This includes:

1. identification of the established approaches for uncertainty representation, based on a literature review;
2. a high-level evaluation of the identified approaches with respect to the main needs; and
3. a low-level evaluation of a restricted set of the approaches which are assessed as potentially useful during the high-level evaluation.

The paper is structured as follows: “Weighted dependency trees and uncertainty” provides background on the notion of WDT and the issue of uncertainty. The research method is then presented in a section titled “The research method”. Next, a section titled “The evaluation criteria” presents the evaluation criteria, with respect to the practical acceptability of the uncertainty handling approaches. The section titled “The high-level evaluation” reports on the high-level evaluation of the approaches identified, while the section titled “The low-level evaluation” presents the results of a more detailed evaluation of the selected approaches. Finally, we discuss our findings and provide our main conclusions in the section titled “Conclusion”. Furthermore, there are three appendices providing background on the literature review, the detailed deduction of the evaluation criteria, and threats to validity and reliability, respectively.

WEIGHTED DEPENDENCY TREES AND UNCERTAINTY

This section introduces the two central notions: dependency trees and uncertainty. Firstly, a general definition of a WDT is proposed. Secondly, the major sources of and the two overall categories of uncertainty, are introduced.

Weighted dependency trees

Various tree structures, such as for example event trees, attack trees or dependency trees, are used in system analysis. We generalize the notion of the tree-based representations with quantitative parameters such as the already mentioned fault trees, event trees, attack trees or dependency views, by proposing the concept of a WDT. A WDT, as illustrated by Figure 1, is a finite tree in which:

- each node may hold one or more quantitative values or a function,
- each arc may hold one or more a quantitative values or a function, and
- each inferred non-leaf node is a tree-specific function of its immediate children and their connecting arcs.
The tree-specific function is basically what distinguishes one kind of WDT from another. For example, in case of the WDT on Figure 1, the value $q_{11}$ is a function of $q_{21}$, $q_{22}$, $e_{11}$ and $e_{12}$. Note that not all tree-specific functions require assignments on the arcs. Furthermore, we distinguish between prior estimates and the inferred or propagated estimates. The former ones are based on the empirical input which may originate from expert judgments or measurements, while the latter one is inferred based on tree-specific functions.

We distinguish between two variations of WDTs:
- Restricted WDT (RWDT): the same node may occur only once, and
- General WDT (GWDT): the same node may occur more than once.

Any directed acyclic graph (DAG) may be represented as GWDT, by duplicating the nodes in the DAG.

**Uncertainty**

The empirical input is always associated with a degree of uncertainty. Uncertainty is generally categorized into two different types: aleatory (due to inherent randomness of the system or variability of the usage profile) and epistemic (due to lack of knowledge or information about the system) (Kiureghian & Ditlevsen, 2009).

The aleatory uncertainty is a property of the system associated with variability. It is irreducible even by additional measurements. Aleatory uncertainty is typically represented by continuous probability distributions and forecasting is based on stochastic models. Epistemic uncertainty, on the other hand, is reducible, non-stochastic and of a discrete nature. It is considered as uncertainty which may be originating from a range of causes that defy pure probabilistic modeling. Examples of such causes are lack or inaccuracy of input, which impedes the specification of a unique probabilistic model. Epistemic quantities have fixed values which may be unknown or poorly known. For example size or cost of an existing system are values which are fixed and existing, but may be difficult to reveal or deduce.

As opposed to for example weather forecasting models which are of stochastic and continuous nature and where the aleatory uncertainty is the dominating one (due to uncontrollable variabilities of many simultaneous factors), models of deterministic artifacts are characterized by rather discrete, sudden, non-stochastic and less frequent changes. As a result, the aleatory uncertainty is, in the deterministic artifacts,

![Figure 1: A weighted dependency tree](image-url)
negligible in terms of magnitude and impact, while the epistemic one is crucial. It is therefore the epistemic uncertainty we focus on when dealing with the deterministic artifacts.

**THE RESEARCH METHOD**

This section provides an overview of the research method. The overall process of research is outlined in the next subsection, which is followed by a subsection presenting the main steps within the literature review.

**Overall process**

The overall process is divided into the two stages depicted in Figure 2. Firstly, a literature review is conducted. The literature review consists of a mapping study (that is, a search in several digital resources, based on pre-defined keywords) and a systematic literature review. The relevant publications related to the state of the art with respect to the uncertainty handling approaches, are identified. The main contents of the relevant publications are extracted, indicating the potentially useful approaches for uncertainty handling and their main properties with respect to the objective specified. The systematic literature review provides a preliminary evaluation and an initial list of the evaluation criteria. Based on the results of the systematic literature review and the experienced needs for the evaluation, the evaluation criteria are iteratively refined through multiple discussions among the researchers. The deduction of the criteria is partially guided by an existing taxonomy for system acceptability.

![Figure 2: An overview of the process undergone](image)

Secondly, an evaluation of the approaches identified and analyzed throughout the systematic literature review is conducted. The evaluation consists of two stages: a high-level evaluation and a low-level evaluation, respectively. The high-level evaluation classifies the approaches identified and summarizes their main properties. The classification is organized according to the kind of the approach (e.g., bayesian, fuzzy, interval-based, hybrid, etc.). Moreover, each classified kind of the approach is assessed as to whether it should be further evaluated as a part of low-level evaluation, that is in the context of the WDTs and with respect to the refined evaluation criteria. In the low-level evaluation each approach is instantiated on a WDT, in order to test its applicability.

**Literature review**

This subsection presents an overview of the literature review. It consists of a mapping study performed in order to gain an overview of the main topics and possible keywords, and a systematic literature review. The details regarding the process of both the mapping study and the systematic literature review are provided in Appendix A.

**The mapping study**

The mapping study involved a search in several digital resources, based on pre-defined keywords related to uncertainty and prediction. About fifty publications from mathematics, statistics and computer science which presented either single or combined approaches for uncertainty handling, were extracted. The ones addressing customization of identified approaches for the different domains and the ones focusing on mainly aleatory uncertainty, were excluded, thus reducing the search results to about 33 publications.
Moreover, uncertainty and inaccuracy handling in other domains, were searched for in the same resources. About ten additional publications were recorded. A walkthrough of the search results was made, providing additional five publications. Thus, the mapping study resulted in 38 articles addressing the uncertainty handling approaches relevant in the context of WDT-based analysis of deterministic systems.

The systematic literature review

For the systematic literature review we followed a procedure recommended by (Kitchenham, et al., 2007). The first phase of planning a systematic literature review is to specify the research objective that should be satisfied by conducting the review. The objective of this systematic literature review is stated in Table 1.

| Objective: | Identify and assess the potentially useful approaches which satisfy the sub-objectives 1, 2 and 3. |
| Sub-objective 1: | The approach is practically applicable for supporting uncertainty handling in the context of WDT-based analysis. |
| Sub-objective 2: | The approach handles the epistemic uncertainty. |
| Sub-objective 3: | The approach has a well defined propagation within the WDT. |

Table 1: The objective of the systematic literature review

By an approach we mean a collection of formal and graphical means available, along with the procedures prescribed for applying those means.

The second step of planning a systematic literature review is to develop the search strategy in the form of a review protocol. As the search captures a wide range of publications, we included a stopping heuristic.

The overall process is sketched in Figure 3. Since the search engines used sort the results by relevance, this procedure is effective in producing search results. In most cases, the density of relevant titles steadily decreased as we examined each list of the search results. The systematic literature review resulted in 17 relevant publications. The total number of the unique publications evaluated as relevant during both the mapping study and the systematic literature review was 42. Thus, there were 13 overlapping articles between the mapping study and the systematic literature review, and four of the overall 29 were found
through the systematic literature review, while 25 were found through the mapping study. 19 of the 25 relevant articles discovered only through the mapping study were published before 2000 – the timeframe which was excluded from the searches during the systematic literature review.

THE EVALUATION CRITERIA

Based on the objectives of our research, the results of the systematic literature review and the evaluation framework in (Nielsen, 1993), we have identified and iteratively refined a list of evaluation criteria. Table 2 provides an overview of the evaluation criteria. Each criterion is specified in detail in the Tables 3 through 9. Rationale behind the selected criteria is given in Appendix B.

<table>
<thead>
<tr>
<th>Criterion number</th>
<th>Criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Complexity of the representation form for the quantitative estimates.</td>
</tr>
<tr>
<td>C2</td>
<td>Suitability of the approach for both expert-judgment-based and measurement-based input.</td>
</tr>
<tr>
<td>C3</td>
<td>Scalability with respect to number of estimates needed on a WDT.</td>
</tr>
<tr>
<td>C4</td>
<td>Precision allowed by the representation form for the quantitative estimates.</td>
</tr>
<tr>
<td>C5</td>
<td>Complexity of the propagation.</td>
</tr>
<tr>
<td>C6</td>
<td>Expressiveness of the arithmetic operations for propagation.</td>
</tr>
<tr>
<td>C7</td>
<td>Empirical evaluation of the approach, as reported in the publications found through the literature review.</td>
</tr>
</tbody>
</table>

Table 2: An overview of the evaluation criteria

<table>
<thead>
<tr>
<th>Category number / Category</th>
<th>Criterion C1: Complexity of the representation form for the quantitative estimates.</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1_1</td>
<td>Simple and intuitive. Correspondence with the empirical input without need for a transformation of the notions from the input to the notions of the approach.</td>
</tr>
<tr>
<td>C1_2</td>
<td>A schema defining the correspondence of the notions used for the empirical input, is needed and relatively straight-forward to define.</td>
</tr>
<tr>
<td>C1_3</td>
<td>A schema defining the correspondence of the notions used for the empirical input, is needed but demanding to define.</td>
</tr>
</tbody>
</table>

Table 3: Specification of Criterion C1

<table>
<thead>
<tr>
<th>Category number</th>
<th>Criterion C2: Suitability of the approach for both expert-judgment-based and measurement-based input.</th>
</tr>
</thead>
<tbody>
<tr>
<td>C2_1</td>
<td>Both domain expert-judgment-based and measurement-based input can be directly related to and expressed by the notions of the approach for representing quantitative estimates.</td>
</tr>
<tr>
<td>C2_2</td>
<td>It is easy to directly express the measurement-based input but the domain expert judgments can only be expressed if its correspondence with the notions of the approach is defined.</td>
</tr>
<tr>
<td>C2_3</td>
<td>It is easy to directly express the domain expert judgments but the measurement-based input can only be expressed if its correspondence with the notions of the approach is defined.</td>
</tr>
<tr>
<td>C2_4</td>
<td>Neither the domain expert judgments nor the measurement-based input can be directly related to and by the notions of the approach regarding representation of the quantitative estimates. Their correspondence with the notions of the approach has to be defined.</td>
</tr>
</tbody>
</table>

Table 4: Specification of Criterion C2
### Table 5: Specification of Criterion C3

<table>
<thead>
<tr>
<th>Category number</th>
<th>Criterion C3: Scalability with respect to number of estimates needed on a WDT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3_1</td>
<td>Linearly proportional to the number of nodes with a low coefficient. A practically manageable number of estimates even on a large WDT.</td>
</tr>
<tr>
<td>C3_2</td>
<td>A practically managable number of estimates but only for a restricted WDT with few nodes. Linearly proportional to the number of nodes, but a large coefficient.</td>
</tr>
<tr>
<td>C3_3</td>
<td>Exponential relationship between number of nodes and the number of estimates needed. Practically demanding to provide the large number of estimates.</td>
</tr>
</tbody>
</table>

### Table 6: Specification of Criterion C4

<table>
<thead>
<tr>
<th>Category number</th>
<th>Criterion C4: Precision allowed by the representation form for the quantitative estimates.</th>
</tr>
</thead>
<tbody>
<tr>
<td>C4_1</td>
<td>Granularity of the uncertainty related to the empirical input can vary between and within parameters of a WDT.</td>
</tr>
<tr>
<td>C4_2</td>
<td>Granularity of the uncertainty related to the empirical input can vary between but not within parameters of a WDT.</td>
</tr>
<tr>
<td>C4_3</td>
<td>Granularity of the uncertainty related to the empirical input can be selected at a model level, but is, once defined by a scale for the model, frozen and can not vary within the WDT.</td>
</tr>
<tr>
<td>C4_4</td>
<td>Representation of uncertainty related to the empirical input is coarse grained, frozen and can not vary within the WDT.</td>
</tr>
</tbody>
</table>

### Table 7: Specification of Criterion C5

<table>
<thead>
<tr>
<th>Category number</th>
<th>Criterion C5: Complexity of the propagation.</th>
</tr>
</thead>
<tbody>
<tr>
<td>C5_1</td>
<td>Simple propagation which can be performed without tool support.</td>
</tr>
<tr>
<td>C5_2</td>
<td>Propagation can be followed by the domain experts without particular formal background. Tool support is needed and available.</td>
</tr>
<tr>
<td>C5_3</td>
<td>Propagation can be followed by the domain experts without particular formal background. Tool support is needed but its availability is not reported.</td>
</tr>
<tr>
<td>C5_4</td>
<td>Propagation is demanding to understand by the domain experts without formal background. Tool support is needed and available.</td>
</tr>
<tr>
<td>C5_5</td>
<td>Propagation is demanding to understand by the domain experts without formal background. Tool support is needed but its availability is not reported.</td>
</tr>
</tbody>
</table>

### Table 8: Specification of Criterion C6

<table>
<thead>
<tr>
<th>Category number</th>
<th>Criterion C6: Expressiveness of the arithmetic operations for propagation.</th>
</tr>
</thead>
<tbody>
<tr>
<td>C6_1</td>
<td>The arithmetic operations fulfill the needs for inference of all kinds of empirical input and its uncertainty.</td>
</tr>
<tr>
<td>C6_2</td>
<td>The arithmetic operations are more suited for subjective estimates than for propagation of the measurement-based input.</td>
</tr>
<tr>
<td>C6_3</td>
<td>The arithmetic operations are more suited for propagation of measurement-based input than for propagation of the subjective estimates.</td>
</tr>
<tr>
<td>C6_4</td>
<td>There are only limited and insufficient kinds of arithmetic operations for propagation of both expert-judgment-based and measurement-based kinds of input and its uncertainty.</td>
</tr>
<tr>
<td>Category number</td>
<td>Criterion C7: Empirical evaluation of the approach, as reported in the publications found through the literature review.</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>C7_1</td>
<td>Promising empirical evaluation reported, in terms of both validity (i.e., size and types of the systems the approach is applied on) and reliability (i.e., the possibility of obtaining the same results in repeated trials) of the evaluation method.</td>
</tr>
<tr>
<td>C7_2</td>
<td>The reported empirical evaluation indicates strong validity but weak reliability.</td>
</tr>
<tr>
<td>C7_3</td>
<td>The reported empirical evaluation indicates strong reliability but weak validity.</td>
</tr>
<tr>
<td>C7_4</td>
<td>Empirical evaluation reported but with considerable limitations with respect to both reliability and validity.</td>
</tr>
<tr>
<td>C7_5</td>
<td>No reports on empirical evaluation have been possible to extract from the publications found through the literature review.</td>
</tr>
</tbody>
</table>

Table 9: Specification of Criterion C7

THE HIGH-LEVEL EVALUATION

The results of our literature review show the following trends in terms of publication years and the kinds of publications for the respective approaches. The publications found regarding the imprecise probabilities date from 1991 to 2005 and consist of a book and two scientific papers. The publications found regarding Bayesian networks date from 1995, and consist of scientific papers. The publication referenced on the subjective logic is from 1997. Most of the publications found regarding Credal networks are from 2003 to 2007. The fuzzy logic relevant publications date from 1965 to 2008. The Bayesian networks relevant publications date from 1996 to 2009. The interval-based approaches have been addressed from 1994 to 2008 in several technical reports and scientific publications uncovered during the literature review.

This section reports on the results from the high-level evaluation of the uncertainty handling approaches, including analytical approaches for uncertainty representation, hybrid approaches in software engineering, as well as some of the uncertainty handling approaches in statistics, meteorology and software development effort estimation. The list is not exhaustive, but introduces the prevailing approaches which have been identified through the literature review. We categorize the approaches identified and outline each approach with respect to its properties, references and reported evaluation, all extracted from the publications which have been found through the literature review. Then we elaborate on whether the approach can be useful in the context of the WDTs.

Bayesian networks

Bayesian networks (Neil, Fenton, & Nielson, 2000) (Heckerman, Mamdani, & Wellman, 1995) are quantitative probabilistic networks that allow incorporating both model uncertainty and parameter uncertainty. A Bayesian network (BN) is a directed acyclic graph in which each node has an associated probability distribution. Observation of known variables (nodes) allows inferring the probability of others, using probability calculus and Bayes theorem throughout the model (propagation). BNs can represent and propagate both continuous and discrete uncertainty distributions. In a BN, probability distributions are calculated according to Baye’s rule, based on conditional probability tables (CPTs) and the probabilities of the parent nodes. The initial uncertainty is placed in the prior distribution of each input parameter. The prior distribution is then updated to a posterior distribution based on the observed data associated with each parameter. Additional data acquisition is undertaken until an acceptable certainty is reached. BNs are however demanding to parametrize and interpret the parameters of, an issue addressed in (Omerovic & Stølen, 2009), where an analytical approach for transforming WDTs to BNs is proposed.

The application of BNs is extensive. Bayesian approach for uncertainty analysis in software reliability modeling is for example applied in (Yin & Trivedi, 1999). Application of the Bayesian networks on large
scale systems has been reported in (Fenton & Neil, 1999). The reported empirical evaluation of the Bayesian networks is promising.

The applicability of the Bayesian networks on a WDT will be considered as a part of the low-level evaluation.

**Bootstrapping**

Bootstrapping may be parametric and nonparametric and is based on resampling methods. The former quantifies the effect of input parameter uncertainty for the parametric formulation. Using the available information, the parameters are first estimated by the maximum likelihood estimation. The estimates are used to draw a new sample of the observations. This process is repeated a number of times obtaining as many estimates as the number of repetitions, to the input parameters. The latter incorporates the error due to input distributions. The nonparametric resampling methods may model the distribution functions of independent input random variables.

The use of a percentile confidence interval is recommended in the bootstrapping approach in the absence of simulation uncertainty. Its use assumes that the statistics of interest are computed deterministically by the resampling methods. When simulation uncertainty is present, percentile confidence intervals are based on a convolution of the input uncertainty and simulation uncertainty. In the bootstrapping simulation method, it is impossible to separate these forms of uncertainty (Batarseh & Wang, 2008).

Bootstrapping is barely presented in the publications found and information on its performance in the empirical studies has not been possible to extract. However, its main disadvantage is the focus on the measurement-based uncertainty, and lack of a representation form suitable for expressing the uncertainty within the expert judgments.

**ISO guide to handling measurement uncertainty**

The ISO guide to handling measurement uncertainty (ISO, 2008) uses a probabilistic representation with normal distribution, and treats both aleatory and epistemic uncertainty equally. It also contains instructions on the propagation of the uncertainties. We argue however that such an approach does not explicitly account for the notion of ignorance about the estimates.

As argued in relation to the deduction of the evaluation criteria, discrete representations are better suited than continuous ones to capture epistemic uncertainty.

**Delta method**

The Delta method developed by Cheng and Holloand (Cheng & Holloand, 1997) for input uncertainties assumes that the model is known while input parameters are uncertain. The true values of the parameters are estimated by the maximum likelihood estimation, assuming that the parameters follow a normal distribution. The total simulation output variance is estimated by simulation variance and the input parameter variance. No report on the evaluation of the performance of this approach in the empirical studies is found within the literature review.

As argued above, discrete representations are better suited than continuous ones to capture epistemic uncertainty.

**Qualitative probabilistic networks**

Qualitative probabilistic networks are qualitative abstractions of probabilistic networks. A qualitative probabilistic network comprises an acyclic digraph which models variables and the probabilistic
relationships between them. Instead of conditional probability distributions, a qualitative probabilistic network associates with its digraph qualitative influences and qualitative synergies (Wellman, 1990).

Qualitative probabilistic networks are barely present in the publications considered by us. We have not been able to extract information on their performance in the empirical studies. A drawback of the qualitative networks is the lack of precision in the representation of the uncertain quantitative parameters. Measurement-based quantities are for example more exact than what can be expressed by the qualitative networks.

Credal networks

A Credal network (Cozman, 2005) is a graphical model that associates nodes and variables with sets of probability measures. The motivation for Credal networks is the need for more relaxed Bayesian networks. In a Bayesian network, the Markov property implies that we must specify a unique probability distribution for every variable conditional on any configuration of the variable’s parents. However, existing beliefs may be incomplete or vague, or there may be no resources to gather or process the input so as to reach a precise probability assessment. The point estimates may not be exact or the domain experts may disagree. Hence, probability intervals may be selected as estimates. As a result, a set of probability distributions is specified for every variable conditional on the variable's parents, thus obtaining a Credal network.

Credal networks are like discrete Bayesian networks, except that they specify closed convex sets of probability mass functions, so-called Credal sets (Levi, 1980), instead of single probability mass functions. Every Credal network variable is associated with a collection of local conditional Credal sets, which do not interfere with each other. (Haenni, 2007) introduces an approximate propagation algorithm for Credal networks. The algorithm represents the Credal network as a compiled logical theory. The resulting graphical structure is the basis on which the subsequent steepest-ascent hill-climbing algorithm operates. The output of the algorithm is an inner approximation of the exact lower and upper posterior probabilities.

Real applications of a complex Credal network constructed both from expert opinions and data, have been reported in (Antonucci, Salvetti, & Zaffalon, 2004). A Credal network constructed from data and used for classification in a medical scenario is presented in (Zaffalon, Wesnes, & Petrini, 2003).

A drawback of the Credal networks is lack of scalability due to the need for providing an extensive number of estimates even for relatively small networks. Another weakness is the interpretation of the probability intervals and their correspondence with the non-probabilistic notions on the WDTs. We consider the quantitative representation of the uncertain estimates on the Credal networks to be too complex in terms of both the number of the estimates needed and their probability-interval-based form, which is non-trivial to interpret by the domain experts. Moreover, the measurement-based input needs to be related to the probability-interval-based representation form.

Intervals

Definitions of intervals, their arithmetics and central tendency operators are provided by (Ferson, Kreinovich, Hajagos, Oberkampf, & Ginzburg, 2007). Additional comprehensive references on the interval-based approach are (Majumdar, et al., 2001), (Kearfott, 1994) and (Kreinovich, et al., 2008). The practical interval solution depends on the monotonicity properties (the relationship between model input and output) of the model. The probability of occurrence of any value within an interval can follow any given arbitrary distribution.
As argued in (Majumdar, et al., 2001), interval arithmetics can serve as a tool to obtain interval extensions of real functions. However, due to a so-called overestimation effect, also known as dependency problem, interval arithmetic does not provide the exact range of a function. The dependency problem is due to the memoryless nature of interval arithmetic in cases when a parameter occurs multiple times in an arithmetic expression, since each occurrence of an interval variable in an expression is treated independently. Since multiple occurrences of interval parameters cannot always be avoided, the dependency problem often causes overestimation of the actual range of an evaluated function. A way to handle this issue is to use interval splitting (Majumdar, et al., 2001), where the input intervals are divided and the arithmetics are performed on the subintervals. The final results are then obtained by computing the minimum of all lower bounds and the maximum of all upper bounds of the intermediate results. In (Skelboe, 1974) it is shown that the results obtained from the interval splitting converge to the actual range when the width of the subintervals approaches zero. The application of this techniques on performance models is reported in (Majumdar & Ramadoss, 1995).

(Majumdar, et al., 2001) present intervals and extended histograms for characterizing system parameters that are associated with uncertainty and variability. Histograms are expressed by a series of intervals with associated probabilities. Adaptation of the existing approaches for analytic performance evaluation for this interval-based parameter characterization is described. Its application is illustrated by two examples: a hierarchical model of a multicomputer system and a queueing network model of an EJB (Enterprise Java Bean) server implementation.

An approach to assembling, structuring and representing evidence in a decision, based on hierarchical modeling of the process leading up to a decision, is presented in (Davis & Hall, 2003). The propagation is achieved through the evidence hierarchy, using interval probability theory. Case studies within oil and civil engineering industries are used to demonstrate the approach.

The applicability of the interval-based approach on a WDT will be considered as a part of the low-level evaluation.

**Subjective logic**

Subjective logic (Jøsang, 1997) is a framework for formal reasoning, which consists of a belief model called *opinion* and a set of operations for combining opinions. A single opinion π is uniquely described as a point (b,d,i) in an opinion triangle, where b, d and i designate belief, disbelief and ignorance, respectively. For each opinion, the three notions sum up to unity. The operations formally defined include: conjunction (“∧”), disjunction (“∨”), negation (“¬”), consensus (“⊕”), recommendation (“⊗”) and ordering (“↑”).

The subjective logic is suited for domain expert judgments, but how the established representations of the automatically acquired input can be related to the concepts of the subjective logic, is unclear. In the context of a WDT, the leaf node estimates could be expressed in terms of opinions and propagation can be based on the algebra of subjective logic. Subjective logic can generally be applied when taking advices from different sources.

The applicability of the subjective logic – based approach on a WDT will be considered as a part of the low-level evaluation.

**Fuzzy logic**

Fuzzy logic provides a simple way of drawing definite conclusions from vague, ambiguous or imprecise information, and allows for partial membership in a set. Furthermore, fuzzy logic facilitates the modeling
of complex systems using higher levels of abstraction originating from the analyst's knowledge and experience (Weber, 1994). A fuzzy set is a class of objects with a continuum of grades of membership. Such a set is characterized by a membership function which assigns to each object a grade of membership ranging between zero and one (L.A Zadeh, 1965).

Using the fuzzy membership functions, a parameter in a model can be represented as a crisp number, a crisp interval, a fuzzy number and a fuzzy interval. The ordinate of a graphical representation of the membership distribution is the membership function, which may be thought of as "level of presumption", while the abscissa is the interval of confidence. Thus, a crisp number is represented by \((a,a,0,0)\) where \(a\) is a sharp value on the x-axis with maximum level of presumption; a crisp interval is represented by \((a,b,0,0)\) where the interval on the x-axis ranges from \(a\) to \(b\), while the lever of presumption is flat and equal to one for the entire range of values from \(a\) to \(b\); a fuzzy number is represented by \((a,a,x,y)\) where the level of presumption continuously increases from zero to one for the x-axis values starting at \(a-x\) and ending at \(a\), while from \(a\) to \(a+y\) values on the x-axis the level of presumption continuously decreases from one to zero. Thus, sketch of the membership function of a fuzzy number forms a triangle. A fuzzy interval is represented by \((a,b,x,y)\) where for the range of values from \(a-x\) to \(a\) on the x-axis the level of presumption continuously increases from zero to one, for the range of values from \(a\) to \(b\) on the x-axis the level of presumption is one, and for the range of values from \(b\) to \(b+y\) on the x-axis the level of presumption continuously decreases from one to zero. In the approach of fuzzy logic, the algebraic operations are easy and straightforward, as argued and elaborated in (Suresh, Babar, & Raj, 1996). Fuzzy logic is non-statistical in nature, but with a well defined arithmetics.

The degrees of membership in a fuzzy set are certain. Therefore, this type of fuzzy sets can not deal, for example, with uncertainty that arises when more than one expert provides judgments. In addition, this type of fuzzy sets can not accommodate different experts’ definitions of the low size fuzzy set. As a result, the corresponding fuzzy logic is incapable of dealing with uncertainties of this kind. This is addressed by an extended definition of fuzzy sets which incorporates the notion of uncertainty discussed above, the so-called type-2 fuzzy sets (Ahmed & Muzaffar, 2009). In the type-2 fuzzy sets the membership function is three-dimensional. The first two criteria allow handling imprecision via modeling the degree of membership; while the third criterion allows handling uncertainty via modeling the amplitude distribution of the degree of membership.

Application of fuzzy logic for uncertainty handling in software effort prediction has been reported in (Ahmed, Omolade Saliu, & AlGhamdi, 2005). A graph-based representation with learning algorithms is presented. A validation experiment has been carried out on artificial datasets as well as the COCOMO public database. Additionally, an experimental validation of the training procedure is presented. The authors argue that the fuzzy logic – based framework incorporates expert knowledge, historical data, imprecision handling and adaptability. The uncertain fuzzy estimates are expressed in terms of triangular membership functions.

A case study regarding a type-2 fuzzy system has been conducted on COCOMO II data for software effort prediction (Ahmed & Muzaffar, 2009). The evaluation is based on root mean square error. Each attribute for prediction is divided into a selected number of fuzzy sets. The evaluation results are promising.

The applicability of the fuzzy logic – based approach on a WDT will be considered as a part of the low-level evaluation.

**Dempster-Shafer structures**

The Dempster-Shafer structures (Ferson, et al., 2007) offer a way of representing uncertainty quantified by mass function distributions. A mechanism for aggregation of such representation stored in distributed
relational databases, is proposed in (Scotney & McClean, 2003). The Dempster-Shafer approach characterizes uncertainties as intervals with degrees of certainty (that is, sets of values with weights which add up to one). It can be seen as a generalization of both interval analysis and probability theory. Weights of evidence are put on a collection of intervals and the structures may overlap.

Dempster-Shafer evidence theory is a possible approach for epistemic uncertainty analysis. It relaxes the assumptions of the probability theory and allows for combining conflicting or ambiguous evidence from multiple sources. In probability theory, however, evidence is associated with only one possible source or event. Dempster-Shafer evidence theory models the epistemic uncertain input variables as sets of intervals. Each variable may be defined by one or more intervals. The user specifies a probability assignment to each interval. The probability assignment indicates how likely it is that the uncertain input falls within the interval. The probability assignments for a particular uncertain input variable must sum to one. The intervals may over-lap, chain or gap relative to one another. Dempster-Shafer has two measures of uncertainty, belief and plausibility. The intervals are propagated to calculate belief and plausibility. Together, belief and plausibility define an interval-valued probability distribution.

Dempster-Shafer interval calculation is computationally expensive due to the number of the interval combinations (within and across each variable) to be included in the propagation. Minimum and maximum function values are searched for within each interval combination. The minimum and maximum values are aggregated to create the belief and plausibility. The accuracy of the Dempster-Shafer approach depends on the number of samples and the number of interval combinations.

Implementing the Dempster-Shafer theory in the context of WDTs involves solving two issues: 1) sorting the uncertainties in the empirical input into a priori independent items of evidence, and 2) carrying out Dempster's rule computationally. The former leads to a structure involving input elements that bear on different but related concerns. This structure can be used to make computations based on Dempster's rule, feasible.

Although it is more precise than e.g. the intervals, the main weakness of the Dempster-Shafer structures in the WDT context is the complexity of the representation of the quantitative estimates. This may compromise the comprehensibility of the uncertainty representation and therefore the correctness of the input provided by the domain experts.

Evidence theory

Both possibility theory, Dempster-Shafer theory and fuzzy sets theory are special cases of the evidence theory. It relies on the notion of random set, each set-valued realization representing an incomplete information item. The evidence theory allows combining evidence from different sources and arriving at a degree of belief (represented by a belief function) that takes into account all the available evidence. In evidence theory, likelihood is assigned to sets, as opposed to probability theory where likelihood is assigned to a probability density function. In the evidence theory, two main functions help us to obtain information about the uncertainty in the knowledge, the plausibility and the belief functions.

The evidence theory of (Shafer, 1976) is a branch of the mathematics of uncertain reasoning that has been presented as an alternative decision theory, as a purely subjectivist, non-statistical approach to uncertain evidence. (Fioretti, 2004) presents the principles of evidence theory and its application in creative decision making. Details on epistemic uncertainty representation using evidence theory can be retrieved from (Simon, Weber, & Levrat, 2007), which argues for shortcomings of the Bayesian networks in representing uncertain knowledge, and proposes an integration of the Bayesian networks model of reliability and the evidence theory, resulting in a so-called Evidential Network.
We consider the evidence theory as too complex for being adopted on the WDTs. The representation form is therefore not necessarily suited for the expert-judgment-based input.

Imprecise probabilities

In the imprecise probability theory, developed in (Walley, 1991), sets of probability functions capture the notion of partial lack of probabilistic information. The imprecise probabilities approach the probability theory through lower and upper probabilities, rather than probabilities. (Cozman, 2005) presents an overview of graphical models that can handle imprecision in probability values. A review of the algorithms for local computation with imprecise probabilities is presented in (Cano & Moral, 1999), where the objective is to carry out a sound global computation by mainly using the initial local representation. These algorithms try to solve problems of propagation (calculation of conditional or unconditional probabilities) in cases in which there is a large number of variables. There are two main types depending on the nature of the assumed independence relationships. In both of them the global knowledge is composed of several pieces of local information. It has not been possible for us to extract information on performance of imprecise probabilities in the empirical studies.

A drawback of the imprecise probabilities is the interpretation of the probability intervals and their correspondence with the non-probabilistic notions of the WDTs. We consider the quantitative representation of the uncertain estimates on the imprecise probabilities to be too complex. The probability interval – based form is non-trivial to interpret by the domain experts. Consequently, the actual estimates available by the domain experts may be difficult to provide in a precise and correct form. Due to the number of estimates needed (as a consequence of the propagation based on Baye’s rule), the imprecise probabilities – based approach also suffers from low scalability.

Hybrid approaches

The propagation of imprecise probabilities in Bayesian networks is addressed in (Kleiter, 1996), where the imprecision is handled by second order probability distributions. The Dirichlet distributions are used to express the uncertainty about probabilities. The problem of how to propagate point probabilities in a Bayesian network is transformed into the problem of how to propagate Dirichlet distributions in Bayesian networks.

(Renoij & van der Gaag, 2002) present a new type of network in which both signs and numbers are specified (a combination of a qualitative and quantitative network). An associated algorithm for probabilistic propagation is also presented. Building upon these semi-qualitative networks, a probabilistic network can be quantified and studied in a stepwise manner. As a result, modeling inadequacies can be detected and amended at an early stage in the quantification process.

(Ferson, et al., 2007) propose considering a hybrid approach comprising both probabilistic and possibilistic (based on intervals) representation, in order to account for both aleatory and epistemic uncertainty.

(Baudrit, Cousso, & Dubois, 2007) propose mathematical models for joint uncertainty propagation involving quantities respectively modeled by probability and possibility distributions in the context of risk analysis.

The Maximum-entropy principle and the Bayesian approach have in a hybrid approach (Dai, Xie, Long, & Ng, 2007) been proposed for uncertainty analysis in software reliability modeling. The approach proposed integrates the capability of the maximum entropy principle the Bayesian approach to derive the priori distribution, which can incorporate historical data with expert judgments, constraints and expected
values. This is specifically appropriate for highly reliable software, where only a few failure data are available from a test.

A hybrid Monte Carlo and possibilistic approach for representation and propagation of aleatory and epistemic uncertainty is presented in (Baraldi, Popescu, & Zio, 2008). The approach is applied for predicting the time to failure of a randomly degrading component, and illustrated by a case study. The hybrid representation captures the aleatory variability and epistemic imprecision of a random fuzzy interval in a parametrized way through $\alpha$-cuts, and displays extreme pairs of the upper and lower cumulative distributions. The Monte Carlo and the possibilistic representations are jointly propagated. The gap between the upper and the lower cumulative distributions represents the imprecision due to epistemic variables. The possibility distributions are aggregated according to the so-called Ferson method. The interpretation of the results in the form of limited cumulative distributions requires the introduction of a degree of confidence directly connected with the confidence on the value of epistemic parameters.

A simulation mechanism which takes into account both aleatory and epistemic uncertainty in an interval-based approach, is proposed in (Batarseh & Wang, 2008). It concentrates on stochastic simulations as input for the interval estimates, when significant uncertainties exist. Probabilistic distributions represent variabilities and intervals capture epistemic uncertainty. The interval width is proportionally related to the uncertainty. The calculations are based on interval arithmetic, and interval statistics methods are used to report the mean and the standard deviation to provide a concise summary of the results.

(Baraldi, et al., 2008) illustrate a hybrid Monte carlo and possibilistic approach for the representation and propagation of aleatory and epistemic uncertainty. The approach obtained promising results in a case study addressing the lifetime of a component which is randomly degrading in time.

The hybrid approaches are mainly based on stochastic simulation or probabilistic notions based on continuous representations, which are, as argued earlier, not suited for handling the epistemic uncertainty. That is also why the hybrid approaches have not been selected for the low-level evaluation.

**Uncertainty handling in other domains**

The approaches to uncertainty handling in other domains, such as weather forecasting (Palmer, 2000), electricity demand forecasting (Taylor & Buizza, 2003), correlations between wind power and meteorological conditions (Lange & Heinemann, 2002), forecast uncertainty in power system planning (Douglas, Breipohl, Lee, & Adapa, 1998), and forecast uncertainty in supply industry (Lo & Wu, 2003) are mainly based on probabilistic representations and stochastic simulations using continuous representations.

Uncertainty representation in approaches to effort estimation in software development effort like (Grimstad & Jørgensen, 2007) and (Gruschke & Jørgensen, 2005) are based on intervals for prior estimates, without a requirement for their propagation within the model. The approach to effort estimation in (Ahmed, et al., 2005) uses fuzzy logic with a learning algorithm for the structure and the parameters. (Sarcia, Basili, & Cantone, 2009) show that uncertainty can be used as an invariant criterion to figure out which effort estimation model should be preferred over others. This work is mainly empirical, applying Bayesian prediction intervals to some COCOMO model variations with respect to a publicly available cost estimation data set coming from the PROMISE repository.

In (Davis & Hall, 2003), an approach to assembling, structuring and representing evidence in a decision, based on hierarchical modeling of the processes leading up to a decision, is presented. Uncertainty in the available evidence is represented and propagated through the evidence hierarchy using interval
probability theory. Case studies in the oil and civil engineering industries are used to demonstrate how the approach facilitates developing shared understanding of the implications of uncertainty.

THE LOW-LEVEL EVALUATION
This section presents the results of the low-level evaluation. Each approach identified for the low-level evaluation is:

- instantiated on a WDT, in order to test how its representation form and propagation can be applied, and
- analyzed further with respect to the evaluation criteria.

Bayesian networks – based approach on a WDT

One way to handle uncertainty in a WDT using a BN representation is sketched in Figure 4, where each node is assigned six states. The uncertainty representation is contained in both the granularity (the number of states on the nodes) and their respective probabilities. Then, Baye’s rule is applied for the propagation. The prior estimates consist of the probabilities associated with each state on the leaf nodes, as well as the dependencies estimated through the values in the conditional probability table (CPT). The probabilities of the states on each node have to sum up to 100. The CPT in the case of the WDT in Figure 4 consists of 216 (that is, the product of the number of states in the children and parent nodes) parameters and must be estimated before the root node can be propagated according to Baye’s rule.

Due to the number of the estimates needed, the BN-based approach does not scale. Furthermore, since the uncertainty representation of several states for each node requires extensive CPTs and a demanding propagation, the BN-based approach suffers from lower comprehensibility. The precision depends on the number of states introduced for each node, which means that e.g. new evidence of higher precision than the one defined through the existing states, may be difficult to represent. Hence, combining evidence of different uncertainty (e.g., expert judgments and measurements) on a node may be problematic. The propagation is well-defined but complex.

Interval-based approach on a WDT

One way of using intervals to capture uncertainty in a WDT is illustrated by Figure 5. The prior intervals are represented by values of type
\[ x = [x; \bar{x}] = \{X \in [0; 1] : x \leq X \leq \bar{x} \} . \]

The propagation in Figure 5 is based on a linear model where the parent node interval is obtained by multiplying the respective values on the arcs and the nodes of the immediate children, and summing up these products. Optimization is performed to find the upper and the lower limits of the inferred interval, so that the quantities on the arcs still sum up to one (Omerovic & Stølen, 2010.). The only two interval arithmetic operations needed for propagation in this example are addition and multiplication. In case of two intervals denoted by \( x \) and \( y \) (of the form given specified above), addition and multiplication are defined as:

\[ x \circ y = [x \circ y; x \circ y], \]

where \( \circ \) denotes the operation symbol.

The interval-based approach has several advantages. It is relatively straight-forward with respect to both estimate representation and propagation. Moreover, it is comprehensible and easy for the domain experts to relate to, regardless of their formal background. The number of estimates needed is lower than in most of the competing approaches. At the same time, measurement-based input can easily be converted to intervals by relating standard deviation to the interval width. There are no restrictions on the interval width, so the precision is adjustable and can vary among the estimates. Interval arithmetics and interval statistics are well defined and expressive. An issue for this kind of propagation is however overestimation as discussed in relation to the high-level evaluation. Moreover, interval splitting is difficult due to its computational complexity.

**Subjective logic – based approach on a WDT**

Figure 6 illustrates how we may introduce uncertainty in a WDT using a subjective logic – based representation in the form of opinions on the nodes and applying the arithmetics of subjective logic for the propagation. Ordering was applied on \( \pi_{31} \) and \( \pi_{32} \) to obtain \( \pi_{24} \). Furthermore, recommendation was applied to obtain \( \pi_{12} \), consensus to obtain \( \pi_{11} \), and conjunction to obtain the opinion of the root node.

The main challenge with the subjective logic approach is relating the notions and values within each opinion to the empirical input. There are no recommendations that we are aware of regarding the interpretation of the notions from the subjective logic to the measurements. How a measurement represented by a mean and a standard deviation can be converted to an opinion, is therefore unclear. However, subjective logic includes the “base rate” (atomicity) that might be interpreted as the deviation...
whereas the mean is an opinion with low uncertainty. Still, lack of guidelines for interpretation and conversion may degrade comprehensibility, result in misleading input, and disable combining the expert-judgment-based input with the measurement-based one.

\[
\pi_0 = (0.09, 0.39, 0.52)
\]

\[
\pi_{11} = (0.7, 0.15, 0.15)
\]

\[
\pi_{21} \odot \pi_{22}
\]

\[
\pi_{21} = (0.6, 0.2, 0.2)
\]

\[
\pi_{22} = (0.6, 0, 0.4)
\]

\[
\pi_{23} = (0.7, 0.1, 0.2)
\]

\[
\pi_{24} = (0.7, 0.3, 0)
\]

\[
\pi_{25} = (0.3, 0.5, 0.2)
\]

\[
\pi_{12} = (0.15, 0.25, 0.6)
\]

\[
\pi_{23} \odot \pi_{24} \odot \pi_{25}
\]

\[
\pi_{13} = (0.9, 0, 0.1)
\]

\[
\pi_{31} = (0.7, 0.3, 0)
\]

\[
\pi_{32} = (0.7, 0.1, 0.2)
\]

\[
\pi_{33} = (0.5, 0.5, 0.0)
\]

\[
\pi_{34} = (0.3, 0.3, 0, 0)
\]

\[
\pi_{35} = (0.6, 0.4, 0.8)
\]

\[
\pi_{36} = (0.7, 0.7, 0, 0)
\]

\[
\pi_{37} = (0.3, 0.5, 0.2, 0.6)
\]

\[
\pi_{38} = (0.3, 0.3, 0.2, 0.4)
\]

\[
\pi_{39} = (0.3, 0.3, 0, 0)
\]

\[
\pi_{40} = (0.5, 0.5, 0.3, 0.7)
\]

Figure 6: Uncertainty handling in a WDT using subjective logic

The arithmetics are rich and well suited for expert-judgment-based input. The propagation is however demanding, and requires tool support. Given that the interpretation is well defined, the estimate representation based on the subjective logic is precise.

**Fuzzy logic – based approach on a WDT**

A possible way of using the Fuzzy logic – based approach is illustrated by Figure 7. In this example, the values on the leaf nodes define the membership functions, while the internal nodes express the operators to be applied on the “value”, which is a variable expressed on the root node. In practice, there may be many “value” variables expressed within the WDT, and they can then be applied on the operators specified. Several kinds of membership functions can be combined, as illustrated by Figure 7.

\[
\text{IF } \text{value} < 0.9 \text{ then correct}
\]

\[
\text{AND}
\]

\[
\text{NOT} \quad (0.3, 0.5, 0.2, 0.6)
\]

\[
(0.2, 0.6, 0, 0) \quad (0.6, 0.6, 0.4, 0.8) \quad (0.7, 0.7, 0, 0) \quad \text{OR} \quad (0.3, 0.3, 0.2, 0.4)
\]

\[
(0.3, 0.3, 0, 0) \quad (0.5, 0.5, 0.3, 0.7)
\]

Figure 7: Uncertainty handling in a WDT using fuzzy logic
The fuzzy approach is expressive in terms of both estimate representation and arithmetics. There is a wide variety of arithmetic operators which can be selected and applied. Although the propagation is non-trivial, tool support is available. The precision level can be selected.

The extensive expressiveness that the membership functions and arithmetic operators offer may however degrade comprehensibility, since their variety also increases complexity if compared to for example the interval-based approach (which can be considered as a special case of the fuzzy one). The increased complexity of the estimate representation may not necessarily contribute to the precision of the parameter values, but rather introduce misinterpretations and incorrectnesses in the provisioning of the expert-judgment-based input. The interpretation of the membership distributions and their correspondence to the practical settings may be demanding.

The measurement-based input may however benefit from the different membership distributions. Since the membership functions can be selected and combined, the preferred one can be used in each case, thus giving a high precision. Thus, both expert-judgment-based and measurement-based kinds of input are fully supported.

**Summary of the low-level evaluation**

Table 10 summarizes the approaches which have undergone low-level evaluation. The properties of each approach are classified according to the definitions of the evaluation criteria and their respective categories, and based on the findings from the evaluation.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C1: Complexity</td>
</tr>
<tr>
<td></td>
<td>of the</td>
</tr>
<tr>
<td></td>
<td>representation</td>
</tr>
<tr>
<td></td>
<td>form for the</td>
</tr>
<tr>
<td></td>
<td>quantitative</td>
</tr>
<tr>
<td></td>
<td>estimates.</td>
</tr>
<tr>
<td></td>
<td>C2: Suitability</td>
</tr>
<tr>
<td></td>
<td>of the approach</td>
</tr>
<tr>
<td></td>
<td>for both</td>
</tr>
<tr>
<td></td>
<td>expert-judgment-</td>
</tr>
<tr>
<td></td>
<td>based and</td>
</tr>
<tr>
<td></td>
<td>measurement-</td>
</tr>
<tr>
<td></td>
<td>based input.</td>
</tr>
<tr>
<td></td>
<td>C3: Scalability</td>
</tr>
<tr>
<td></td>
<td>with respect to</td>
</tr>
<tr>
<td></td>
<td>number of</td>
</tr>
<tr>
<td></td>
<td>estimates needed</td>
</tr>
<tr>
<td></td>
<td>on a WDT.</td>
</tr>
<tr>
<td></td>
<td>C4: Precision</td>
</tr>
<tr>
<td></td>
<td>allowed by the</td>
</tr>
<tr>
<td></td>
<td>representation</td>
</tr>
<tr>
<td></td>
<td>form for the</td>
</tr>
<tr>
<td></td>
<td>quantitative</td>
</tr>
<tr>
<td></td>
<td>estimates.</td>
</tr>
<tr>
<td></td>
<td>C5: Complexity</td>
</tr>
<tr>
<td></td>
<td>of the propagation.</td>
</tr>
<tr>
<td></td>
<td>C6: Expressiveness of the arithmetic operations for propagation.</td>
</tr>
<tr>
<td></td>
<td>C7: Empirical evaluation of the approach, as reported in the publications found through the literature review.</td>
</tr>
</tbody>
</table>

| Bayesian networks – based approach | C1_2 | C2_4 | C3_3 | C4_2 | C5_3 | C6_1 | C7_1 |
| Interval-based approach           | C1_2 | C2_3 | C3_1 | C4_1 | C5_4 | C6_3 | C7_4 |
| Subjective logic – based approach | C1_3 | C2_4 | C3_1 | C4_1 | C5_4 | C6_2 | C7_5 |
| Fuzzy logic – based approach      | C1_1 | C2_1 | C3_2 | C4_1 | C5_3 | C6_1 | C7_1 |

Table 10: Summary of the low-level evaluation
The values assigned to the approaches represent the category of the criterion which the approach fits into. Note that in the case of each criterion, the categories have been ordered from highest possible level of fulfillment, to the lowest one. Where appropriate, the domain-expert-judgment-related categories are ranking higher among categories of a criterion, than the measurement-related ones. Thus, the rightmost digit in the category indicates the level of fulfillment of the criterion.

CONCLUSION

Based on a literature review, the potentially useful approaches for uncertainty handling in WDTs are identified and evaluated with respect to a set of evaluation criteria. The evaluation criteria are deduced from the objective of the systematic literature review and related to an established framework for practical system acceptability. Hence, in addition to the literature review and the evaluation, this paper contributes with a proposal of a set of evaluation criteria regarding selection of uncertainty handling approaches in a WDT context. The proposed criteria are concerned with:

- Complexity of the representation form for the quantitative estimates.
- Scalability of the approach for both expert-judgment-based and measurement-based input.
- Scalability with respect to number of estimates needed on a WDT.
- Precision allowed by the representation form for the quantitative estimates.
- Complexity of the propagation.
- Expressiveness of the arithmetic operations for propagation.
- Empirical evaluation of the approach, as reported in the publications found through the literature review.

The paper may serve to researchers and practitioners as a roadmap for examining the uncertainty handling approaches, or as a resource for identifying the adequate one. The summary presented in Table 10 indicates a high degree of variation with respect to the scores of the approaches evaluated at low level.

A weakness of all the approaches evaluated at low level is the assumption that the model is certain, while only the input parameters are uncertain. For deterministic systems, this assumption is justifiable. However, in most practical cases, both the model and the input parameters are uncertain. Here, we also have aleatory uncertainty. A model-general uncertainty measure should therefore be introduced in relation to the WDTs, in order to express the uncertainty at an overall level due to for example context uncertainty or usage profile uncertainty.

We argue that the epistemic uncertainty is suited for possibilistic uncertainty representations, that is the representations where pieces of information take the form of fuzzy sets (L. A. Zadeh, 1999) of possible values. The merit of this framework lies in its simplicity, which enables incomplete probabilistic information on the real line to be encoded in the form of fuzzy intervals (Dubois & Prade, 1987). Possibility distributions can also straightforwardly accommodate linguistic information on quantitative scales. As argued by (Möller & Beer, 2008), in many cases of engineering practice, not even subjective probabilistic information is available. Examples are uncertain quantities for which mere bounds or linguistic expressions are known. A probabilistic modeling would then introduce unwarranted information in the form of a distribution function that is totally unjustified.

Selection of the appropriate approach for uncertainty handling implies finding the right balance between the practical applicability on the one hand and the functional properties of the approach on the other hand.

By practical applicability we mean the usability of the approach for the domain experts, who have in-depth system knowledge, but not necessarily a profound insight into formal uncertainty representation. That is, a good approach should offer a representation form for empirical input and uncertain parameter
estimates that domain experts find easy to interpret and use. This is partly evaluated through criteria C1 and C5. In addition, usability implies the possibility of combining both expert-judgment-based and measurement-based input, as captured by criterion C2. Another aspect of the practical acceptability is the scalability of the approach with respect to the number of prior estimates needed for generating a WDT, as captured by criterion C3.

The preciseness and expressiveness of the approach are captured by criteria C2, C4 and C6, which also address the functional properties of the approach, that is, handling of epistemic uncertainty and well defined propagation.

The needs regarding practical applicability and functionality are obviously not independent. For example, an approach which is overly advanced in terms of propagation or quantitative estimate representation may be too complex for the domain experts to relate to. Such an approach may be formally sound and precise, but not applicable in a practical setting.

The comprehensibility of an approach may degrade with the increased level of the detail or the size of the model. A coarse grained representation may be more comprehensible, but its model accuracy may degrade and the available input may be lost due to the low expressiveness.

On the other hand, an approach which is well suited for representing measurement-based input may not support equally well the representation of the expert judgments. Similarly, an approach which is very expressive may not scale in an industrial setting, due to the size of the model or the necessity of providing too many parameter estimates.

Based on our evaluation, we can of course not claim that one approach is better than the others for all purposes. The approach must be selected based on the concrete needs and how they are reflected by our evaluation criteria. The criteria should be weighted based on the needs of the context in question. The appropriate trade off is particularly important when the input is based on both expert judgments and measurements, since presence of both kinds of input strengthens the interaction between the functional and the non-functional properties mentioned above.

Nevertheless, it can not be denied that the interval-based approach performs well, due to its well defined formalisms and comprehensibility. The width of the interval is proportional to the uncertainty, and propagation is based on interval arithmetics. The interval-based approach allows comprehensible representation of uncertainty on all kinds of parameters, with the needed accuracy. Estimation, propagation and analysis in the interval-based approach are scalable and efficient. The interval-based approach can easily be applied on both measurement-based and expert-judgment-based input simultaneously, since the conversion of the uncertain estimate representation of the former to the latter is well defined.

In the case of the Dempster-Shafer structures, the uncertainty may vary across the fractions of the intervals. This additional expressiveness however weakens the comprehensibility due to an extensive increase in the level of detail and may result in compromised correctness of the input.

We argue that Bayesian networks are expressive and precise, but in their general form they are demanding to parametrize and interpret. Thus, they suffer from lower scalability and comprehensibility.

In fuzzy logic the algebraic operations are easy and straightforward. The interval-based approach may be seen as a special case of the fuzzy approach, where only the crisp intervals are used as the membership function. The additional expressiveness that the overall types of the membership functions offer increases complexity of the estimate representation and may introduce misinterpretations and incorrectness in the
provisioning of input. The interpretation of the membership distributions and their correspondence to the practical settings may be demanding and therefore unlikely to increase accuracy. Still, the fuzzy logic-based approach performs very well in the evaluation, due to its high expressiveness in terms of representation of the quantitative estimates, rich arithmetics and flexibility in terms of complexity needed to be adopted.

A subjective logic-based approach may provide additional accuracy for domain expert-judgment-based estimates, but how measurement-based input can be converted to the notions within subjective logic, is not clear.

The hybrid approaches address both epistemic and aleatory uncertainty and the propagation is based on stochastic simulation. In most cases, both possibilistic and probabilistic representations are adopted for handling the two respective kinds of uncertainty. Although the hybrid approaches address both kinds of uncertainty, their major weakness is the high complexity, which degrades the overall comprehensibility. They generally focus equally much on both kinds of uncertainty, something that may represent a bias in our case where the epistemic uncertainty is the prevailing one.

Based on our results we recommend the further development of the uncertainty handling approaches to focus on the practical aspects such as scalability, comprehensibility and support for soft computing, in order to facilitate human interaction in the input acquisition. Integrated approaches supporting handling of both measurement-based and expert-judgment-based input in a manner which is comprehensible and accurate, will be particularly useful. Moreover, the existing approaches should be exposed to further empirical investigations in a real life setting.

ACKNOWLEDGMENT
This work has been conducted within the DIGIT (180052/S10) project, funded by the Research Council of Norway. The authors acknowledge the help received from Olav Ligaarden, in performing the systematic literature review.
REFERENCES


APPENDIX A: THE LITERATURE REVIEW

This section presents the steps of a literature review we conducted prior to the identification and evaluation of the potentially useful approaches for uncertainty handling. Firstly, we performed a mapping study in order to gain an overview of the main topics and possible keywords. Secondly, we performed a systematic literature review in order to reveal state of the art based on a rigorous walkthrough of the available publications. The remainder of this section presents the process and the results of the mapping study and the systematic literature review, respectively.

The mapping study

The pre-review mapping study was performed in February 2010, to help in scoping the research question and to get a preliminary overview of the existing uncertainty handling approaches. The mapping study involved a search in ACM Digital Library, IEEE Xplore, Lecture Notes in Computer Science and Google Scholar with keywords such as "uncertainty", "inaccuracy", "imprecise", "predictability" and "predictability domain". The keywords applied were general upon the beginning, and then more focused when the follow-up papers were recorded. No constraints on the time since publishing were used on the resources acquired. About fifty publications from mathematics, statistics and computer science which presented either single or combined approaches for uncertainty handling, were extracted. The ones addressing customization of identified approaches for the different domains and the ones focusing on mainly aleatory uncertainty, were excluded, thus reducing the search results to about 33 publications. Moreover, uncertainty and inaccuracy handling in other domains, were searched for in the same resources, by adding the keywords using as: "meteorology", "weather forecast", "effort estimation", "uncertainty" and "inaccuracy" individually and in combination. About ten additional publications were recorded. A walkthrough of the search results was made. The mapping study resulted in 38 articles addressing the uncertainty handling approaches relevant in the context of system quality prediction based on the weighted dependency trees.

The systematic literature review

For the systematic literature review we followed a procedure recommended by (Kitchenham, et al., 2007). The systematic literature review was performed in March and April 2010. The first phase of planning a systematic literature review is to specify the research objective that should be satisfied by conducting the review. The objective of this systematic literature review is to identify and assess the potentially useful approaches which:

1. are practically applicable for supporting uncertainty handling in the context of WDT-based analysis,
2. have a well defined propagation within the WDT, and
3. handle the epistemic uncertainty.

By an approach we mean the collection of the formal and the graphical means available along with the procedure prescribed for applying those means.

The second step of planning a systematic literature review is to develop the search strategy, as a part of developing the review protocol. As the search specified below captures a wide range of publications, we developed a stopping heuristic and applied the following protocol:

1. The researchers clarify the objective and make sure that the background of the study, the research question and the research method are understood.
2. For each of the search strings, run the search on all five search engines.
3. Researcher A and Researcher B independently record and collect the bibliographic information for any source not previously recorded that appeared to be relevant, based on title and type of publication.
4. Continue through the search results until reaching 10 consecutive, irrelevant results.
5. Researchers record each other’s result lists from steps 3 and 4, and vote on the relevant titles based on the objective and the research question.

6. Consolidate the results obtained by Researcher A and Researcher B, from step 5.

7. Researcher A and Researcher B independently record the abstracts of the titles collected in step 6, and vote on the relevance (by assigning binary votes: relevant or not relevant) of each abstract.

8. Consolidate the results obtained by Researcher A and Researcher B, from step 5.

9. Researcher A reviews full texts of the selected publications, based on the criteria (presented in the following section).

In step 2 of the protocol we used the following search engines to conduct our review:

1. Google scholar
2. IEEE explore
3. Science direct
4. ACM digital library
5. ACM guide

The keywords used were deduced from the research objective. Since the different search engines had different search categories, kinds of possible constraints and various degrees of support for logical keyword compositions, the keyword sets and constraints used in step 2 varied as specified below.

Two sets of keywords were searched on Google scholar:

1. (uncertainty OR inaccuracy OR imprecise OR imprecision OR uncertain OR inaccurate) (system OR software OR meteorology OR “weather forecasting” OR “effort estimation”) (prediction OR simulation OR model OR models OR modeling OR modelling)
2. Uncertainty (prediction OR simulation OR model OR models OR modeling OR modelling) (software OR system) (tree OR graph OR DAG)

Each search was constrained to articles excluding patents with at least summaries, published since year 2000. The first search resulted in 796000 items, and the second one in 223000 items.

Two sets of keywords were searched on IEEE explore:

1. (uncertainty OR inaccuracy OR imprecise OR imprecision OR uncertain OR inaccurate) AND (system OR software OR meteorology OR “weather forecasting” OR “effort estimation”) AND (prediction OR simulation OR model OR models OR modeling OR modelling))
2. Uncertainty (prediction OR simulation OR model OR models OR modeling OR modelling) AND (software OR system) AND (tree OR graph OR DAG))

Each search was constrained to conferences and journals as types of content within the subject defined as "Computing and Processing (Hardware/Software)", published between years 2000 and 2010. The first search resulted in 329212 items, and the second one in 290038 items.

Two sets of keywords were searched on Science direct:

1. (uncertainty OR inaccuracy OR imprecise OR imprecision OR uncertain OR inaccurate) AND (system OR software OR meteorology OR “weather forecasting” OR “effort estimation”) AND (prediction OR simulation OR model OR models OR modeling OR modelling))
2. Uncertainty (prediction OR simulation OR model OR models OR modeling OR modelling) AND (software OR system) AND (tree OR graph OR DAG))

Each search was constrained to the computer science category published since 1999. The first search resulted in 3629 items, and the second one in 5631 items.
<table>
<thead>
<tr>
<th>Search engine</th>
<th>Nr. of results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google scholar, keyword set 1</td>
<td>496 000</td>
</tr>
<tr>
<td>Google scholar, keyword set 2</td>
<td>223 000</td>
</tr>
<tr>
<td>IEEE explore, keyword set 1</td>
<td>329 212</td>
</tr>
<tr>
<td>IEEE explore, keyword set 2</td>
<td>290 038</td>
</tr>
<tr>
<td>Science direct, keyword set 1</td>
<td>3 629</td>
</tr>
<tr>
<td>Science direct, keyword set 2</td>
<td>5 631</td>
</tr>
<tr>
<td>ACM digital library, keyword set 1</td>
<td>607</td>
</tr>
<tr>
<td>ACM digital library, keyword set 2</td>
<td>1 232</td>
</tr>
<tr>
<td>ACM guide, keyword set 1</td>
<td>2 879</td>
</tr>
<tr>
<td>ACM guide, keyword set 2</td>
<td>8 462</td>
</tr>
</tbody>
</table>

Researcher A records and collects titles

Researcher B records and collects titles

Researcher A votes among the titles

Researcher B votes among the titles

Collect all relevant titles

Collect all relevant abstracts

Researcher A votes on the abstracts

Researcher B votes on the abstracts

Collect all relevant abstracts

Researcher A records full texts and votes on them

Researcher B records follow-ups from the full texts and votes on them

Collect all relevant full texts

Researcher A extracts the relevant info about the potentially useful methods for uncertainty handling

Figure 8: The process applied in the literature review
Two sets of keywords were searched on both ACM digital library and ACM guide:
1. uncert* software*
2. uncert* model*

The stars in the keywords allow for arbitrary ending of the keyword. Each search was constrained to items published since 2000. The search in ACM guide with the first and the second keyword set resulted in 607 and 1232 items, respectively. The search in ACM digital library with the first and the second keyword set resulted in 2879 and 8462 items, respectively.

The entire process is sketched on Figure 8. Since the search engines used sort the results by relevance, this procedure was considered to be effective at producing search results. In most cases, the density of relevant titles steadily decreased as we examined each list of the search results.

In the first pass (step 3 of the protocol), the researchers were as inclusive as possible, since the filtering was only based on titles and the types of the publications. All kinds of publications except sole abstracts, books, presentations and white papers dating no more that ten years, were included. The procedure resulted in the data presented in Table 11. The reviewed titles and the passed titles columns come from executing the steps 3 and 4 of the protocol. The reviewed titles column reports on the sequence number on the search result list, of the last title. The fifth column of Table 11 indicates which researcher the results from columns 3 and 4 were obtained by.

<table>
<thead>
<tr>
<th>Search engine</th>
<th>Keyword set</th>
<th>Reviewed titles</th>
<th>Passed titles</th>
<th>Researcher</th>
<th>Nr of the relevant titles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google scholar</td>
<td>1</td>
<td>51</td>
<td>14</td>
<td>A</td>
<td>9</td>
</tr>
<tr>
<td>Google scholar</td>
<td>2</td>
<td>38</td>
<td>6</td>
<td>A</td>
<td>5</td>
</tr>
<tr>
<td>IEEE explore</td>
<td>1</td>
<td>15</td>
<td>11</td>
<td>A</td>
<td>11</td>
</tr>
<tr>
<td>IEEE explore</td>
<td>2</td>
<td>44</td>
<td>6</td>
<td>A</td>
<td>6</td>
</tr>
<tr>
<td>Science direct</td>
<td>1</td>
<td>20</td>
<td>6</td>
<td>A</td>
<td>6</td>
</tr>
<tr>
<td>Science direct</td>
<td>2</td>
<td>34</td>
<td>4</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>ACM digital library</td>
<td>1</td>
<td>17</td>
<td>6</td>
<td>A</td>
<td>5</td>
</tr>
<tr>
<td>ACM digital library</td>
<td>2</td>
<td>18</td>
<td>10</td>
<td>A</td>
<td>7</td>
</tr>
<tr>
<td>ACM guide</td>
<td>1</td>
<td>19</td>
<td>13</td>
<td>A</td>
<td>8</td>
</tr>
<tr>
<td>ACM guide</td>
<td>2</td>
<td>19</td>
<td>15</td>
<td>A</td>
<td>13</td>
</tr>
<tr>
<td>Google scholar</td>
<td>1</td>
<td>26</td>
<td>9</td>
<td>B</td>
<td>6</td>
</tr>
<tr>
<td>Google scholar</td>
<td>2</td>
<td>26</td>
<td>6</td>
<td>B</td>
<td>5</td>
</tr>
<tr>
<td>IEEE explore</td>
<td>1</td>
<td>41</td>
<td>15</td>
<td>B</td>
<td>13</td>
</tr>
<tr>
<td>IEEE explore</td>
<td>2</td>
<td>35</td>
<td>10</td>
<td>B</td>
<td>10</td>
</tr>
<tr>
<td>Science direct</td>
<td>1</td>
<td>12</td>
<td>4</td>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>Science direct</td>
<td>2</td>
<td>9</td>
<td>2</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>ACM digital library</td>
<td>1</td>
<td>12</td>
<td>2</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>ACM digital library</td>
<td>2</td>
<td>36</td>
<td>7</td>
<td>B</td>
<td>7</td>
</tr>
<tr>
<td>ACM guide</td>
<td>1</td>
<td>23</td>
<td>6</td>
<td>B</td>
<td>6</td>
</tr>
<tr>
<td>ACM guide</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>B</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 11: Summary of the title votes

Next, the researchers voted on the relevance of the titles (step 5 of the protocol). Researcher A voted on whether the results found by Researcher B were relevant to the review, and vice versa. Researcher A collected 91 titles, of which Researcher B accepted 64. Researcher B collected 61 sources, of which Researcher A accepted 55. A total of 89 (not the sum of the two due to the overlaps) unique titles from the
voting confirmation was accepted to proceed to the next phase of the review. The last column of Table 11 reports on the number of the titles deemed relevant by the other researcher during step 5 of the protocol.

Step 6 was done by collecting all titles assessed as relevant by at least one researcher. The 89 unique abstracts of the publications that had passed the step 6 of the protocol were recorded by each researcher separately. Each researcher voted whether the abstract was relevant or not. Researcher A voted yes to 22 abstracts and Researcher B voted yes to 9 abstracts. The abstracts that at least one researcher deemed relevant, were collected. This resulted in 24 (not the sum of the two due to the overlaps) relevant abstracts.

Step 8 was done by collecting all abstracts assessed as relevant by at least one researcher. The full texts that had passed step 8 of the protocol were recorded by Researcher A and evaluated for the relevance and applicability in the context of system quality prediction based on the weighted trees. A total of 13 texts were found relevant. Additionally, 11 distinct follow-up articles (referenced from the recorded full texts) were recorded and 4 of them were found relevant. The total number of the unique full texts evaluated as relevant during the mapping study and the systematic literature review was 42. Thus, there were 13 overlapping articles between the mapping study and the systematic literature review, and 4 of the overall 29, were found through the systematic literature review, while 25 were found through the mapping study. 19 of the 25 relevant articles discovered only through the mapping study were published before 2000 – the timeframe which was discluded from the searches during the systematic literature review.

MS Excel sheets were found convenient and used for documenting the votes, counting the results and identifying the overlapping titles.
APPENDIX B: DEDUCTION OF THE EVALUATION CRITERIA

This section presents in detail the deduction of the evaluation criteria specified in tables two through nine. The systematic literature review provided, in addition to a preliminary evaluation, an initial list of the evaluation criteria. Based on the results of the systematic literature review and the practical experiences, the evaluation criteria were iteratively refined through multiple discussions among the authors. The deduction of the criteria was partially guided by an existing taxonomy for system acceptability. Thus, the original list of criteria was incrementally refined during the study.

The objective of the systematic literature review has been decomposed into three sub-objectives specified in Table 1. These sub-objectives are specified and decomposed into aspects. Each aspect implies one or more evaluation criteria. Each criterion was then defined and specified in detail, further decomposing it into a set of distinct categories of fulfillment.

The taxonomy for system acceptability proposed in (Nielsen, 1993) has been adapted to our objective, by mapping the sub-objectives of the systematic literature review to the relevant parts of the taxonomy. Table 12 shows the structure of the model by Nielsen, which we were guided by towards a refined definition of the evaluation criteria. The nodes contained in the model by Nielsen are presented in the form of italic (bold and non-bold) text. The bold italic text represents the adopted ones, while the overall elements from the table which are written in italic are contained in the model by Nielsen but not needed for our objective. In addition, Table 12 shows the aspects we have found necessary to add in order to fully cover the objective specified in Table 1. The added aspects are presented in the form of non-italic text.

The aspects adopted from Nielsen or added are annotated with the name of the aspect of one of out three sub-objectives and the criteria (denoted as CX, where X is the number of the criterion) implied by the aspect. Thus, we only focus on two branches of the practical acceptability (usefulness and scalability).

The first sub-objective of the systematic literature is non-functional, while the remaining two are functional. Hence, utility (or functionality) in our context is:

- handling of epistemic uncertainty, and
- well defined propagation.

The practical applicability is de-composed into two aspects: scalability and usability. We consider that this influences usability:

- the complexity of the estimate representation form, and
- suitability of estimate representation for expressing both domain expert-judgment-based and measurement-based input.

The aspects identified are related to evaluation criteria. Seven evaluation criteria have been identified, which in conjunction cover the objective and form the basis for evaluation of the potentially useful approaches. Six of the criteria are related to the individual aspects, while the last criterion addresses the empirical evaluation of the approaches and therefore spreads across all the aspects of the three sub-objectives.

Assessing an uncertainty handling approach should not only be based on accuracy of estimates made, but on the practical applicability aspects discussed above. In the sequel we state two necessary prerequisites for the approaches evaluated and define a set of evaluation criteria in relation to the sub-objectives of the systematic literature review.

The epistemic uncertainty is better suited for possibilistic uncertainty representations, than the probabilistic ones. Possibility distributions can also straightforwardly accommodate linguistic information
on quantitative scales. A probabilistic modeling may introduce unwarranted information in the form of a
distribution function that is not sufficiently justified. Continuous probability distributions are also more
challenging to provide and interpret by the domain experts, and the precision expressed by the probability
distribution may be higher than what in reality is the case. We only focus on the epistemic uncertainty
associated with the empirical input which is acquired from the domain experts and from the
measurements. Moreover, practical applicability of the approach is one of the sub-objectives. Therefore,
two prerequisites for the approaches evaluated are identified:
- only discretized estimates or the estimates that can be discretized are considered, thus
  accommodating for a possibilistic approach; and
- the number of the prior estimates (that is, the estimated to be fed into the model before the
  propagation can be performed) that can be acquired is limited.

Both prerequisites are due to the conditions that the practical settings in which the approaches are to be
applied in, define. The prerequisites are motivated by the experiences from practical trials, in particular
development and application the models for prediction of system quality, in the form of WDTs
(Omerovic, et al., 2009).

<table>
<thead>
<tr>
<th>System acceptability</th>
<th>Practical acceptability</th>
<th>Usefulness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost, compatibility, reliability, etc.</td>
<td>Scalability (C3)</td>
<td></td>
</tr>
<tr>
<td>Easy to learn, easy to use, easy to remember, few errors, subjectively pleasing</td>
<td>Complexity of estimate representation form (C1)</td>
<td></td>
</tr>
<tr>
<td>Complexity of estimate representation form (C1)</td>
<td>Suitability of estimate representation form for expressing input (C2)</td>
<td></td>
</tr>
<tr>
<td>Handling epistemic uncertainty (C2, C4)</td>
<td>Well defined propagation (C5, C6)</td>
<td></td>
</tr>
<tr>
<td>Social acceptability</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 12: Decomposition of system acceptability, based on our objective and motivated by the model of the attributes of system acceptability provided in (Nielsen, 1993)

An uncertainty handling approach should facilitate provisioning of reliable estimates which are provided by domain experts and through other forms of data acquisition. Finding optimal balance between soundness and practical applicability is challenging due to the mutual interaction between the two. While for example precision (as an attribute of soundness) of the representation may improve efficiency, it may be difficult to understand, achieve and apply the higher precision in a practical setting. The criteria defined are therefore not mutually exclusive, and their interaction and importance may depend on the objective of the system analyzed and the context. Such a weighting of the criteria is however beyond the scope of this paper.

The objective for the uncertainty handling approaches has been decomposed into the three sub-objectives specified in Table 1. Each sub-objective is elaborated on below, by interpreting it, decomposing it to aspects, and refining the aspects in the form of evaluation criteria. Table 2 provides an overview of the evaluation criteria deduced in the sequel.
Sub-objective 1: practical applicability

The aspects of the practical applicability are first deduced by the authors, based on the experiences from development and application the models for prediction of impacts of architecture design changes on the system quality, in the form of WDTs (Omerovic, et al., 2009). Thereafter, the aspects are related to the relevant elements within the practical acceptability framework proposed in (Nielsen, 1993).

The practical applicability from the point of view of the domain experts implies two aspects:

1. the usability of the approach, and
2. the scalability of the approach.

The first aspect addresses whether the representation form of the quantitative estimates is suitable for the domain experts to relate to. In particular, complexity and suitability of the notions used by the approach are assessed. It is crucial that the formal representation of uncertainty is comprehensible to those who have in-depth system knowledge, but not necessarily a profound insight into the formal representation techniques. We assume that complexity of the quantitative notions influences comprehensibility. The approach should facilitate provisioning of the uncertain WDT estimates which are easy for domain experts to provide and interpret. Thus, for the first aspect of practical applicability we propose the following criteria:

- C1: complexity of the representation form for the quantitative estimates. The criterion is specified in Table 3.
- C2: suitability of the approach for both expert-judgment-based and measurement-based input. The criterion is specified in Table 4.

In order to be practically applicable on a realistic system, the approach has to scale with respect to the number of prior estimates that need to be provided in order to create a WDT and before any propagation can start. Thus, for the second aspect of practical applicability we propose criterion C3:

- C3: scalability with respect to number of estimates needed on a WDT. The criterion is specified in Table 5.

Sub-objective 2: handling of the epistemic uncertainty

This sub-objective is rooted in the need for addressing the deterministic systems and involves explicitly expressing the epistemic uncertainty. The explicit expression involves precisely and correctly representing the quantities of the uncertain estimates. The notions used by the approach have to be related to the empirical input and its representation. This implies again the above specified criterion C2.

How fine grained the expressions of the quantitative estimate and their uncertainty may be, is an important property of the approach. In case the precision is constrained, information about the actual degree of uncertainty may be lost. Each individual parameter estimate and its associated uncertainty needs to be represented in as a precise form as possible to provide by the domain experts or the other sources for data acquisition. The right granularity of the uncertain estimate representation at the level of each parameter is needed, thus we propose criterion C4: precision allowed by the representation form for the quantitative estimates. The criterion is specified in Table 6.

Sub-objective 3: well defined propagation

The sub-objective of a well defined propagation addresses the complexity and expressiveness of the propagation techniques provided by the approach. The values on the leaf nodes and arcs of the WDTs have to be propagated according to an applicable technique which is suitable for reasoning on dependencies regarding both expert-judgment-based estimates and measurement-based input.

Thus, we propose the following two criteria:
• C5: complexity of the propagation, and
• C6: the expressiveness of the arithmetic operations for propagation.

The criteria C5 and C6 are specified in Table 7 and Table 8, respectively.

**A general evaluation criterion**

The empirical evaluations provide input on the performance of the approach with respect to all three sub-objectives and the entire objective in general. The existing empirical evaluations of the approach will give an indication of its general applicability in an industrial setting on a realistic system. Thus, this general aspect implies criterion C7: empirical evaluation of the approach, as reported in the publications found through the literature review. The criterion is specified in Table 9.
APPENDIX C: THREATS TO VALIDITY AND RELIABILITY

The aim of this section is to address the main issues regarding the validity and the reliability of this evaluation. We address the research method, its effects on the evaluation and the validity of the main findings. In addition to reliability threats, four types of validity threats presented in (Hyman, 1982) are addressed below: conclusion validity, construct validity, internal validity and external validity.

Overall, the main threats to validity are:
1. publication bias - there might be unpublished negative experiences with different approaches - this issue was not possible to mitigate, but we do not consider it to represent a problem,
2. selection of articles – there might be other which are not revealed through the literature review or not included in the study,
3. data extraction, and
4. identification of criteria.

The high-level evaluation is based on the detailed information extraction of the approaches from the publications found through the literature review. The high-level evaluation presents exclusively the properties reported within the relevant literature obtained through the mapping study and the literature review. The information otherwise available is not taken into account. The quality of the results is therefore dependent on the thoroughness of the research method. Inspite of the structured process during the literature review, the contextual factors are difficult to fully eliminate. One should therefore view the high-level evaluation within the frames of and the quality of the underlying research method, particularly the literature review and the data extraction.

The low-level evaluation is however based on the instantiations in the WDTs and the discussion of the findings from the high-level evaluation, with respect to the evaluation criteria. The instantiation is based on the presentations of the respective approaches, provided in the literature obtained. Still, the low-level evaluation is to a much lesser extent bound to the literature review.

Reliability is concerned with demonstrating that the study can be repeated with the same results. Two of the factors influencing the reliability of the literature review are: the subjective influence of the researcher A on the mapping study and the step 9 of the protocol. These steps were only performed by one researcher. During the overall steps of the protocol, the consensus in any deviations was made by consolidating all the results, thus a third voter was unnecessary. In addition, the information extraction and the instantiation are, although systematic and based on the evaluation criteria, exposed to minor subjective judgments. However, since the evaluation is driven by the criteria which are clearly defined, the bias should be minimal. The summary provided in Table 10 involved negligible doubt when assigning the criteria categories to the individual approaches.

The main threats to validity for this study are publication selection bias, inaccuracy in data extraction, and misclassification. To help ensure an unbiased selection process, we defined the objective of the evaluation in advance, organized the selection of articles as a multistage process, and involved two researchers in the literature review. Still, the process was complex and we may not have managed to detect all the publications that are relevant for inclusion. Moreover, data extraction from prose is difficult due to lack of both standard terminology and standards for reporting the properties of the approaches. This may have resulted in some inaccuracy in the data. Another challenge was that there is no up to date keyword standard that we are aware of that exactly captures the topics relevant for this study.

Moreover, except from the framework in (Nielsen, 1993) which we used to deduce the evaluation criteria, we have not revealed any up to date standard for evaluation criteria or generally accepted methodology for evaluation of approaches. Hence, the criteria deduced are, apart from the evaluation itself, another one
of the contributions of the paper. The variation of the assessments in the evaluation summary indicated that the criteria and their respective categories are capable of differentiating between the approaches at intended detail level.

Construct validity concerns whether we measure what we believe we measure. In this context, construct validity is the traceability of the conjunction of the evaluation criteria to the overall objective. This aspect has been addressed during the deduction of the criteria.

Conclusion validity concerns the composition of participants and the statistical analysis. In this context, conclusion validity is the breadth and quality of the literature review. Given the total number of publications the literature review has covered, we have reason to believe in strength of the conclusion validity. The subjective aspects of the literature review and particularly the parts performed by only one researcher present the major threats to the conclusion validity.

Internal validity concerns matters that may affect the causality of an independent variable, without the knowledge of the researcher. In this context, internal validity is the traceability of the evaluation itself to the respective criteria, that is, whether we actually provide the evaluation asked for by the criteria. Since the evaluation is purely based on the literature uncovered during the literature review and since the criteria are defined with distinct categories, the threats to the internal validity have been addressed.

External validity concerns the generalization of findings of this case study to other contexts and environments. The evaluation is limited to the systems where the epistemic uncertainty is the dominating one. The epistemic uncertainty is the crucial one in the context of the deterministic systems and therefore given most attention. (Baraldi, et al., 2008) argues about the limitations associated to a probabilistic representation of epistemic uncertainty under limited information. Being of a discrete and non-stochastic nature, the epistemic uncertainty should be handled by a possibilistic approach. Hence, we have only addressed the epistemic uncertainty. This is however a simplification, since the aleatory uncertainty is present mainly due to the usage profile. Ideally, both the epistemic and the aleatory uncertainty should be addressed by a hybrid approach. Such an approach would handle the epistemic and the aleatory uncertainty by possibilistic and probabilistic approaches respectively, and by weighing the two types of uncertainty in a system. It would however need to be much more complex, which would compromise its usability.
KEY TERMS AND DEFINITIONS

**Bayesian network**: a probabilistic graphical model that represents a set of random variables and their conditional independencies via a directed acyclic graph.

**Dependency tree**: a finite tree in which: each node may hold one or more quantitative values or a function, each arc may hold one or more a quantitative values or a function, and each inferred non-leaf node is a tree-specific function of its immediate children and their connecting arcs.

**Dependency view**: a dependency tree in which each node holds a quantitative value representing degree of fulfillment of a system quality characteristic, each arc holds a quantitative value representing the estimated impact of the child node on the parent node, and each inferred non-leaf node is a linear function of its immediate children and their connecting arcs.

**Fuzzy logic**: a form of multi-valued logic derived from fuzzy set theory to deal with reasoning that is approximate. In contrast with "crisp logic", where binary sets have binary logic, fuzzy logic variables may have a truth value that ranges between 0 and 1 and is not constrained to the two truth values of classic propositional logic.

**Inaccuracy**: the condition of being imprecise or insufficiently exact.

**Intervals**: a set containing all points (or all real numbers) between two given endpoints.

**Modeling**: representing, designing and analyzing a representation of a system to study the effect of changes to system variables.

**Simulation**: the imitation of some real thing, state of affairs, or process. The act of simulating something generally entails representing certain key characteristics or behaviours of a selected physical or abstract system.

**Subjective logic**: a type of probabilistic logic that explicitly takes uncertainty and belief ownership into account.

**Uncertainty**: the condition in which reasonable knowledge or information regarding the present state or the future is not available.
Chapter 12

Paper 4: Evaluation of Experiences from Applying the PREDIQT Method in an Industrial Case Study
We have developed a method called PREDIQT for model-based prediction of impacts of architectural design changes on system quality. A recent case study indicated feasibility of the PREDIQT method when applied on a real-life industrial system. This paper reports on the experiences from applying the PREDIQT method in a second and more recent case study – on an industrial ICT system from another domain and with a number of different system characteristics, compared with the previous case study. The analysis is performed in a fully realistic setting. The system analyzed is a critical and complex expert system used for management and support of numerous working processes. The system is subject to frequent changes of varying type and extent. The objective of the case study has been to perform an additional and more structured evaluation of the PREDIQT method and assess its performance with respect to a set of success criteria. The evaluation argues that: 1) the PREDIQT-based analysis facilitates predictions providing sufficient understanding of the impacts of architectural design changes on system quality characteristics, so that informed decisions can be made; 2) the PREDIQT-based analysis is cost-effective; 3) the prediction models are sufficiently expressive to adopt the relevant architectural design changes and analyze their effects on quality; 4) the prediction models are sufficiently comprehensive to allow the domain experts to be actively involved in all phases of the PREDIQT process and achieve the goals of each phase with a common understanding of the results; and 5) the PREDIQT-based analysis facilitates knowledge management and contributes to a common understanding of the target system and its quality. Moreover, the study has provided useful insights into the weaknesses of the method and suggested directions for future research and improvements.

**Key words:** Quality prediction, System architectural design, Change impact analysis, Modeling, Simulation

<table>
<thead>
<tr>
<th>KEYWORDS</th>
<th>ENGLISH</th>
<th>NORWEGIAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>GROUP 1</td>
<td>Quality prediction, System architectural design, Empirical evaluation</td>
<td>Kvalitetsprediksjon, Systemarkitektur, Empirisk evaluering</td>
</tr>
<tr>
<td>GROUP 2</td>
<td>Modeling, Change impact analysis</td>
<td>Modellering, Konsekvensanalyse</td>
</tr>
<tr>
<td>SELECTED BY AUTHOR</td>
<td>Quality prediction, Architectural design</td>
<td>Kvalitetsprediksjon, Systemarkitektur</td>
</tr>
</tbody>
</table>
Evaluation of Experiences from Applying the PREDIQT Method in an Industrial Case Study

Aida Omerovic\textsuperscript{1,2}, Bjørnar Solhaug\textsuperscript{1} and Ketil Stølen\textsuperscript{1,2}
\textsuperscript{1}\textsuperscript{1}SINTEF ICT, P.O. Box 124, 0314 Oslo, Norway
\textsuperscript{2}\textsuperscript{2}University of Oslo, Department of Informatics, P.O. Box 1080, 0316 Oslo, Norway
Email: \{aida.omerovic, bjornar.solhaug, ketil.stolen\}@sintef.no

Abstract—We have developed a method called PREDIQT for model-based prediction of impacts of architectural design changes on system quality. A recent case study indicated feasibility of the PREDIQT method when applied on a real-life industrial system. This paper reports on the experiences from applying the PREDIQT method in a second and more recent case study – on an industrial ICT system from another domain and with a number of different system characteristics, compared with the previous case study. The analysis is performed in a fully realistic setting. The system analyzed is a critical and complex expert system used for management and support of numerous working processes. The system is subject to frequent changes of varying type and extent. The objective of the case study has been to perform an additional and more structured evaluation of the PREDIQT method and assess its performance with respect to a set of success criteria. The evaluation argues that: 1) the PREDIQT-based analysis facilitates predictions providing sufficient understanding of the impacts of architectural design changes on system quality characteristics, so that informed decisions can be made; 2) the PREDIQT-based analysis is cost-effective; 3) the prediction models are sufficiently expressive to adopt the relevant architectural design changes and analyze their effects on quality; 4) the prediction models are sufficiently comprehensible to allow the domain experts to be actively involved in all phases of the PREDIQT process and achieve the goals of each phase with a common understanding of the results; and 5) the PREDIQT-based analysis facilitates knowledge management and contributes to a common understanding of the target system and its quality. Moreover, the study has provided useful insights into the weaknesses of the method and suggested directions for future research and improvements.

Index Terms—Quality prediction, System architectural design, Change impact analysis, Modeling, Simulation.

I. INTRODUCTION

When adapting a system to new usage patterns, processes or technologies, it is necessary to foresee the implications that the architectural design changes have on system quality. Predictability with respect to non-functional requirements is one of the necessary conditions for the trustworthiness of a system. Examination of quality outcomes through implementation of the different architecture design alternatives is often unfeasible. A model-based approach is then an alternative. We have developed a method called PREDIQT with the aim to facilitate model-based prediction of impacts of architectural design changes on system quality. Examples of quality characteristics include availability, scalability, security and reliability.

A recent case study [16] indicated feasibility of the PREDIQT method when applied on a real-life industrial system. The promising empirical results and experiences from the previous case study encouraged further and more structured evaluation of the PREDIQT method. This paper addresses experiences from applying PREDIQT on another real-life industrial system from a different domain and with different system characteristics (lifetime, purpose, technology the system is implemented on, number of users and kind of users), compared to the previous case study.

The target system analyzed serves as a semantic model and a repository for representation of the system owner’s core working processes and rules, and as a knowledge database. It is a business-critical and complex expert system used for management and support of numerous working processes, involving hundreds of professional users every day. The system is subject to frequent architectural design changes of varying type and extent. The system owner, who was also the client commissioning the analysis, required full confidentiality with respect to the kind of system targeted, the models obtained, the personnel involved and the name of the organization. This paper reports solely on the experiences obtained by the participants of the real-life case, describes the process undergone, the evaluation results, the observations and the properties of the artifacts. The reported experiences and results have provided valuable insight into the strengths and weaknesses of the method.

The case study was conducted in the year 2010. The first overall two phases of the PREDIQT method were conducted in their entirety, while the last phase was partially covered. In addition, the method is assessed through a thought experiment based evaluation of predictions and a postmortem review. All prediction models were developed during the analysis and the entire target system (within the predefined scope) was analyzed. The analysis was performed in the form of five workshops and six intermediate meetings in a fully realistic setting in terms of the scope, the objectives, the process, the prediction models and the participants.

The rest of the paper is structured as follows. We briefly present the PREDIQT method in Section II. The research method is summarized in Section III. Section IV presents five success criteria, which cover the needs of the three stakeholder groups, and which both the case study and the contents of this paper have primarily been driven by. The process undergone during the PREDIQT-based analysis is presented in Section V. Results of evaluation and a postmortem review are summarized.
Phase 1: Target modeling
- Sub-phase 1: Characterization of the target and the objectives
- Sub-phase 2: Development of Quality Models
- Sub-phase 3: Mapping of Design Models
- Sub-phase 4: Development of Dependency Views

Phase 2: Verification of prediction models
- Sub-phase 1: Evaluation of prediction models
- Sub-phase 2: Fitting of prediction models
- Sub-phase 3: Approval of the final prediction models

Phase 3: Application of prediction models
- Sub-phase 1: Specification of a change
- Sub-phase 2: Application of the change on prediction models
- Sub-phase 3: Quality prediction

Fig. 1. A simplified overview of the process of the PREDIQT method

in Section VI. Section VII provides an evaluation of the experiences and results, with respect to the five pre-defined success criteria, before concluding in Section VIII. A more thorough presentation of the research method is provided in Appendix 1. Setup and data collection during the PREDIQT-based analysis are outlined in Appendix 2. The outcomes of the process, in terms of artifacts, evaluation results and observations, are reported in Appendix 3. Appendix 4 presents the design of the evaluation template used in relation to the postmortem review. A summary of the feedback received through the evaluation template is provided in Appendix 5. Threats to validity and reliability are discussed in Appendix 6. Appendix 7 summarizes some of the related work.

II. OVERVIEW OF THE PREDIQT METHOD

The PREDIQT method produces and applies a multi-layer model structure, called prediction models, which represent system relevant quality concepts (through “Quality Models”), architectural design (through “Design Models”), and the dependencies between architectural design and quality (through “Dependency Views”). The Design Models are used to specify the target system and the changes whose effects on quality are to be predicted. The Quality Models are used to formalize the quality notions and define their interpretations. The values and the dependencies modeled through the DVs are based on the definitions provided by the Quality Models. The DVs express the interplay between the system architectural design and the quality characteristics. Once a change is specified on the Design Models, the affected parts of the DVs are identified, and the effects of the change on the quality values are automatically propagated at the appropriate parts of the DV. This section briefly outlines the PREDIQT method in terms of the process and the artifacts. For further details on PREDIQT, see [16, 17, 19].

The process of the PREDIQT method consists of three overall phases. Each phase is decomposed into sub-phases, as illustrated in a simplified form by Figure 1. Based on the initial input, the stakeholders involved deduce a high level characterization of the target system, its scope and the objectives of the prediction analysis, by formulating the system boundaries, system context (including the operational profile), system lifetime and the extent (nature and rate) of design changes expected. Quality Models are created in the form of a tree, by defining quality notions with respect to the target system. The Quality Models represent a taxonomy with interpretations and formal definitions of system quality notions. The total quality of the system is decomposed into characteristics, sub-characteristics and quality indicators. The Design Models represent the architectural design of the system.

For each quality characteristic defined in the Quality Model, a quality characteristic specific Dependency View (DV) is deduced from the Design Models and the Quality Models of the system under analysis. This is done by modeling the dependencies of the architectural design with respect to the quality characteristic that the DV is dedicated to, in the form of multiple weighted and directed trees. A DV comprises two notions of parameters:

1. EI: Estimated degree of Impact between two nodes, and
2. QCF: degree of Quality Characteristic Fulfillment.

Each arc pointing from the node being influenced is annotated by a quantitative value of EI, and each node is annotated by a quantitative value of QCF.

Figure 2 shows an excerpt of an example DV with fictitious values. In the case of the Encryption node of Figure 2, the QCF value expresses the goodness of encryption with respect to the quality characteristic in question, e.g., security. A quality characteristic is defined by the underlying system specific Quality Models, which may for example be based on the ISO 9126 product quality standard [1]. A QCF value on a DV expresses to what degree the node (representing system part, concern or similar) is realised so that it, within its own domain, fulfills the quality characteristic. The QCF value is based on the formal definition of the quality characteristic (for the system under analysis), provided by the Quality Models. The EI value on an arc expresses the degree of impact of a child node (which the arc is directed to) on the parent node, or to what degree the parent node depends on the child node, with respect to the quality characteristic under consideration.

“Initial” or “prior” estimation of a DV involves providing QCF values to all leaf nodes, and EI values to all arcs. Input to the DV parameters may come in different forms (e.g., from domain expert judgments, experience factories, measurements, monitoring, logs, etc.), during the different phases of
the PREDIQT method. The DV parameters are assigned by providing the estimates on the arcs and the leaf nodes, and propagating them according to the general DV propagation algorithm. Consider for example the Data protection node on Figure 2 (denoting: DP: Data protection, E: Encryption, AT: Authentication, AAT: Authorization, and O:Other):

\[
QCF_{(DP)} = QCF_{(E)} \cdot EI_{(DP \rightarrow E)} + QCF_{(AT)} \cdot EI_{(DP \rightarrow AT)} + QCF_{(AAT)} \cdot EI_{(DP \rightarrow AAT)} + QCF_{(O)} \cdot EI_{(DP \rightarrow O)} \tag{Eq. 1}
\]

The DV based approach constrains the QCF of each node to range between 0 and 1, representing minimal and maximal characteristic fulfillment (within the domain of what is represented by the node), respectively. This constraint is ensured through the formal definition of the quality characteristic rating (provided in the Quality Models). The sum of EIs, each between 0 (no impact) and 1 (maximum impact), assigned to the arcs pointing to the immediate children must be 1 (for model completeness purpose). Moreover, all nodes having a common parent have to be orthogonal (independent). The dependent nodes are placed at different levels when structuring the tree, thus ensuring that the needed relations are shown at the same time as the tree structure is preserved.

The general DV propagation algorithm, exemplified by Eq. 1, is legitimate since each quality characteristic DV is complete, the EIs are normalized and the nodes having a common parent are orthogonal due to the structure. A DV is complete if each node which is decomposed, has children nodes which are independent and which together fully represent the relevant impacts on the parent node, with respect to the quality characteristic that the DV is dedicated to.

The rationale for the orthogonality is that the resulting DV structure is tree-formed and easy for the domain experts to relate to. This significantly simplifies the parametrization and limits the number of estimates required, since the number of interactions between the nodes is minimized. Although the orthogonality requirement puts additional demands on the DV structuring, it has shown to represent a significant advantage during the estimation.

The “Verification of prediction models” phase aims to validate the prediction models (with respect to the structure and the individual parameters), before they are applied. A measurement plan with the necessary statistical power is developed, describing what should be evaluated, when and how. Both system-as-is and change effects should be covered by the measurement plan. Model fitting is conducted in order to adjust the DV structure and the parameters, to the evaluation results. The objective of the “Approval of the final prediction models” sub-phase is to evaluate the prediction models as a whole and validate that they are complete, correct and mutually consistent after the fitting. If the deviation between the model and the new measurements is above the acceptable threshold after the fitting, the target modeling is re-initiated.

The “Application of prediction models” presupposes that the prediction models are approved. During this phase, a specified change is applied to the Design Models and the DVs, and its effects on the quality characteristics at the various abstraction levels are simulated on the respective DVs. The change specification should clearly state all deployment relevant facts, necessary for applying the change. The “Apply the change on prediction models” phase involves applying the specified architectural design change on the prediction models. When an architectural design change is applied on the Design Models, it is according to the definitions in the Quality Model, reflected to the relevant parts of the DV. Thereafter, the DV provides propagation paths and quantitative predictions of the new quality characteristic values, by propagating the change throughout the rest of each one of the modified DVs, based on the general DV propagation algorithm. We have earlier developed tool support [16] based on MS Excel for simulation and sensitivity analysis related to the DVs.

### III. RESEARCH METHOD

The research method is motivated by the guidelines for case study research provided by Yin [21]. A deductive approach is undertaken, where the already defined PREDIQT method is exposed to an empirical trial in the form of a case study. For more details on the research method, see Appendix 1.

The main stages of the research method are depicted by Figure 3. The case study design included characterization of research question, the units of analysis and the success criteria, as the main outcomes.

The PREDIQT-based analysis was performed by following the pre-defined process of the PREDIQT method. However, instead of performing predictions of effects of future changes during the last workshop (as specified by the PREDIQT process), we chose to demonstrate how prediction models can be applied by simulating the effects of reversal of a very large already implemented change. As such, the model application phase is not fully covered, but only demonstrated. The affected Design Model and DV elements were identified and their modified parameter values estimated by the domain experts. Thereafter, the simulation on the DVs was made by the analyst.

Additionally, in order to evaluate the predictions obtained, a thought experiment regarding the effect of the change on the root nodes of the respective DVs, was performed by the domain experts. Thus, this was a part of the method assessment. The overall assessment measures included: written feedback

---

**Case study design**

- The research question
- Units of analysis
- Success criteria

**PREDIQT-based analysis**

**Assessment**

- Evaluation of predictions
- Written feedback after the analysis
- Verbal feedback during the analysis
- Observations made during the analysis

**Evaluation with respect to the success criteria**

---

*Fig. 3. Main stages of the research method*
(based on an evaluation template) from the analysis participants (affiliated with the customer organization) provided upon completion of the analysis and the above mentioned thought experiment based evaluation, verbal feedback during the analysis from the analysis participants (affiliated with the customer organization), and observations made by the analyst during the analysis. Based on the results of the PREDIQT-based analysis and the assessment, an evaluation with respect to the evaluation criteria was provided.

The research question of this case study is: How does the PREDIQT method perform in a fully realistic setting and when applied on a system from a different domain than the previously evaluated one.

PREDIQT phase-specific and the PREDIQT analysis general propositions are deduced. The propositions are then merged into a set of main success criteria and related to the objectives of the different stakeholders.

The units of analysis are identified as:
- the prediction models developed during the analysis
- the predictions obtained in terms of propagation paths and the QCF values
- the process of the PREDIQT-based analysis
- the participants of the analysis

The contents of the paper have been authored by the research group, which the analyst is a part of. In an attempt to avoid bias in the interpretation of the results, emphasis has been put on neutrally presenting the factual results, rather than interpreting and analyzing them in detail. The paper has been approved by the customer organization, with the aim of ensuring that agreement on the facts presented is achieved, as well as that no confidential information has been disclosed.

IV. SUCCESS CRITERIA

Many concerns are relevant in evaluation of a method like PREDIQT. In order to efficiently cover most prevailing concerns, we start by identifying the stakeholder groups involved: the customers, the domain experts and the analyst. Success criteria (SC) are then deduced from the point of view of each stakeholder group. Note that the degree of relevance of each success criterion may vary between the stakeholder groups.

The customers are the ones needing, requesting and paying for the PREDIQT-based analysis. The customers are represented by decision makers, managers, system architects or personnel responsible for quality assurance. The customers are not necessarily involved in the the process of a PREDIQT analysis, but have interest in added value through enhanced decision making related to architectural design changes of the system and improved knowledge management in the organization. These two concerns should facilitate trustworthiness, reliability, usability and maintainability of the system. They should also decrease dependency of the organization on individuals with system or business critical knowledge.

For the customer of a PREDIQT-based analysis, the overall goal is to accomplish useful predictions. By useful predictions we mean predictions providing sufficient understanding of the impacts of the architectural design changes on system quality, so that informed decisions can be made. Hence,

**SC1**: The PREDIQT-based analysis facilitates predictions providing sufficient understanding of the impacts of architectural design changes on system quality characteristics, so that informed decisions can be made.

The customers’ objective is also to be able to justify the cost of the analysis, compared to the benefit from it. Hence,

**SC2**: The PREDIQT-based analysis is cost-effective.

The analyst is the one conducting the PREDIQT analysis and documenting the results. This implies that the analyst has expertise on PREDIQT, leads the process, fully understands and in some cases participates in development of the prediction models, and documents the results. The analyst does however not necessarily have expertise on the target system under analysis, but should understand it sufficiently and be capable of collecting and processing the input needed in order to manage the development of the prediction models.

One objective for the analyst is to successfully conduct and document the analysis within the frame of the limited resources allocated. This implies that the PREDIQT-based analysis should be sufficiently simple to be feasible within the allocated resources, while still providing the requested added value for the customer. These goals are however already expressed through SC1 and SC2. In addition, the analyst aims to capture through the prediction models the relevant knowledge, information and requirements on system architecture, usage profile, assumptions and constraints. This is crucial for ensuring quality of the model-based predictions in terms of both prediction certainty and the ability of the prediction models to handle the relevant architectural design changes. Hence,

**SC3**: The prediction models are sufficiently expressive to adopt the relevant architectural design changes and analyze their effects on quality.

The domain experts participate actively in all stages of the analysis. The domain experts may be represented by system architects, system users, developers, engineers, managers or experts in specific scientific areas that the system supports. The PREDIQT method should help the domain experts communicating their knowledge in such a way that the analysis results in correct and sufficiently detailed prediction models which the participants agree upon and have a harmonized understanding of. The prediction models should therefore be comprehensible by the domain experts when properly guided by the analyst. Hence,

**SC4**: The prediction models are sufficiently comprehensible to allow the domain experts to be actively involved in all phases of the PREDIQT process and achieve the goals of each phase with a common understanding of the results.

Moreover, both the customer and the domain experts share the objective of improved knowledge management in the organization. For both, this is motivated by the concerns of efficient knowledge exchange, improved understanding of the system, reduced dependency on individuals, as well as increased maintainability and reliability of the system (as a
result of a model-based decision support). Hence,

**SC5**: The PREDIQT-based analysis facilitates knowledge management and contributes to a common understanding of the target system and its quality.

V. OVERVIEW OF THE PROCESS UNDERGONE DURING THE PREDIQT-BASED ANALYSIS

This section focuses on the process of the PREDIQT-based analysis (see Figure 3). We chronologically outline the relevant events and meetings in terms of their contents, participation, preparation and the time spent.

The analysis took place during the year 2010. Two preliminary meetings were held between the customer representatives and the analyst. The preliminary meetings were spent for motivating the analysis and identifying the challenges which the analysis should address. Thereafter, the analysis was organized in the form of five workshops and six working sessions in between some of the workshops. The workshops gathered both the domain experts and the customer (managers), and aimed to report on the current results and reach a milestone which the management should be involved in. The intermediate working sessions gathered the domain experts and the analyst to work tightly together on a particular task as a prerequisite for the forthcoming workshop. Table I outlines the process of the analysis. The first column specifies the type of the meeting (PM: preliminary meeting, W: workshop and S: working session) followed by the sequence number with respect to the kind of meeting. Column two specifies the date of the meeting. The third column lists the participants (note that all managers and domain experts are affiliated with the customer organization, while the analyst and the secretary belong to an external research group). The fourth column describes the contents and achievements of the meeting. The fifth column specifies the preparation activities for the meeting in question. The last column shows the approximate time spent (in terms of man-hours) during the meeting and in preparing for it. $T$ denotes the total number of hours spent by all participants of the meeting (including the analyst), while $A$ denotes the number of hours spent by the analyst only. The time spent on reporting and dissemination of the results after completion of meeting W5, is not included in the last column. At the end of each workshop (W), the approximate time until the next meeting was agreed upon, and the action points summarized. Minutes from each workshop were written and disseminated by the secretary. The sub-phases of the pre-defined process of the PREDIQT method (see Figure 1) were organized as follows:

- **Target modeling**
  - Characterize the target and the objectives: PM1, PM2, W1, S1
  - Create Quality Models: W2, S2, W3
  - Map Design Models: W2, S2, W3
  - Create DVs: S3, S4, S5
- **Verification of the prediction models**
  - Evaluation of prediction models: S5, W4, S6
- **Fitting of the prediction models:** W4, S6
- **Approval of the final prediction models:** S6
- **Application of the prediction models**
  - Specify the change: S6
  - Apply the change on the prediction models: W5
  - Quality prediction: W5

The case study was conducted in a realistic setting, with the objective of fully testing the feasibility of the method and providing added value for the customer. The target system analyzed serves within the customer organization as a semantic model and a repository for representation of the organization’s core working processes, rules, and as a knowledge database. It is a business-critical expert system used for management and support of numerous working processes, involving hundreds of professional users every day. It is developed in-house, is rather complex and is used by numerous surrounding systems. The system represents an important asset for the customer organization. The changes on the system are implemented collectively approximately two times a year, while the individual changes are considered and designed frequently. The extent and number of changes are increasing. There is a requirement on the time to market of certain types of changes. The system and the associated semantic model are complex and it is therefore very hard to test all effects of changes (i.e. the cost of testing becomes an increasing problem). Alternative or complementing methods for testing are therefore desirable. For instance, prediction of change impacts can potentially be used to tune testing.

Different departments of the customer organization are responsible for the operation and the development of the system, respectively. The change planning and deployment (including adjustments and extensions) is, based on standardized procedures and tools, undertaken by domain experts of the department which is in charge of development of the system. The procedures for initiating, evaluating and carrying out the changes are well defined within the customer organization. The PREDIQT analysis is initiated by the organization’s research department, on behalf of the overall stakeholders. Thus, the diversity of the participants and stakeholders (in terms of expertise, affiliation, interest, roles and background) is evident.

VI. ASSESSMENT

This section reports on the assessment part of the research method, depicted by Figure 3. Evaluation of the predictions based on a thought experiment is presented first. Secondly, the written feedback provided by the analysis participants from the customer organization upon completion of the above mentioned evaluation, is summarized. The written feedback is also referred to as a postmortem review. The third subsection reports on the verbal feedback provided, during the study, by the analysis participants from the customer organization. Lastly, the experiences and observations made by the analyst during the case study, are summarized.
<table>
<thead>
<tr>
<th>Meeting</th>
<th>Date</th>
<th>Participants</th>
<th>Contents</th>
<th>Preparation</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM1</td>
<td>March 25</td>
<td>Two managers. The analyst.</td>
<td>Customer’s presentation of the needs and challenges regarding quality, particularly security and interoperability of the systems. A brief presentation of the PREDIQT method and its possible application in the case study. Planning of the forthcoming meeting with the domain experts and the overall customer representatives.</td>
<td>Clarified formalities regarding communication channels and information exchange.</td>
<td>T:5</td>
</tr>
<tr>
<td>PM2</td>
<td>May 11</td>
<td>Four managers. Three domain experts. The analyst.</td>
<td>Characterization (by the customer organization representatives) of the system architecture and main challenges that the case study may focus on. A presentation of the PREDIQT method and its possible application to the context.</td>
<td>The analyst received the input requested: system and enterprise architecture documentation, requirements specification, system design documentation, service level agreement and operational environment specification.</td>
<td>T:10</td>
</tr>
<tr>
<td>W1</td>
<td>June 15</td>
<td>Three managers. Three domain experts. The analyst. The secretary.</td>
<td>The customer organization representatives characterized the target and the scope of the analysis: defined the target, defined the operational profile (current variability and expected changes in usage pattern, number of users, number of requests and amount of data), defined the expected lifetime of the system, specified type and extent of the expected changes, and characterized the main quality characteristics of the system.</td>
<td>The documentation studied by the analyst and clarifications or additional information needs communicated with the customer.</td>
<td>T:15</td>
</tr>
<tr>
<td>S1</td>
<td>June 17</td>
<td>Two domain experts. The analyst.</td>
<td>Given to the analyst by the domain experts: a demo of the target system, a presentation of the functional properties of the system, specification of typical faults and failures due to changes of the system, and an overview of the testing procedures. Clarifications of the written input.</td>
<td>The analyst specified questions and additional information needs to the domain experts.</td>
<td>T:10</td>
</tr>
<tr>
<td>W2</td>
<td>Aug. 17</td>
<td>Two domain experts. Three managers. The analyst. The secretary.</td>
<td>The analyst presented initial Quality Models (compliant with ISO 9126 [1]) and Design Models. Model revision in the group.</td>
<td>The analyst requested and received further documentation regarding system design. Development of system Quality Models and Design Models, by the analyst.</td>
<td>T:30</td>
</tr>
<tr>
<td>S2</td>
<td>Sept. 6</td>
<td>Three domain experts. The analyst.</td>
<td>The analyst presented the updated Quality Models and Design Models. Selected use scenarios and change cases were undergone in the group, in order to check if the current models support their specification. Revision of all quality and Design Models in the group.</td>
<td>Updates (based on the discussion from W2 meeting) of system Quality Models and Design Models, by the analyst.</td>
<td>T:15</td>
</tr>
<tr>
<td>W3</td>
<td>Sept. 9</td>
<td>Two domain experts. Three managers. The analyst. The secretary.</td>
<td>The analyst presented the current version of all prediction models. Revision of the Quality Models. Characterization of the types of potential architectural design changes. Preliminary approval of the available prediction models (Quality Models and Design Models).</td>
<td>Updates (based on the discussion from S2 meeting) of system Quality Models and Design Models, by the analyst.</td>
<td>T:20</td>
</tr>
<tr>
<td>S3</td>
<td>Sept. 28</td>
<td>Four domain experts. The analyst.</td>
<td>The analyst presented the approach regarding the DV structure development (assumptions, rules, DV syntax and DV semantics) and an early draft of a DV, for the domain experts. Development of the DV structures in the group.</td>
<td>Development of an initial draft of a DV structure (by the analyst), for triggering the discussion and exemplification.</td>
<td>T:20</td>
</tr>
<tr>
<td>S4</td>
<td>Sept. 29</td>
<td>Four domain experts. The analyst.</td>
<td>The analyst presented the approach regarding the (DV) parameter estimation (how to deduce the values, how to use the Quality Models, syntax and semantics of QCFs and ELs [16]), for the domain experts. Further development of the DV structures and DV parameter estimation in the group.</td>
<td>Documentation of the DV structure in the tool (MS Excel sheet customized for DVs in PREDIQT analysis). The analyst received documentation on typical system changes.</td>
<td>T:20</td>
</tr>
<tr>
<td>S5</td>
<td>Oct. 11</td>
<td>Four domain experts. The analyst.</td>
<td>Further DV parameter estimation.</td>
<td>Documentation of the updated DVs in the tool.</td>
<td>T:15</td>
</tr>
<tr>
<td>W4</td>
<td>Oct. 20</td>
<td>Three domain experts. One manager. The analyst. The secretary.</td>
<td>Validation of the DVs based on a thought experiment addressing randomly selected parts of the DVs. Model fitting of the DVs.</td>
<td>The analyst prepared a thought experiment setup based on the changes that the system has undergone.</td>
<td>T:20</td>
</tr>
<tr>
<td>S6</td>
<td>Oct. 22</td>
<td>Two domain experts. The analyst. The secretary.</td>
<td>Continued validation of the DVs based on a thought experiment of addressing randomly selected parts of the DVs. Model fitting of the DVs. Final approval of the prediction models. Specification of changes which are to be simulated in the demo of meeting W5.</td>
<td></td>
<td>T:15</td>
</tr>
<tr>
<td>W5</td>
<td>Nov. 3</td>
<td>Three domain experts. One manager The analyst. The secretary.</td>
<td>A summary of the results provided by the analyst; overview of the process undergone, and a presentation of the final prediction models. A demo of application of the prediction models: change specification, application of the change on the prediction models and quality prediction in terms of propagation paths and the modified QCF values.</td>
<td>The analyst prepared a simulation demo.</td>
<td>T:20</td>
</tr>
</tbody>
</table>

**TABLE I**

**OUTLINE OF THE PROCESS OF THE PREDIQT-BASED ANALYSIS**
A. Evaluation of predictions

During the last part of the W5 meeting (that is, upon completion of the PREDIQT-based analysis), a thought experiment was performed by asking the domain experts to estimate the new root node QCF values on the respective DVs, due to a specified change (given the current and the new QCF values of the leaf nodes affected, as well as the current QCF value of the root node). The change specified was a major, already implemented architectural design change, which added a new functionality to the system. The evaluation (simulation and thought experiment) assumed reversal of the change. The change affected up to three leaf nodes on each DV. The purpose of the thought experiment was to test usefulness of the predictions obtained from the models. That is, we assume that the domain experts have thorough knowledge of the system, and that their root node estimates reflect the reality of how the quality characteristics are affected by the change. Then, the simulated root node value is compared to the thought experiment provided one. Since propagation during the simulation is subject to structure and parameter values of the prediction models, as well as the identified leaf nodes and their modified QCFs, all these aspects are incorporated into the evaluation when the simulated and the estimates (through the thought experiment) root node QCFs are compared.

The thought experiment showed the following relationship between the simulated root node QCF values and their corresponding estimates (provided by the domain experts), regarding the respective above presented simulations on:

- the first one of the two DVs dedicated to Maintainability: no deviation between estimated (by the domain experts) and simulated (by PREDIQT)
- the second one of the two DVs dedicated to Maintainability: estimated is 4.5% higher than simulated
- the first one of the two DVs dedicated to Usability with respect to the contents: estimated is 3% higher than simulated
- the second one of the two DVs dedicated to Usability with respect to the contents: estimated is 7.7% higher than simulated

B. Written feedback after the analysis

The summary provided here is based on contents analysis of the answers of five respondents. The answers have been provided on a pre-defined evaluation template (see Appendix 4). The answers have been abstracted and categorized in order to reduce the volume of raw text and reveal possible similarities and contrasts. More details from the written feedback are reported in Appendix 5.

All respondents have participated in the whole or parts of the analysis, and are affiliated with the customer organization. Table II summarizes the background of the respondents.

The main strengths pointed out are: “The PREDIQT method is useful and it suits well the problem addressed” (R2), “It was a way to in a systematic manner divide the problem in smaller parts, and then aggregate the quality level for the whole model” (R3), and “Modeling concept – propagation of assessments” (R4). A weakness repeatedly pointed out is the missing formal mapping of the parameter estimates to the model, i.e. the parameter estimates may be too sensitive to the context and the interpretation (R1, R3, R4, R5).

All five respondents agreed that the models facilitate communication, knowledge exchange and understanding of the target system, its architecture and its quality characteristics. R1 argues that “the workshops force people to communicate and harmonize into one model; the system is clarified and parts of the architecture are disclosed and discussed; the most important part is assigning estimates on quality characteristics, which forces people to make statements”. R2 argues that “the method provides a good model of the system, which can be communicated around; when a multi-disciplinary group manages to make a model of a complex problem and communicate around it, you have achieved a good result: when you additionally can make predictions based on the model, the result is even better.”

R1 points out that the effort needed for conducting the analysis is reasonable from a typical management consulting perspective, but in an engineering context, more effort should be directed towards specific parts.

Regarding the future use of the method, R1 expresses the intention to use the models developed in the future, for purpose of architecture development and dependability analysis. R2 and R3 express the wish to use the method in future projects, given that financing can be provided. R4 intends to use the prediction models if they can be tailored to specific use cases, while R5 writes: “I believe the model can be used to understand and predict the result/risk in different changes”.

R1 expresses that the PREDIQT method “has already served the purpose in creating understanding and analysis. If incorporated with more tool support, I think it can be utilized in practice”. R2 expresses that PREDIQT is very much better than no method, but it is unknown what it takes for it to be perfect. R3 and R4 express that the benefit from the method and quality of the predictions depend on the modeling skills and granularity of the models. R5 points out the challenge of interpreting the predictions due to the lack of documentation of the assumptions made during the parameter estimation.

Regarding challenges with usage of the method, R2 expresses two main issues: “access to competent resources to make the models and interpretation of the predictions and the corresponding uncertainty which requires competence”. R3 points out three challenges: “be sure that you have modeled the most important aspects; models need to be verified; define the values in a consistent way”. R4 sees the uncertainty challenge in the fact that the changes are marginal and therefore give small effects on the numbers, while R5 relates uncertainty to the insufficiently formal interpretation of the parameter values due to the assumptions made during their estimation.

Regarding the main benefit of the method, R2 expresses that PREDIQT “reduces uncertainty at change, but does not eliminate it; but it does systematize the uncertainty and reduce it sufficiently so that the method absolutely is valuable”. R3 sees the discussion of the quality characteristics and agreement
upon the most important ones, as the main benefit.

The improvements suggested include simpler tool support, stricter workshops, increased traceability between the models, reuse of the Design Models based on other notations, and in-advance preparation of the experts.

C. Verbal feedback during the analysis

The verbal feedback includes the responses and comments from the analysis team, given during the different meetings – mainly by the end of the analysis. These include:

- The quality values (or their relative distance) should be mapped to monetary values or a similar measure of cost/gain in order to facilitate a cost-benefit analysis and ease interpretation of the DV parameters.
- The granularity of the changes is given by the granularity of the models. That is, minor changes may have very negligible impact on the models, unless the models are fine grained. A remedy is to deliberately increase the detail level of certain parts of the models. Still, although the parameters in such cases are almost unchanged, the prediction models help understand the propagation paths.
- The process of developing and verifying the models facilitates discussions, system understanding and knowledge exchange among the participants.
- The analyst should be aware of the possible bias or interests of the participants, particularly when the parameters are based on domain expert judgments.
- Certain parameters require a holistic approach (e.g. business perspective) or a special background (e.g. end-user). Some parameters may be uncertain due to lack of representation of such competence in the domain expert panel.
- Better documentation of the semantics and contextual information regarding the DV nodes, is needed. This would ease the use of DVs and particularly parameter estimation when some time has passed after the DV structure is developed.
- Active participation of the domain experts in the model development contributes not only to the model quality, but also to the experts’ understanding of the models, and ability to use and maintain the models after the analysis.
- The time spent on development of the prediction models is much longer, than the time spent on the model verification. This has shown to be beneficiary, since model development was founded on numerous documents which the domain experts could interpret and relate to the quality notions. Doing this early in the process and consistently on all parts of the models while discussing the models in the group, is preferred to verifying certain parts of the models. Ideally, one should do both, but when the resources are limited, the choice we made was preferred (due to higher model quality early in the process, as well as more extensive brainstorming and discussions in the group) provided that the verification is satisfactory.

- The estimates are much more informative when considered and interpreted relative to each other, than individually. When one estimate is unambiguous in terms of the interpretation of the value and the assumptions made during its estimation, values of the others (on the same DV) may be compared to the well known one, in order to be interpreted.

D. Observations made during the analysis

Some of the main experiences and observations made by the analyst are presented in the sequel.

- One of the main challenges for the analyst during the development of the Design Models was acquiring an understanding of the expert terminology used in the system. The documentation received and the S1 meeting rectified this.
- Regardless of how well the analyst understands the target system and its quality characteristics, it is crucial that the analyst does not develop the prediction models alone. The model development and verification trigger many useful discussions among the domain experts, and help reveal inconsistencies and misunderstandings. In addition, the prediction models are intended to be used and maintained by the domain experts, who need to be able to relate to the models and the tools they are developed in. The optimal approach is that the analyst presents an initial version of the models, which are discussed, corrected and further developed in the group. Errors or missing parts in the initial models are often an advantage, as they trigger the discussions in the group.
- It is important to dedicate sufficient resources to characterization of the target, provision of the input and formation of a common understanding of the Quality Models. These are prerequisites for avoiding elementary discussions and ambiguities during the rest of the analysis.
- The analyst has to be aware of the inconsistencies of the terminology used in documents and the verbal communication among the domain experts, as well as between the overall stakeholders. Any such inconsistencies should
be clarified, preferably through the Quality Models or the Design Models.

- The PREDIQT method has to be sufficiently understood by all parties.
- It is important to use a notation for the prediction models, that all analysis participants can relate to.
- The time taken to estimate the parameters of the DVs is at least twice as long as the time needed to develop the structure of the DVs. It is necessary to explain that the DV structure is developed with respect to both Design Models and Quality Models, since dependencies are modeled with respect to the respective quality characteristic that the DV is dedicated to. Availability and common understanding of the Quality Models during parameter estimation is crucial.
- The structure of the DVs may need to be adjusted during the DV parameter estimation. For this, tool support more flexible than what our MS Excel sheets currently offer, is needed.
- When developing the DVs, certain assumptions and choices are made. Traces to the specific Design Model elements may exist, and only certain indicators from the Quality Models may be used in estimation. The current tool support is insufficient for efficiently documenting these aspects “on the run” during the meetings.
- Since a PREDIQT-based analysis requires considerable effort from the customer organization, it is essential to ensure commitment of the management and allocate the resources needed.
- It is important to make the right balance between the representativeness of the domain expert panel and the effectiveness of the analysis, when choosing the size of the analysis group. Although a larger group is likely to increase statistical significance of the data, time consumption on the discussions may rapidly grow with the number of the participants. Therefore, one should ensure that a fraction of the domain expert panel is present at all meetings and provides continuity, while some turnover of the overall participants depending on the goal of the meeting may be beneficiary. The turnover however necessitates updates of the participants on both the PREDIQT method and on the current status/results of the analysis. There is clearly a trade-off between the resource consumption and the model quality.
- The meetings should be as tightly scheduled as possible, provided that the necessary preparations are feasible. The rationale is to prevent the need to updates on recent results.
- Approximately half a year has been a reasonable time allocation for this case study. In a commercial analysis, a tighter course during a shorter period of time could be achieved, if the participants can prioritize the analysis even more among their overall tasks and if the tool support is improved.

VII. Evaluation with respect to the success criteria

In this section we evaluate the performance of the PREDIQT method in this case study, with respect to the success criteria presented in Section IV. Thus, this section addresses the last stage of the research method depicted by Figure 3.

**SC1**: The PREDIQT-based analysis facilitates predictions providing sufficient understanding of the impacts of architectural design changes on system quality characteristics, so that informed decisions can be made.

The ability of simulating a realistic change during meeting W5 and the assessment reported in Section VI, indicate that we have been able to develop an understandable and harmonized model of the system, communicate around the model, identify the dependencies and simulate the impacts of changes.

By performing thought experiments on the root node, the change propagation and its impact from the leaves throughout the different parts of the DVs, was evaluated. Whether the deviation reported is sufficiently small, is up to the customer to assess. The answers obtained in Section VI suggest that this is the case.

The thought experiment based evaluation of the predictions resulted in no deviation on the first DV, and some degree of overestimation during the thought experiments. This can be due to varying quality of the specific models or optimism of the domain experts. We observe however that the deviation between the simulated (based on the DV models) and the estimated (through the thought experiments) root node values during both model validation and the evaluation of the predictions, has no repeatable pattern but considerably high variance. Therefore, we do not have reason to assume bias in the relationship between the simulation and the thought experiments.

Many different parts of the DVs were affected during the evaluation, which ensured both variation and complexity in the change propagation – that is, coverage of the evaluation. Moreover, the number of parameters (QCFs and Els) in each one of the four different DVs was around 60-70. Being able for a domain expert to remember the values and the structure of the four different DVs (which had been developed incrementally weeks before) should be improbable. Together with the above mentioned variance, this should exclude the possibility that the domain experts were able to quickly calculate propagation of the changes during the thought experiments. They were whatsoever asked by the analyst to purely use their system knowledge when performing the thought experiments.

The evaluation of the degree to which the simulated and the thought-experiment based estimates coincide would have been more reliable if uncertainty [19] had been expressed in the estimates. Then, one could have based the evaluation on whether the deviation is within the already present uncertainty of the estimates. Due to the limited time and the extent of the prediction models, we did not have the resources for also including the uncertainty handling in the analysis.

**SC2**: The PREDIQT-based analysis is cost-effective.
The analysis indicates that the PREDIQT method is feasible in a fully realistic setting and within the limited resources allocated. The process of the PREDIQT method was undergone, addressing the whole target of analysis and resulted in prediction models that, as the assessment indicates, provide the customer organization with useful basis for understanding the impacts of changes, capturing the propagation paths and obtaining the predictions. The domain experts have actively participated in development and revision of the Design Models and the Quality Models, and fully developed the DVs which cover the target of the analysis.

The feedback from R1 and R2 (customer management representatives) presented in Section VI, indicates cost-effectiveness of the analysis. The analysis has required approximately 215 (see Table I) man-hours (apart from the reporting), which is within the resources allocated. There are, however, some issues that must be taken into consideration when evaluating these numbers. Firstly, this was the second time the PREDIQT-based analysis was performed to a real industrial case. Hence, even though the analysis team included one of the inventors of the PREDIQT method, the process is not fully streamlined yet, due to limited empirical experience with PREDIQT. It can reasonably be assumed that the process will be more effective as the analysts gain experience with applying the PREDIQT method.

Furthermore, the process of the PREDIQT method assumes that the Design Models are in place prior to the analysis. Since this was not the case, considerable time had to be spent on modeling the system. Based on the experience gained and given that the Design Models are available as input to the analysis, we believe that it should be possible to carry out this kind of analysis within a time frame of approx. 60 man-hours spent by analyst (not including writing a final report) and ca. 50 man-hours spent by the overall participants. Hence, the success criterion appears to be fulfilled in this case. There is however still a need for a reference/baseline for comparing our results with the results from possible alternative methods. The future studies should address this, as well as cost-effectiveness per DV/quality characteristic/Design Model. Reusability of results (e.g. through experience factories) also contributes to the cost-effectiveness and should be examined in the future work.

**SC3: The prediction models are sufficiently expressive to adopt the relevant architectural design changes and analyze their effects on quality.**

The diversity of changes in the demo and the validation, the ability of simulating a realistic change during meeting W5 and the assessment, indicate that we have been able to develop a harmonized model of the system and use it for identifying the dependencies and simulating the impacts of all proposed changes. The participants provided a lot of information about the target during the analysis process. There were no instances where we were not able to capture the relevant information in the prediction models. Further application of the prediction models is however needed in order to evaluate their expressiveness and whether they can be maintained and used during the needed time period.

**SC4: The prediction models are sufficiently comprehensible to allow the domain experts to be actively involved in all phases of the PREDIQT process and achieve the goals of each phase with a common understanding of the results.**

The number of diagrams and parameter estimates was considerable. Still, the multi-disciplinary domain expert panel affiliated with several departments of the customer organization managed to discuss and agree upon the the different parts of the eventually harmonized and approved prediction models. The fact that the domain experts actively participated and continuously made progress according to the schedule of the analysis, managed to perform thought experiments and apply the models, indicates comprehensibility of the models. One of the most demanding parts of the analysis – development of the DVs, was entirely performed by the domain experts and only facilitated by the analyst.

The available prediction models were presented by the analyst during the meetings, in order to validate the correctness of the models or use them as basis for the forthcoming stages. There were many occasions where the participants suggested modifications, explained their rationale, or asked relevant questions about some detail in a model. This indicates that the models were in general comprehensible for the participants, and the postmortem review suggests that the models served well as an aid in establishing a common understanding of the target.

Still, comprehensibility of the models may vary among the participants and between the models depending on the knowledge of the system and the modeling notation. The fact that all the participants in this analysis had a strong technical background may have contributed to making the models easier for them to understand than would be the case for an even more diverse group. It is still necessary to have an analyst explain the method and the models, as well as facilitate and manage the process, since the current tool support is insufficient for ensuring a structured process and since an adequate PREDIQT manual currently does not exist. The analyst has played a rather active part during the analysis. A disadvantage is that the active role may have influenced the analysis. However, the involvement of the analyst is openly reported and reflected upon. It has also allowed better insight into the process and a more detailed evaluation of the results.

**SC5: The PREDIQT-based analysis facilitates knowledge management and contributes to a common understanding of the target system and its quality.**

The answers reported in Section VI consistently suggest that the PREDIQT-based analysis facilitates knowledge management. The models have served as a means of documenting the system, triggering discussions and exchanging knowledge. The means of triggering the discussions and further increasing participation of the domain experts can still be developed as a part of the method. It is for example essential that the analyst does not too actively develop any models or uses the tools alone, which would make it more demanding for the domain experts to use and maintain the models.
More structured process, improved traceability between the models, documentation of assumptions and rationale, as well as improved tool support (in terms of flexibility of modifications, usability, process guidance, as well as documentation of traces, rationale and assumptions) would facilitate the knowledge exchange and certainty of the models.

 VIII. CONCLUSIONS

The PREDIQT method makes use of models that capture the system design, the system quality notions and the interplay between system architecture and quality characteristics, respectively. The predictions result in propagation paths and the modified values of the parameters which express the quality characteristic fulfillment at the different abstraction levels. PREDIQT aims at establishing the right balance between the practical usability of the models, and the usefulness of the predictions. We are not aware of other approaches that combine notions of architectural design and quality in this way. However, the issues of metrics estimation, system quality and the various notations for modeling system architecture, have received much attention in the literature [8, 15, 11, 1, 3, 2, 9, 7, 13, 12, 4, 6, 14].

The paper has presented experiences from using the PREDIQT method in an industrial case study. The contributions of the paper include:

1) a detailed account of how the PREDIQT method [16] scales in an industrial context
2) an evaluation of the performance of the method in an industrial context.

The experiences and results obtained indicate that the PREDIQT method can be carried out with limited resources (five workshops and 215 man-hours), on a real-life system and result in useful prediction models. Furthermore, the observations indicate that the method, particularly its process, facilitates understanding of the system architecture and its quality characteristics, and contributes to structured knowledge management through system modeling. All stakeholders, including the customer, the domain experts and the analyst gained a better and a more harmonized understanding of the target system and its quality characteristics, during the process. The knowledge management in the context of this case study has concerned acquisition, exchange and documentation of the knowledge available (in forms such as domain expert knowledge, documentation or logs), regarding the architectural design of the system, non-functional (quality) characteristics of the system and the interplay between the architectural design and the system quality.

Four evaluation methods have been used: a thought experiment in order to evaluate the predictions obtained; written feedback from the analysis participants; verbal feedback during the analysis from the analysis participants; and observations made by the analyst during the case study. The evaluation methods complement each other and are to a varying degree used during the discussion of the success criteria. For example, when discussing success criterion 1 (usefulness of the predictions for making informed decisions), the thought experiment is mainly referred to.

The issue of method scalability concerns two aspects which our results indicate have been achieved and balanced: resources required to perform the analysis and the usefulness of the prediction models. In particular, the evaluation argues that:

- the PREDIQT-based analysis facilitates predictions providing sufficient understanding of the impacts of architectural design changes on system quality characteristics, so that informed decisions can be made,
- the PREDIQT-based analysis is cost-effective,
- the prediction models are sufficiently expressive to adopt the relevant architectural design changes and analyze their effects on quality,
- the prediction models are sufficiently comprehensible to allow the domain experts to be actively involved in all phases of the PREDIQT process and achieve the goals of each phase with a common understanding of the results, and
- the PREDIQT-based analysis facilitates knowledge management and contributes to a common understanding of the target system and its quality within the scope of the characterized target system and objectives.

Full documentation of the case study exists, but its availability is restricted due to confidentiality required by the customer. Hard evidence in the form of measurements to validate the correctness of the predictions would have been desirable, but this was unfortunately impossible within the frame of this case study. Instead, we have relied on extensive documentation and the domain expert group with solid background and diversity. Still, thought experiment based validation of models and evaluation of the predictions have weaknesses compared to the measurement based ones. Particularly, we can not exclude that possible undocumented or inconsistent assumptions have been made in model development, although the Quality Models and the active participation of the domain experts in all model development should prevent this. Statistical power was limited, due to low number of participants. The careful selection of experienced participants and the variety of the changes specified during model validation, compensated for some of this. Another weakness is that the same domain expert group has developed and validated the prediction models. However, given the complexity of the prediction models (which are very unlikely to be remembered), the variation of the changes applied and variance of the deviation pattern obtained (between the simulations and the thought experiment based estimates), we can not see any indication of bias due to the same expert group.

Although the above mentioned threats to validity and reliability are present in such a study, we argue that the results indicate the feasibility and usefulness of the method in a real-life setting. The study has also provided useful insight into the current strengths and weaknesses of the method, as well as suggested directions for future research and improvements.
Particularly, the needs for improved traceability, even more structured process guidelines and better tool support have been highlighted.

Note that PREDIQT has only architectural design as the independent variable – the Quality Model itself is, once developed, assumed to remain unchanged. This is of course a simplification, since system quality prediction is subject to more factors than architectural design. Usage profile, quality definitions and process are examples of the factors whose variation PREDIQT does not address. Although this case study has evaluated PREDIQT in a different domain compared to the one reported in [16], many more evaluations are needed for evaluating the external validity of the method.

The target system is representative for the systems intended to be within the scope of the PREDIQT method. This is the second trial of PREDIQT in a real-life setting and both trials have given strong indications of feasibility of the method, reported similar benefits (understanding of system architecture and its quality, usefulness of estimates particularly when interpreted relative to each other, and usefulness of the process) and undergone the same stages of the PREDIQT process. There is no significant difference in the size or complexity of the prediction models between the two case studies. No particular customizations of the method were needed for this trial. Thus, we have reason to believe that it should be possible to reapply PREDIQT in another context.

Acknowledgments: This work has been conducted as a part of the DIGIT (180052/S10) project funded by the Research Council of Norway, as well as a part of SecureChange project and the NESSoS network of excellence both funded by the European Commission within the 7th Framework Programme.

REFERENCES

APPENDIX 1: RESEARCH METHOD

The research method is motivated by the guidelines for case study research provided by Yin [21]. This section reports on the rationale for main decisions regarding case study design, data acquisition and the analysis. A deductive approach is undertaken, where the already defined PREDIQT method is exposed to an empirical trial in the form of a case study. The structure of this section follows the structure from [21].

A case study

The technical definition of a case study is as follows [21].

1) A case study is an empirical inquiry that:
   - investigates a contemporary phenomenon within its real-life context, especially when
   - the boundaries between phenomenon and context are not clearly evident.

2) The case study inquiry:
   - copes with the technically distinctive situation in which there will be many more variables of interest than data points, and as one result
   - relies on multiple sources of evidence, with data needing to converge in a triangulating fashion, and as another result
   - benefits from the prior development of theoretical propositions to guide data collections and analysis.

A case study method is, according to [21], used when the researcher deliberately wants to uncover contextual conditions – believing that they might be highly pertinent to the phenomenon of study. A case study comprises an all-encompassing method – covering the logic of design, data collection techniques, and specific approaches to data analysis.

The main stages of the research method performed in this case study are depicted by Figure 3. The case study design included characterization of the research question, the units of analysis and the success criteria, as the main outcomes.

The PREDIQT-based analysis was performed by following the pre-defined process of the PREDIQT method. However, instead of performing predictions of effects of future changes during the last workshop (as specified by the PREDIQT process), we chose to demonstrate how prediction models can be applied by simulating the effects of reversal of a very large already implemented change. As such, the model application phase is not fully covered, but only demonstrated. The affected Design Model and DV elements were identified and their modified parameter values estimated by the domain experts. Thereafter, the simulation on the DVs was made by the analyst.

Additionally, in order to evaluate the predictions obtained, a thought experiment regarding the effect of the change on the root nodes of the respective DVs, was performed by the domain experts. Thus, this was a part of the method assessment. Besides the thought experiment, the assessment measures included: written feedback (based on an evaluation template) from the analysis participants (affiliated with the customer organization) provided upon completion of the analysis and the above mentioned thought experiment based evaluation; verbal feedback during the analysis from the analysis participants (affiliated with the customer organization); and observations made by the analyst during the analysis. Based on the results of the PREDIQT-based analysis and the assessment, we provide an evaluation with respect to the evaluation criteria.

Design of the case study

The research question of this case study is "How does the PREDIQT method perform in a fully realistic setting and when applied on a system from a different domain than the previously evaluated one?". The propositions are deduced based on the three phases of the process of the PREDIQT method. Since the earlier performed case study [16] indicated feasibility of the PREDIQT method, the objective of this one is to investigate the performance of the method in a more structured manner. Furthermore, as proposed by [21], we included the following theory in the case study design:

The case study will show the weaknesses, the strengths and the cost-effectiveness of the PREDIQT method when exposed in a new domain, as well as within a fully realistic and organizationally complex setting. This will provide the insight into feasibility and performance of the method. Particularly, we will uncover if the method still scales in terms of resource consumption and the size of the prediction models, when performed on a different domain and in a more complex setting compared to the earlier case study. The case study will also identify the needs for further research.

Phase 3: “Application of prediction models” is successful if:
   - the prediction models are sufficiently expressive to adopt the relevant architectural design changes and analyze their effects on quality
   - the prediction models can indicate the change propagation paths and provide the resulting quantitative values (of the quality characteristics) with sufficient certainty
   - the prediction models are sufficiently comprehensible to the domain experts so that the specified changes can be applied and the predictions (propagation paths and the modified quality characteristic values) can be interpreted.

Phase 2: “Verification of prediction models” is successful if:
   - the prediction models are sufficiently comprehensible to the domain experts so that they can be evaluated, fitted and approved with a common understanding
   - the prediction models are approved by the domain expert panel with regard to their expressiveness, granularity, correctness and completeness in representing the target system within the objectives of the analysis. That is, the prediction models include the necessary representation of the target system and can adopt the kinds of changes characterized in Sub-Phase 1 of Phase 1.
   - the prediction models are approved by the domain expert panel with regard to certainty of the estimates.

Phase 1: “Target modeling” is successful if:
   - the process and the objectives of the PREDIQT analysis are comprehensible to the domain experts so that a
The four analytic tactics available by [21] for increasing construct validity applied in this case study are:

- do pattern matching; this is done through comparing the simulated to the thought experiment based effects of the changes applied on one or more leaf nodes (independent variables), on the root node (dependent variable) of the DV. The procedure is applied on different leaf nodes of each DV, and based on multiple independent changes. Moreover, the changes have previously been implemented and their effects are known (they have therefore been reversed in the prediction models). Additionally, a diverse group of the domain experts has been involved in performing the thought experiments. A weakness is however that the same domain expert group has developed the prediction models. However, given the complexity of the prediction models which is unlikely to be remembered, the variation of the changes applied and variance of the pattern obtained, we can not see any indication of bias due to the same expert group.
- do explanation building; the deviations obtained from comparing the simulation results to the thought experiment based estimates are explained through model quality and expert optimism. The data collected through the evaluation is however not representative for substantiating the explanation.
- address rival explanations; the rival explanation suggesting a bias due to memorization of the models by the experts, is rejected due to complexity and inaccessibility of the DVs.
- use logic models; this related the quality characteristic decompositions to the estimates of the leaf nodes on the respective DVs, and their further propagation to the root nodes. The above mentioned measures from the pattern matching apply.

One of the two tactics available for increasing external validity, is applied in this case study:

- use theory in single-case studies; the theory is specified above.
- use replication logic in multiple-case studies; only partially applied, this being the second trial of PREDIQT in an industrial setting.

Two tactics are available for increasing reliability of a case study:

- use case study protocol
- develop case study database

As mentioned above, full documentation of the case study exists, but its availability is restricted due to confidentiality required by the customer.

Due to several units of analysis and fulfillment of one type of rationale for single-case study design, our case study is classified as embedded single-case study. The types of rationale for a single-case study are:
critical case; we are testing an earlier prescribed method with the aim to confirm, challenge or extend it. However, since a critical case should be designed so that it can be used to generalize or falsify a theory [10], we can not claim that our case is critical.

• testing a unique or an extreme case; neither unit of analysis is considerer to be extreme or unique, given the frames of a typical analysis.

• representative or typical case; we believe to have captured circumstances and conditions of a realistic case study.

• revelatory case; we do not consider the units of analysis to previously having been inaccessible for scientific investigation.

• longitudinal case; this case has not been studied at several points of time.

Presence of one rationale is, according to [21], sufficient for choosing a single-case study. The embedded single-case study allows focusing on specific units of measure, while enabling study of the rest of the context. Thus, the evaluation is targeted, while the larger units are included as well.

The sample size and the data quality have been given by the resources available within the study. The extent of the documentation, number of participants, qualifications of the participants and the resampling effort have defined this. We have fully utilized all resources available within the frames of the analysis.

Preparing data collection

Yin [21] emphasizes the skills of the analyst an an important prerequisite. Given the analyst’s professional background and the role in development and earlier trial of the PREDIQT method, we consider this condition to be fulfilled.

Another prerequisite is the training and preparation for the case study. Given the preliminary meetings when the method was presented, as well as systematic guidance of the analysis participants provided throughout the case study, we consider this condition to be fulfilled. The goal has been to have all participants agree upon the objectives, understand the basic concepts, terminology, the process, the rationale and the issues relevant to the study. Discussions rather than presentations have been the key approach, in order to ensure that the desired level of understanding has been achieved.

The third prerequisite is the protocol development. The meeting notes, minutes from the workshops, meeting presentation slides and all models have been either produced in groups, or reviewed once they are produced.

The fourth prerequisite – screening of the case study nominations involved nomination of the participants of the analysis and characterization the the target system. The respective nominations were done by the customer organization and the established analysis team, respectively.

Collecting the evidence

Yin [21] discusses six sources of evidence: documentation, archival records, interviews, direct observation, participant-observation, and physical artifacts which are highly complementary to each other. Additionally, [21] presents three essential data collection principles:

• use multiple sources of evidence
• create a case study database
• maintain a chain of evidence

The documentation has included: administrative documents, minutes from meetings, presentation slides, meeting notes, filled evaluation forms, system and enterprise architecture documentation, requirements specification, system design documentation, service level agreement, operational environment specification, procedural descriptions for change request in the organization, information model of the system, and the prediction models developed.

The archival records are participant contact details, the disclosure agreements, e-mail correspondence, and a listing of challenges from the preliminary meetings.

The interview form applied was structured conversation with the analysis participants. The statements have been documented in form of models and meeting notes.

The analyst has reported on the main direct observations and participant observations from the case study. The analyst has played a rather active part during the analysis. A disadvantage is that the active role may have influenced the analysis. However, the involvement of the analyst is openly reported and reflected upon. It has also allowed better insight into the process and a more detailed evaluation of the results. Still, the target characterization, the model revisions and approvals and the evaluations have purely been performed by the domain experts who were only guided by the analyst. Additionally, the participant observations from the overall participants have been collected in written and verbal forms, and reported as a part of the postmortem review.

The multiple sources of evidence used in the case study have developed converging lines of inquiry, a process of triangulation which makes a conclusion more valid when based on several sources of correlating evidence. Documentation, informers, data, responders and observers have been triangulated. The triangulation is a means of increasing the construct validity.

As mentioned above, all documentation is stored and traceable, but its availability is restricted due to the confidentiality. The models have been stored in their incremental versions. The documentation as such provides a chain of evidence which allows tracing the origins of the main results.

Analyzing case study evidence

The case study has relied on the research question stated above, and the evaluation has been driven by the success criteria specified in Section IV. The success criteria serve as propositions. The evaluation template for postmortem also addresses the main success criteria. The rival explanations have been considered as a part of the section which discusses threats to validity and reliability, but due to inability of determining statistical significance of the input, no null hypothesis has been formulated. The prediction models have been analyzed in terms of their size, complexity, comprehensibility and the
deviation between the simulated and the thought experiment based estimates. Furthermore, the results of the postmortem review have been summarized based on contents analysis of the answers which are abstracted and categorized in order to reduce the volume of raw text and reveal possible similarities and contrasts.

As mentioned above, pattern matching has been performed during both model validation and model application, by comparing the simulated estimates with the ones obtained through thought experiments. The patterns are related to the dependent variables (root nodes) but they also validate the affected parts of the DV which are involved in the propagation from the modified leaf nodes. If the patterns coincide sufficiently, it helps strengthen the internal validity.

The validation is based on multiple independent changes which address different parts of each DV. Moreover, a multi-disciplinary expert panel was involved in the thought-experiment based evaluation. The change which the final evaluation was based on was realistic, extensive, known and affected several parts of each DV. Both propagation paths and the values obtained from the simulation, were evaluated.

Uncertainty handling was not included due to the limited resources. The priority was rather to develop the prediction models which cover the target of the analysis. However, extensive use of the Design Models and other documentation, as well as discussions when developing the DVs did aim at increasing the precision.

Comparison of two case studies and cross-case synthesis is another means of explanation building. This case will be briefly compared to the previous one, in relation to the discussion of the threats to external validity.

Logic models are presented in [21] as a technique for stipulating complex chain of events over time. In a logic model, the dependencies are modeled and the data acquisition is planned for testing the effects of changes of the modified parts on the related ones. In our case, the DVs and the evaluation addressing the root nodes. The structure of the Quality Models is also indirectly tested, as the estimates are based on it. The DV structure is also based on the quality characteristic definition, as dependencies are also expressed with respect to the quality characteristic. The DVs are also partially traceable elements of the Design Models. Thus, the relationships between the prediction models, as well as relationships between the nodes of a DV may be considered as a logic model.

Reporting the case study

The contents of this report have been driven by the success criteria and the related reporting needs. The audience is the research community and the practitioners interested in future trials of the PREDIQT method. [21] provides guidelines on the reporting of a case study. One of the issues addressed is the anonymity, which is accepted when absolutely necessary. Due to the confidentiality requested by the customer, the concrete context and the results have only been reported to the degree approved. Still, we believe that the paper provides useful insight into the experiences from the trial, and fulfills the objective regarding evaluation of the method, as well as suggestion of the future work.

The contents of the paper have been authored by the research group, which the analyst is a part of. In an attempt to avoid bias in the interpretation of the results, emphasis has been put on neutrally presenting the factual results, rather than interpreting and analyzing them in detail. The relevant results have been presented, including both supporting and challenging data. The selectiveness is, as argued by [21], relevant in limiting the paper to the most critical evidence, instead of cluttering the presentation with supportive but secondary evidence (which may sway or bore the reader). The paper has been approved by the customer organization, with the aim of ensuring that agreement on the facts presented is achieved, as well as that no confidential information has been disclosed. [21] argues that such an approval increases the construct validity of a study.

APPENDIX 2: SETUP AND DATA COLLECTION DURING THE PREDIQT-BASED ANALYSIS

The analyst had more than nine years of relevant professional experience in software engineering. The customer management representatives and the domain experts had between 15 and 33 years of relevant professional experience each.

The system documentation received by the analyst from the customer organization contained mostly descriptions and specifications in the form of verbal input, presentation slides, textual documents, sketch-like models integrated in various documents, samples of change request forms, MS Excel documents in which the system structure is described, and webpages. Thus, the Design Models of the target system had to be developed as a part of the analysis. All Design Models and Quality Models were developed in UML, using the MS Visio tool. The DV structure was developed in MS Visio, and then transferred to an already developed MS Excel based tool [16]. The latter tool displays DVs and supports automatized parameter propagation and sensitivity analysis.

The original input and presentations provided in relation to the meetings PM1 and PM2 were, in terms of scope of the systems presented and abstraction level, considerably broader than the target of the analysis defined at the meeting W1. During the W1 meeting, the group was guided by the analyst (using precise questions and examples) to characterize the target in terms of scope and the quality characteristics. This input, in addition to the written documentation already received, was sufficient for the analyst’s development of the initial Design Models and Quality Models of the system. The quality characteristics identified during W1 were compiled to the relevant parts of the ISO 9126 [1] standard, where the interpretations and formal definitions of all elements are provided. All quality characteristics in the compiled model were decomposed into sub-characteristics and indicators, in accordance with the relevant parts of the ISO 9126 standard.

Because verification only addresses certain parts of the models, it was deliberately chosen to spend more resources
on the model development, particularly estimation of the DV parameters, than on their verification. Much of the existing documentation was used to deduce the values, rather than to verify them afterwards.

All DVs were (in terms of structure and parameter estimations) entirely developed and revised by the domain expert panel. Support from the analyst included guidance on the procedures, rules, model syntax and documentation.

At meeting W4 the focus was on validation. The objective was to check whether the models can predict within an acceptable threshold, in which case they are approved. After a walkthrough of the four DVs, the following procedure was followed for 9 independent change simulations:

1) the analyst presented the leaf node in question, its parent node, the DV it is a part of and the current QCF of the leaf node in question, without showing the DV
2) the domain experts specified a known change which has already been implemented on the system and which influenced the selected leaf node
3) the analyst presented the current QCF of the root node
4) the analyst asked the domain expert panel for an estimate of the new QCF of the node in question, given that the change is deployed on the current system
5) the analyst asked the domain expert panel for an estimate of the new QCF of the root node
6) the analyst used the DV to predict the new root node value after the change.

On the two maintainability DVs, 2 and 3 leaf nodes were modified, respectively. On the two usability with respect to contents DVs, 2 different leaf nodes on each DV were modified. Each node was modified independently according to the procedure above. In case the change specified already has been applied on the system, its reversal is estimated and simulated during the validation. A table with the parameters from the procedure was dedicated to each change and displayed on the presentation slides.

At meeting W5 the focus was on demonstrating application of the prediction models. The demo included:

1) Specification of a change which has already taken place
2) Identification of the Design Model elements affected (By domain experts))
   - Specification and substantiating of the Design Model changes
3) Identification of the related parts of the DVs (By domain experts))
   - Modification of the DV parameter values
4) Simulation of the change propagation on the DVs (By analyst)
5) Documentation of the DV nodes affected and the modified QCF values

The demo of model application assumed reversing a major change which has already been deployed on the system. Hence, the current prediction models incorporated the state after the change whose reversal was to be demonstrated.

At each meeting, handouts of the current prediction models and presentation slides were provided to all participants.

APPENDIX 3: OUTCOMES OF THE PREDIQT-BASED ANALYSIS

This section reports on the main outcomes of the process presented in the previous section. We focus particularly on the final prediction models, and the result of their validation and application carried out during the respective stages of the process.

Characteristics of the prediction models

Both Quality Models and Design Models were developed using UML. The Quality Models decomposed the total quality of the system into two main quality characteristics: maintainability and usability with respect to contents. Maintainability was decomposed into three sub-characteristics (changeability, testability and stability), which again were decomposed into three, three and two indicators, respectively. Usability with respect to contents was decomposed into three sub-characteristics (information correctness, information availability and operability), which again were decomposed into four, four and three indicators, respectively. In addition, each subtree was supplemented by a node called “Other”, for model completeness purpose. All nodes of the Quality Model (except the ones called “Other”) were defined qualitatively and formally. Most of the definitions were retrieved from the ISO 9126 standard, the remaining ones were customized with respect to the target system. All formal definitions ensured normalized values between zero and one, where 0 denotes no fulfillment, and 1 maximum fulfillment.

The Design Models specified concepts, system structure and workflow. The Design Models consisted of 10 UML diagrams. Mostly, class diagrams were used and their size ranged over 10-20 classes. One activity diagram specified the workflow, and contained 10 (activity) elements and one decision point.

Two DVs were developed for each quality characteristic, that is, four DVs in total. The two DVs dedicated to a quality characteristic covered the two main perspectives of the system’s purpose. The same two perspectives were covered by the remaining two DVs for the other quality characteristic, but (in terms of structure) the DVs of the second quality characteristic were to a limited degree different from the respective corresponding DVs dedicated to the first quality characteristic. The main difference was in names and semantics of certain nodes. The parameter values are not comparable between the DVs of two quality characteristics.

For the quality characteristic Maintainability, the first DV had 31 nodes, of which 26 nodes were leaves. The second DV had 35 nodes, of which 28 nodes were leaves. For the quality characteristic Usability with respect to contents, the first DV had 31 nodes, of which 26 nodes were leaves. The second DV had 34 nodes, of which 27 nodes were leaves.

Results of the model validation

Table III summarizes the results of the above presented validation. The first column specifies the DV on which the
change is introduced. The second column shows the difference between the estimated (i.e., modified due to the change by the domain expert panel) and the old (prior to the change) value of QCF on the leaf node addressed. The third column shows the difference between the simulated (by the DV) and the old (prior to the change) value of QCF on the root node of the DV. The last column shows the difference between the simulated (by the DV) and the estimated (by the domain experts) value of QCF on the root node of the DV.

Results of the demonstrated application of the prediction models

Of the 10 existing diagrams of the Design Models, 7 diagrams were affected by the change specified. Within these 7 diagrams, the number of elements affected by the change was 2 out of 4, 1 out of 20, 5 out of 8, 2 out of 9, 1 out of 6, 2 out of 12 and 2 out of 5, respectively.

On the first one of the two DVs dedicated to Maintainability, three leaf nodes were affected as follows: increase by 1%, increase by 1% and unchanged QCF, respectively. The simulation showed that one internal node was affected and no significant effect on the root node was indicated by the simulation.

On the second one of the two DVs dedicated to Maintainability, one leaf node was affected by a decrease of 30%. The simulation showed that one internal node was affected and the simulated effect on the root node was a decrease of QCF by 3%.

On the first one of the two DVs dedicated to Usability with respect to the contents, QCFs of three leaf nodes were affected as follows: decrease by 3%, decrease by 2.5% and decrease by 8%, respectively. The simulation showed that one internal node was affected and the simulated effect on the root node was a decrease of QCF by 3%.

On the second one of the two DVs dedicated to Usability with respect to the contents, one leaf node was affected by a decrease of 30%. The simulation showed that one internal node was affected and the simulated effect on the root node was a decrease of QCF by 3%.

Appendix 4: Design of the evaluation template

The evaluation template, used in relation to the postmortem review, was designed in the form of a questionnaire in MS Word form, as follows:

Title: Evaluation of the PREDIQT method in the XXX case study

Introduction We need your feedback in order to further improve the PREDIQT method. Can you please provide your answers and comments to the following questions? All questions are regarding the case study you have been involved in during the second half of the year 2010, that is: “Analysis of XXX system, based on the PREDIQT Method”.

DATE:

1. Please specify your background and role in the case study:
   - Work place:
   - Position:
   - Education (degree):
   - Years of professional experience:
   - Role in the case study:

2. What is your general impression of the PREDIQT method? Please describe the experience from the case study in your own words. What do you think are strengths and weaknesses of the PREDIQT method? You may comment on both the process undergone and the final prediction models.

3. To what degree do you think the method (including the process and the final prediction models) facilitates communication, knowledge exchange and understanding with respect to:
   - the XXX system in general,
   - its architecture, and
   - its quality characteristics?

4. What is your experience from the process undergone? We are particularly interested in your opinion regarding the effort needed to develop the final prediction models, your understanding of the process and your opinion on the involvement or interest of the different participants, during the case study.

5. Please comment on your opinion regarding certainty, understandability, completeness and usability of the prediction models:
   - Design Models
   - Quality Models
   - Dependency Views

6. Do you intend to make use of any prediction models (Design Models/Quality Models/DVs), in the future? If so, which models do you intend to use further and in which context? If not, why not?

7. To what degree do you think the PREDIQT method (the process undergone and the resulting models) can aid understanding, analyzing and predicting the impacts of changes of XXX on its quality?

8. What do you see as the main challenges or problems with usage of the method and the final prediction models?

9. What do you see as the main benefits of the method (process and the prediction models)?

10. Which prediction models (Design Models/Quality Models/DVs) or properties of the models do you find most useful and why?

11. What kinds of improvements of the PREDIQT method (process and the prediction models) would you recommend?

12. Do you have further comments or suggestions?

Appendix 5: The feedback received through the evaluation template

This section summarizes the feedback provided by five respondents on a pre-defined evaluation template (see Appendix 4). The summary is based on contents analysis of the answers obtained. All respondents are affiliated with the customer organization. Table II shows the background of the respondents, as reported through answers on question 1 from the template.
On question 2, the main strengths pointed out are: “The PREDIQT method is useful and it suits well the problem addressed” (R2), “Going for structure, going for reuse, utilizing group judgment and calculation of scores” (R1), “It was a way to divide the problem in smaller parts, and then aggregate the quality level for the whole model” (R3), “Modeling concept – propagation of assessment” (R4). A weakness repeatedly pointed out is the missing formal mapping of the parameter estimates to the model, i.e. the parameter estimates may be too sensitive to the context and the interpretation (R1, R3, R4, R5). Complexity of the method and need for better tool support were pointed out by R5 and R1, respectively.

On question 3, all five respondents agreed that the models facilitate communication, knowledge exchange and understanding of the three aspects specified. R1 argues that “the workshops force people to communicate and harmonize into one model; the system is clarified and parts of architecture are disclosed and discussed; the most important part is assigning estimates on quality characteristics, which forces people to make statements”. R1 however points out that the semantics of the characteristics should be more formal and the process of their harmonization in the group more strict. R2 argues that “the method provides a good model of the system, which can be communicated around; when a multi-disciplinary group manages to make a model of a complex problem and communicate around it, you have achieved a good result; when you additionally can make predictions based on the model, the result is even better.” R3 argues that the communication is to a lesser degree efficient towards people outside the expert group, while R5 argues that the models are more useful for mapping consequences of changes, then for providing sufficiently precise predictions.

On question 4, R1 points out that the effort needed is reasonable from a typical management consulting perspective, but in an engineering context, more effort should be directed towards specific parts. R2 focuses on the benefit from useful discussions and insight into other’s understanding of the system and the model. R3 thinks it is difficult to be a part of such a process without using more time, and expresses that more time should have been used on verification, while R4 and R5 think that the time needed for modeling is extensive.

On question 5, the answers vary. R2 and R3 point out that the process itself and the fact that the participants are encouraged to provide numbers is important for improved understanding. R2 thinks the uncertainty challenge lies in the estimates, while R1 sees the main uncertainty challenge in the Design Models, while Quality Models are in accordance with the goal. R3 expresses that the uncertainty of Design Models and Quality Models comes from unknown usage profile, while DVs give clear dependencies and propagation of assessments. R5 emphasizes that non-linearities are difficult to explicitly model in the Design Models and the DVs.

On question 6, R1 confirms the intention to use the models developed in the future, for purpose of architecture development and dependability analysis. R2 and R3 express the wish to use the method in future projects, given that financing can be provided. R4 intends to use the prediction models if they can be tailored to specific use cases, while R5 writes “I believe the model can be used to understand and predict the result/risk in different changes”.

On question 7, R1 expresses “it has already served the purpose in creating understanding and analysis. If incorporated with more tool support, I think it can be utilized in practice”. R2 expresses that PREDIQT is very much better than no method, but it is unknown what it takes for it to be perfect. R3 and R4 express that the benefit from the method and quality of the predictions depend on the modeling skills and granularity of the models. R5 points out the challenge of interpreting the predictions due to the lack of documentation of the assumptions made during the parameter estimation.

On question 8, R2 expresses two main challenges “access to competent resources to make the models and interpretation of the predictions and the corresponding uncertainty which requires competence”, R3 points out three aspects: “be sure that you have modeled the most important aspects; models need to be verified; define the values in a consistent way”. R4 sees the uncertainty challenge in the fact that the changes are marginal and therefore give small effects on the numbers, while R5 relates uncertainty to the insufficiently formal interpretation of the parameter values due to the assumptions made during their estimation.

On question 9, R2 expresses that the method “reduces uncertainty at change, but does not eliminate it; but it does systematize the uncertainty and reduce it sufficiently so that the method absolutely is valuable”. R3 sees the discussion of the quality characteristics and agreement upon the most important characteristics, as the main benefit. R4 answers “the
final model”, while R5 writes: “the Design Model itself and DVs; making the model forces you to consider all parts and their dependencies”.

On question 10, R1 emphasizes the harmonized system understanding. R2 expresses that the totality is important, and it is difficult to select certain models that are more important, just as is the case when thinking if wheels or the motor are most important on a car. R3 answers: “since I think the process and not the prediction is most useful, the model is just a tool to facilitate the discussion”. R4 and R5 answer “DV’s”, and R5 also adds: “making the DVs forces you to consider all parts and their dependencies”.

On question 11, R1 emphasizes tool support, stricter workshops, and in-advance preparation of the experts. R2 also points out simpler tool support, which would enable large-scale use of the method by many end-users. Furthermore, R2 points out the need for increased traceability between the models, so that the relationships (and impacts among the models) are more explicit to the user, instead of model consistency/validity keeping (after a change deployment) being a manual task. R2 also expresses the challenge of keeping the models consistent with the underlying system which is undergoing an evolution and needs continuous support for prediction of the impacts of changes. Furthermore, R2 asks if existing system models can be reused and serve as the Design Models, as it may enable use of existing methods and analysis related to domain specific notations. R3 expresses that “it was difficult to define a quality value. Maybe the method should recommend how the teams could work to get consistent values through the model”. R4 suggests “detailed views of parts of the system”, while R5 again expresses the uncertainty regarding the possibility of modeling the system realistically in a linear model.

No respondent had comments in relation to question 12.

APPENDIX 6: THREATS TO VALIDITY AND RELIABILITY

The validity of the findings with respect to 1) the performance of the PREDIQT method; and (2) the results of the evaluation of the predictions based on the thought experiment and the overall assessment, depends to a large extent on how well the threats have been handled. In addition to reliability threats, four types of validity threats, presented in [5, 20], are addressed: construct validity, conclusion validity, internal validity and external validity.

Reliability is concerned with demonstrating that the operations of the case study can be repeated with the same results. Of course, an industrial case like this can never give solid repeatable evidence. There are too many contextual factors influencing what happens, as has been pointed out in Section VI with respect to assumptions and interpretations related to parameters. In the case of our analysis it may be argued that we should have focused more on verification, used historical data and performed more measurements rather than using merely expert judgments, but such data were not available. Instead, various excel sheets and other relevant documents (that models and numbers could be mapped to) were extensively used during Design Model development and DV estimation. We did however perform thought experiments on known and implemented changes (whose effects on quality are well known) at several meetings, in order to validate the models and evaluate the predictions. Still, the thought experiment based evaluation does have more weaknesses compared to the measurement based one.

The active involvement of the analyst may to some degree have influenced the analysis and needs to be reflected upon. However, since the role of the analyst is included in the method, the analyst should not have been a significant source of bias. The involvement of the analyst has also allowed better insight into the process and a more detailed evaluation of the results.

Construct validity concerns whether we measure what we believe we measure. Both the model development and the thought experiment relied on the subjective estimates provided by domain experts. There was some turnover among the domain experts, but two of them participated throughout the case study. The simulations themselves were conducted by the method developer after and independently from the thought experiment.

The change specifications included diverse, non-overlapping changes covering major parts of the prediction models. The quality attribute specific DVs were very complex, which minimizes the possibility that the domain experts were able to remember the DVs and thus quickly calculate propagation of the changes during the thought experiment. The variance of the discrepancy between the simulated and the thought experiment based values is quite high. All this considered, the risk that the prediction models, the change impact simulations and the thought experiment based estimates were consistently wrong, should be relatively small. Hard evidence in the form of measurements to validate the correctness of the predictions would have been desirable, but this was unfortunately impossible within the frame of this case study.

Another threat to the construct validity is the possible discrepancy in the understanding of the prediction models, particularly the Quality Models, by the domain experts. In case different assumptions have been made implicitly, they are not documented in the estimates. However, the active involvement of the participants in the model development and the ability of reaching agreement during development, validation and use of the models, do not give reason to assume that the models express else than specified.

Conclusion validity concerns the composition of participants and the statistical analysis. Statistical power was limited, due to low number of participants. The careful selection of experienced participants and the variety of the changes should have compensated for some of this.

Internal validity concerns matters that may affect the causality of an independent variable, without the knowledge of the researcher. Causality is present in decomposition of the quality characteristics to quality indicators, in the development of the DVs, and in the selection of the relevant indicators during the estimation of the DV parameters. We
have addressed these threats by involving the domain expert panel in all model development and validation, by actively using the available evidence and documentation during the development of the Design Models and the estimation, and by using the established ISO 9126 standard for the decomposition and definition of the quality notions.

Additionally, in order to ensure that the DV structure fulfills the requirements regarding completeness, orthogonality and a dependency model developed from the perspective of the quality characteristic in question, the analyst has systematically guided the domain experts and explained the principles, during the DV structure development. The Quality Models and the Design Models were also actively used during the DV development, as they provide the underlying quality definitions and system representation, respectively.

External validity concerns the generalization of findings of this case study to other contexts and environments. The target system is representative for the systems intended to be within the scope of the PREDIQT method. Moreover, this is the second trial of PREDIQT in a real-life setting and both trials have given promising results and useful insights. Both trials have given strong indications of feasibility of the method, reported similar benefits (understanding of system architecture and its quality, usefulness of estimates particularly when interpreted relative to each other, and usefulness of the process) and undergone the same stages of the PREDIQT process. More measurements and their analysis were however provided during the verification in the first case study (while thought experiments were performed during the evaluation). This case study has involved more domain experts and both the validation and the evaluation have merely been based on thought experiments. The individual DVs developed in each case study were of similar size. The first case study resulted however in three DVs – one for each quality characteristic, while this one resulted in four DVs – two for each quality characteristic. Apart from that, there is no significant difference in the size or complexity of the prediction models between the two case studies.

No particular customizations of the method were needed for this trial. Thus, we have reason to believe that it should be possible to reapply PREDIQT in another context.

We are not aware of other approaches that combine these elements in this way. However, the issues of metrics estimation, system quality and the various notations for modeling system architecture, have received much attention in the literature.

According to [8], most prediction models use size and complexity metrics to predict defects. Others are based on testing data, the quality of the development process, or take a multivariate approach. In many cases, there are fundamental statistical and data quality problems that undermine model validity. In fact, many prediction models tend to model only part of the underlying problem and seriously misspecify it.

Bayesian networks (BNs) [15, 11] allow incorporating both model uncertainty and parameter uncertainty. A BN is a directed acyclic graph in which each node has an associated probability distribution. Observation of known variables (nodes) allows inferring the probability of others, using probability calculus and Bayes theorem throughout the model (propagation). BNs are however demanding to parametrize and interpret the parameters of. This issue has been addressed by [18] where an analytical method for transforming the DVs to Bayesian networks is presented. It also shows that DVs are, although easier to relate to in practice, compatible with BNs.

PREDIQT is compatible with the established software quality standard [1], which is applied in this case study. The goal/question/metric paradigm [3, 2] is a significant contribution to quality control which also is compatible with PREDIQT and can be used for development of Quality Models and design of a measurement plan [9, 7].

[6] and [14] provide surveys of the software architecture analysis methods (SAAM, ATAM, ALPSM, SAEM etc.). Compared to PREDIQT, they are more extensive and provide a more high-level based architecture assessment of mainly single quality characteristic (maintainability or flexibility). Furthermore, they are not predictive, do not incorporate measurement, and quality is defined and quantified differently. ATAM [13, 12, 4] is, for example, more coarse-grained than PREDIQT in terms of both quality definitions and measurement. PREDIQT allows a more fine-grained analysis of several quality characteristic and their trade-offs simultaneously and with limited effort. Hence, an integration of the two may be worthwhile examining.

APPENDIX 7: RELATED WORK

The PREDIQT method for model-based prediction of impact or architectural design changes on system quality characteristics makes use of models that capture the system design, the system quality notions and the interplay between system architecture and quality characteristics, respectively. The predictions result in propagation paths and the modified values of the parameters which express the quality characteristic fulfillment at the different abstraction levels. The PREDIQT method aims at establishing the right balance between the practical usability of the models through the simplicity of the model structures, and the soundness of the predictions through a multi-stage structured process.
Chapter 13

Paper 5: Traceability Handling in Model-based Prediction of System Quality
Report

Traceability Handling in Model-based Prediction of System Quality

Authors
Aida Omerovic
Ketil Stølen
Traceability Handling in Model-based Prediction of System Quality

Abstract heading

Our earlier research indicated the feasibility of the PREDIQT method for model-based prediction of impacts of architectural design changes, on the different quality characteristics of a system. The PREDIQT method develops and makes use of a multi-layer model structure, called prediction models. Usefulness of the prediction models requires a structured documentation of both the relations between the prediction models and the rationale and assumptions made during the model development. This structured documentation is what we refer to as trace-link information. In this paper, we propose a traceability scheme for PREDIQT, and an implementation of it in the form of a prototype tool which can be used to define, document, search for, and represent the trace-links needed. The solution is applied on prediction models from an earlier PREDIQT-based analysis of a real-life system. Based on a set of success criteria, we argue that our traceability approach is useful and practically scalable in the PREDIQT context.
### Document history

<table>
<thead>
<tr>
<th>VERSION</th>
<th>DATE</th>
<th>VERSION DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2011-05-01</td>
<td>First draft version</td>
</tr>
<tr>
<td>0.2</td>
<td>2011-06-01</td>
<td>Complete draft version</td>
</tr>
<tr>
<td>1.0</td>
<td>2011-07-22</td>
<td>Final approved version</td>
</tr>
</tbody>
</table>
## CONTENTS

I  Introduction  5

II  Background on traceability  5

III  The challenge  7
  III-A  Structure of the prediction models  7
  III-B  Success criteria  8

IV  Our solution  9
  IV-A  Traceability scheme  9
  IV-B  Prototype traceability tool  9

V  Applying the solution on an example  11

VI  Why our solution is a good one  12
  VI-A  Success Criterion 1  12
  VI-B  Success Criterion 2  12
  VI-C  Success Criterion 3  12
  VI-D  Success Criterion 4  12

VII  Why other approaches are not better in this context  13

VIII  Conclusion and future work  14

References  15

Appendix 1: An overview of the PREDIQT method  16

Appendix 2: Guidelines for application of prediction models  18
Traceability Handling in Model-based Prediction of System Quality

Aida Omerovic*† and Ketil Stølen*†
*SINTEF ICT, Pb. 124, 0314 Oslo, Norway
†University of Oslo, Department of Informatics, Pb. 1080, 0316 Oslo, Norway
Email: \{aida.omerovic,ketil.stolen\}@sintef.no

Abstract—Our earlier research indicated the feasibility of the PREDIQT method for model-based prediction of impacts of architectural design changes, on the different quality characteristics of a system. The PREDIQT method develops and makes use of a multi-layer model structure, called prediction models. Usefulness of the prediction models requires a structured documentation of both the relations between the prediction models and the rationale and assumptions made during the model development. This structured documentation is what we refer to as trace-link information. In this paper, we propose a traceability scheme for PREDIQT, and an implementation of it in the form of a prototype tool which can be used to define, document, search for and represent the trace-links needed.

The solution is applied on prediction models from an earlier PREDIQT-based analysis conducted on a real-life system. Based on a set of success criteria, we argue that our traceability approach is useful and practically scalable in the PREDIQT context.

Keywords—traceability; system quality prediction; modeling; architectural design; change impact analysis; simulation.

I. INTRODUCTION

We have developed and tried out the PREDIQT method [1] [2] aimed for predicting impacts of architectural design changes on system quality characteristics and their trade-offs. Examples of quality characteristics include availability, scalability, security and reliability.

Important preconditions for model-based prediction are correctness and proper usage of the prediction models. The process of the PREDIQT method guides the development and use of the prediction models, but the correctness of the prediction models and the way they are applied are also highly dependent on the creative effort of the analyst and his/her helpers. In order to provide additional help and guidance to the analyst, we propose in this paper a traceability approach for documenting and retrieving the rationale and assumptions made during the model development, as well as the dependencies between the elements of the prediction models.

The approach is defined by a traceability scheme, which is basically a feature diagram specifying capabilities of the solution and a meta-model for the trace-link information. A prototype tool is implemented in the form of a relational database with user interfaces which can be employed to define, document, search for and represent the trace-links needed. The solution is illustrated on prediction models from an earlier PREDIQT-based analysis conducted on a real-life system [3]. We argue that our approach is, given the success criteria for traceability in PREDIQT, practically useful and better than any other traceability approach we are aware of.

The paper is organized as follows: Section II provides background on traceability. The challenge of traceability handling in the context of the PREDIQT method is characterized in Section III. Our traceability handling approach is presented in Section IV. Section V illustrates the approach on an example. Section VI argues for completeness and practicability of the approach, by evaluating it with respect to the success criteria. Section VII substantiates why our approach, given the success criteria outlined in Section III, is preferred among the alternative traceability approaches. The concluding remarks and future work are presented in Section VIII. An overview of the PREDIQT method is provided in Appendix 1. Guidelines for application of both the prediction models and the trace-link information are provided in Appendix 2.

II. BACKGROUND ON TRACEABILITY

Traceability is the ability to determine which documentation entities of a software system are related to which other documentation entities according to specific relationships [4]. IEEE [5] also provides two definitions of traceability:

1) Traceability is the degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another; for example, the degree to which the requirements and design of a given software component match.

2) Traceability is the degree to which each element in a software development product establishes its reason for existing.

Traceability research and practice are most established in fields such as requirements engineering and model-driven engineering (MDE). Knethen and Paech [4] argue: “Dependency analysis approaches provide a fine-grained impact analysis but can not be applied to determine the impact of a required change on the overall software system. An imprecise impact analysis results in an imprecise estimate of costs and increases the effort that is necessary to implement a required change because precise relationships have to be identified during changing. This is cost intensive and error
prone because analyzing the software documents requires detailed understanding of the software documents and the relationships between them." Aizenbud-Reshef et al. [6] furthermore state: “The extent of traceability practice is viewed as a measure of system quality and process maturity and is mandated by many standards” and “With complete traceability, more accurate costs and schedules of changes can be determined, rather than depending on the programmer to know all the areas that will be affected by these changes”.

IEEE [5] defines a trace as “A relationship between two or more products of the development process.” According to the OED [7], however, a trace is defined more generally as a “(possibly) non-material indication or evidence showing what has existed or happened”. As argued by [8]: “If a developer works on an artifact, he leaves traces. The software configuration management system records who has worked on the artifact, when that person has worked on it, and some systems also record which parts of the artifacts have been changed. But beyond this basic information, the changes themselves also reflect the developer’s thoughts and ideas, the thoughts and ideas of other stakeholders he may have talked to, information contained in other artifacts, and the transformation process that produced the artifact out of these inputs. These influences can also be considered as traces, even though they are usually not recorded by software configuration management systems.”

A traceability link is a relation that is used to interrelate artifacts (e.g., by causality, content, etc.) [8]. In the context of requirements traceability, [8] argues that “a trace can in part be documented as a set of meta-data of an artifact (such as creation and modification dates, creator, modifier, and version history), and in part as relationships documenting the influence of a set of stakeholders and artifacts on an artifact. Particularly those relationships are a vital concept of traceability, and they are often referred to as traceability links. Traceability links document the various dependencies, influences, causalities, etc. that exist between the artifacts. A traceability link can be unidirectional (such as depends-on) or bidirectional (such as alternative-for). The direction of a link, however, only serves as an indication of order in time or causality. It does not constrain its (technical) navigability, so traceability links can always be followed in both directions”.

In addition to the different definitions, there is no commonly agreed basic classification [8]. A taxonomy of the main concepts within traceability is suggested by [4].

An overview of the current state of traceability research and practice in requirements engineering and model-driven development is provided by [8], based on an extensive literature survey. Another survey [9] discusses the state-of-the-art in traceability approaches in MDE and assesses them with respect to five evaluation criteria: representation, mapping, scalability, change impact analysis and tool support. Moreover, Spanoudakis and Zisman [10] present a roadmap of research and practices related to software traceability and identify issues that are open for further research. The roadmap is organized according to the main topics that have been the focus of software traceability research.

Traces can exist between both model- and non-model artifacts. The means and measures applied for obtaining traceability are defined by so-called traceability schemes. A traceability scheme is driven by the planned use of the traces. The traceability scheme determines for which artifacts and up to which level of detail traces can be recorded [8]. A traceability scheme thus defines the constraints needed to guide the recording of traces, and answers the core questions: what, who, where, how, when and why. Additionally, there is tacit knowledge (such as why), which is difficult to capture and to document. A traceability scheme helps in this process of recording traces and making them persistent.

As argued by [6], the first approach used to express and maintain traceability was cross-referencing. This involves embedding phrases like “see section x” throughout the project documentation. Thereafter, different techniques have been used to represent traceability relationships including standard approaches such as matrices, databases, hypertext links, graph-based approaches, formal methods, and dynamic schemes [6]. Representation, recording and maintenance of traceability relations are by Spanoudakis and Zisman [10] classified into five approaches: single centralized database, software repository, hypermedia, markup, and event-based.

According to Wieringa [11], representations and visualizations of traces can be categorized into matrices, cross-references, and graph-based representations. As elaborated by Wieringa, the links, the content of the one artifact, and other information associated with a cross reference, is usually displayed at the same time. This is however not the case with traceability matrices. So, compared to traceability matrices, the user is (in the case of cross-references) shown more local information at the cost of being shown fewer (global) links. As models are the central element in MDE, graph-based representations are the norm. A graph can be transformed to a cross-reference. Regarding the notation, there is, however, no common agreement or standard, mostly because the variety and informality of different artifacts is not suitable for a simple, yet precise notation. Requirements traceability graphs are usually just plain box-and-line diagrams [11].

Kneten and Paech [4] argue that the existing traceability approaches do not give much process support. They specify four steps of traceability process: 1) define entities and relationships, 2) capture traces, 3) extract and represent traces, and 4) maintain traces. Similarly, Winkler and Pilgrim [8] state that traceability and its supporting activities are currently not standardized. They classify the activities when working with traces into: 1) planning for traceability, 2) recording traces, 3) using traces, and 4) maintaining traces. Traceability activities are generally not dependent on any
particular software process model.

Trace models are usually stored as separate models, and links to the elements are (technically) unidirectional in order to keep the connected models or artifacts independent. Alternatively, models can contain the trace-links themselves and links can be defined as bidirectional. While embedded trace-links pollute the models, navigation is much easier [8]. Thus, we distinguish between external and internal storage, respectively. Anquetil et al. [12] argue: “Keeping link information separated from the artifacts is clearly better; however it needs to identify uniquely each artifact, even fine-grained artifacts. Much of the recent research has focused on finding means to automate the creation and maintenance of trace information. Text mining, information retrieval and analysis of trace links techniques have been successfully applied. An important challenge is to maintain links consistency while artifacts are evolving. In this case, the main difficulty comes from the manually created links, but scalability of automatic solution is also an issue.”

As outlined by [6], automated creation of trace-links may be based on text mining, information retrieval, analysis of existing relationships to obtain implied relations, or analysis of change history to automatically compute links.

Reference models are an abstraction of best practice and comprise the most important kinds of traceability links. There is nothing provably correct about reference models, but they derive their relevance from the slice of practice they cover. Nevertheless, by formalizing a reference model in an appropriate framework, a number of elementary desirable properties can be ensured. A general reference model for requirements traceability is proposed by [13], based on numerous empirical studies.

Various tools are used to set and maintain traces. Surveys of the tools available are provided by [4], [8], [10] and [6]. Bohner and Arnold [14] found that the granularity of documentation entities managed by current traceability tools is typically somewhat coarse for an accurate impact analysis.

III. THE CHALLENGE

The PREDIQT process consists of three overall phases: Target modeling, Verification of prediction models, and Application of prediction models. Three interrelated sets of models are developed during the process of the PREDIQT method: Design Model which specifies system architecture, Quality Model which specifies the system quality notions, and Dependency Views (DVs) which represent the interrelationship between the system quality and the architectural design.

Trace-link information can be overly detailed and extensive while the solution needed in a PREDIQT context has to be applicable in a practical real-life setting within the limited resources allocated for a PREDIQT-based analysis. Therefore, the traceability approach should provide sufficient breadth and accuracy for documenting, retrieving and representing of the trace-links, while at the same time being practically applicable in terms of comprehensibility and scalability. The right balance between the completeness and accuracy of the trace information on the one side, and practical usability of the approach on the other side, is what characterizes the main challenge in proposing the appropriate solution for traceability handling in PREDIQT. Therefore, the trace-link creation efforts have to be concentrated on the traces necessary during the application of the prediction models.

A. Structure of the prediction models

Figure 1 provides an overview of the elements of the prediction models, expressed as a UML [15] class diagram. A Quality Model is a set of tree-like structures which clearly specify the system-relevant quality notions, by defining and decomposing the meaning of the system-relevant quality terminology. Each tree is dedicated to a target system-relevant quality characteristic. Each quality characteristic may be decomposed into quality sub-characteristics, which in turn may be decomposed into a set of quality indicators. As indicated by the relationship of type aggregation, specific sub-characteristics and indicators can appear in several Quality Model trees dedicated to the different quality characteristics. Each element of a Quality Model is assigned a quantitative normalized metric and an interpretation (qualitative meaning of the element), both specific for the target system. A Design Model represents the relevant aspects of the system architecture, such as for example process, dataflow, structure and rules.

A DV is a weighted dependency tree dedicated to a specific quality characteristic defined through the Quality Model. As indicated by the attributes of the Class Node, the nodes of a DV are assigned a name and a QCF (Quality Characteristic Fulfillment). A QCF is value of the degree of fulfillment of the quality characteristic, with respect to what is represented by the node. The degree of fulfillment is defined by the metric (of the quality characteristic) provided in the Quality Model. Thus, a complete prediction model has as many DVs as the quality characteristics defined in the Quality Model. Additionally, as indicated by the Semantic dependency relationship, semantics of both the structure and the weights of a DV are given by the definitions of the quality characteristics, as specified in the Quality Model. A DV node may be based on a Design Model element, as indicated by the Based on dependency relationship. As indicated by the self-reference on the Node class, one node may be decomposed into children nodes. Directed arcs express dependency with respect to quality characteristic by relating each parent node to its immediate children nodes, thus forming a tree structure. Each arc in a DV is assigned an EI (Estimated Impact), which is a normalized value of degree of dependence of a parent node, on the immediate child node. Thus, there is a quantified depen-
The intended application of the prediction models does not assume implementation of change on the target system, but only simulation of effects of the independent architectural design changes quality of the system (in its currently modelled state). Since the simulation is only performed on the target system in its current state and the changes are simulated independently (rather than incrementally), versioning of both prediction models and trace information is beyond the scope of PREDIQT. A more detailed overview of the PREDIQT method and the prediction models, is provided in Appendix 1.

B. Success criteria

It is, as argued by [8], an open issue to match trace usage and traceability schemes, and to provide guidance to limit and fit traceability schemes in a such way that they match a projects required usage scenarios for traces. One of the most urgent questions is which requirements a single scenario imposes on the other activities (in particular planning and recording) in the traceability process.

Moreover, it is argued by Aizenbud-Reshef et al. [6] that the lack of guidance as to what link information should be produced and the fact that those who use traceability are commonly not those producing it, also diminishes the motivation of those who create and maintain traceability information. In order to avoid this trap, we used the PREDIQT guidelines (as documented in Appendix 2) for the analyst as a starting point, for deriving the specific needs for traceability support. The guidelines are based on the authors’ experiences from industrial trials of PREDIQT [3] [2]. As such, the guidelines are not exhaustive but serve as an aid towards a more structured process of applying the prediction models and accommodating the trace information during the model development, based on the needs of the “Application of prediction models”-phase.

The specific needs for traceability support in PREDIQT are summarized below:

1) There is need for the following kinds of trace-links:
   • links between the Design Model elements to support identification of dependencies among the elements of the Design Model
   • links from the Design Model elements to DV elements to support identification of DV nodes which are based on specific elements of the Design Model
   • links from DV elements to Quality Model elements to support acquisition of traces from the prior estimates of the DV to the relevant quality indicators
   • links to external information sources (documents, measurement, domain experts) used during the development of DV structure and estimation of the parameters to support documenting the traces from the DV to the more detailed information sources available outside the prediction models.
   • links to rationale and assumptions for: Design Model elements, the semantics of the DV elements, as well as structure and prior parameter estimates of the DVs to support documenting the parts of the process of development of the prediction models, particularly the understanding and interpretations that the models are based on

2) The traceability approach should have facilities for both searching with model types and model elements as input parameters, as well as for reporting linked
elements and the link properties

3) The traceability approach should be flexible with respect to granularity of trace information

4) The traceability approach should be practically applicable on real-life applications of PREDIQT

These needs are in the sequel referred to as the success criteria for the traceability approach in PREDIQT.

IV. OUR SOLUTION

This section starts by presenting our traceability scheme for PREDIQT. Then, a prototype tool for trace-link management, implementing the needs specified through the traceability scheme, is presented.

A. Traceability scheme

We propose a traceability scheme in the form of a meta-model for trace-link information and a feature diagram for capabilities of the solution. The types of the trace-links and the types of the traceable elements are directly extracted from Success Criterion 1 and represented through a meta-model shown by Figure 2. The Element abstract class represents a generalization of a traceable element. The Element abstract class is specialized into the five kinds of traceable elements: Design Model Element, DV Element, Quality Model Element, External Information Source, and Rationale and Assumptions. Similarly, the Trace Link abstract class represents a generalization of a trace-link and may be assigned a rationale for the trace-link. The Trace Link abstract class is specialized into the six kinds of trace-links.

Pairs of certain kinds of traceable elements form binary relations in the form of unidirectional trace-links. Such relations are represented by the UML-specific notations called association classes (a class connected by a dotted line to a link which connects two classes). For example, trace-links of type Design Model Element to Design Model Element may be formed from a Design Model Element to a Dependency View Element. The direction of the link is annotated by the origin (the traceable element that the trace-link goes from) and the target (the traceable element that the trace-link goes to). Since only distinct pairs (single instances) of the traceable elements (of the kinds involved in the respective trace-links defined in Figure 2) can be involved in the associated specific kinds of trace-links, uniqueness (property of UML association classes) is present in the defined trace-links. Due to the binary relations (arity of value 2) in the defined trace-links between the traceable elements, only two elements can be involved in any trace-link. Furthermore, multiplicity of all the traceable elements involved in the trace-links defined is of type “many”, since an element can participate in multiple associations (given they are defined by the meta-model and unique).

The main capabilities needed are represented through a feature diagram [8] shown by Figure 3. Storage of trace-links may be internal or external, relative to the prediction models. A traceable element may be of type prediction model element (see Figure 1) or non-model element. Reporting and searching functionality has to be supported. Trace-link info has to include link direction, link meta-data (e.g. date, creator, strength) and cardinality (note that all links are binary, but a single element can be origin or target for more than one trace-link). Typing at the origin and the target ends of a trace-link as well as documenting rationale for trace-link, are optional.

B. Prototype traceability tool

We have developed a prototype tool in the form of a database application with user interfaces, on the top of MS Access [16]. The prototype tool includes a structure of tables for organizing the trace information, queries for retrieval of the trace info, a menu for managing work flow, forms for populating trace-link information, and facilities for reporting trace-links. A screen shot of the entity-relationship (ER)
diagram of the trace-link database is shown by Figure 4. The ER diagram is normalized, which means that the data are organized with minimal needs for repeating the entries in the tables. Consistency checks are performed on the referenced fields. The data structure itself (represented by the ER diagram) does not cover all the constraints imposed by the meta-model (shown by Figure 2). However, constraints on queries and forms as well as macros can be added in order to fully implement the logic, such as for example which element types can be related to which trace-link types.

The five traceable element types defined by Figure 2 and their properties (name of creator, date, assumption and comment), are listed in Table TraceableElementType. Similarly, the six trace-link types defined by Figure 2 and their properties (scope, date, creator and comment), are listed in Table TraceLinkType. Table TraceableElement specifies the concrete instances of the traceable elements, and assigns properties (such as the pre-defined element type, hyperlink, creator, date, etc.) to each one of them. Since primary key attribute in Table TraceableElementType is foreign key in Table TraceableElement, multiplicity between the two respective tables is one-to-many.

Most of the properties are optional, and deduced based on: 1) the core questions to be answered by traceability scheme [8] and 2) the needs for using guidelines for application of prediction models, specified in Appendix 2. The three Tables TargetElements, OriginElements and TraceLink together specify the concrete instances of trace-links. Each link is binary, and directed from a concrete pre-defined traceable element – the origin element specified in Table OriginElements, to a concrete pre-defined traceable element – the target element specified in Table TargetElements. The trace-link itself (between the origin and the target element) and its properties (such as pre-defined trace-link type) are specified in Table TraceLink. Attribute TraceLinkName (associated with a unique TraceLinkId value) connects the three tables TraceLink, OriginElements and TargetElements when representing a single trace-link instance, thus forming a
cross-product when relating the three tables. The MS Access environment performs reference checks on the cross-products, as well as on the values of the foreign key attributes. Target elements and origin elements participating in a trace-link, are instances of traceable elements defined in Table TraceableElement. They are connected through the Attribute ElementId (displayed as ElementName in the tables where it has the role of foreign key). Thus, multiplicity between Table TraceableElement and Table TargetElements, as well as between Table TraceableElement and Table OriginElements, is one-to-many. Similarly, since primary key attribute in Table TraceLinkType is foreign key in Table TraceLink, multiplicity between the two respective tables is one-to-many.

A screen shot of the start menu is shown by Figure 5. The sequence of the buttons represents a typical sequence of actions of an end-user (the analyst), in the context of defining, documenting and using the trace-links. The basic definition of the types of the traceable elements and the trace-links are provided first. Then, concrete traceable elements are documented, before defining specific instances of the trace-links and their associated specific origin and target elements, involved in the binary trace-link relations. Finally, reports can be obtained, based on search parameters such as for example model types, model elements, or trace-link types.

V. APPLYING THE SOLUTION ON AN EXAMPLE

This section exemplifies the application of our solution for managing traces in the context of prediction models earlier developed and applied during a PREDIQT-based analysis [3] conducted on a real-life system.

The trace-link information was documented in the prototype tool, in relation to the model development. The trace-links were applied during change application, according to the guidelines for application of prediction models, specified in Appendix 2. We present the experiences obtained, while the process of documentation of the trace-links is beyond the scope of this paper.

The prediction models involved are the ones related to “Split signature verification component into two redundant components, with load balancing”, corresponding to Change 1 in [3]. Three Design Model diagrams were affected, and one, two and one model element on each, respectively. We have tried out the prototype traceability tool on the Design Model diagrams involved, as well as Availability (which was one of the three quality characteristics analyzed) related Quality Model diagrams and DV. Documentation of the trace-links involved within the Availability quality characteristic (as defined by the Quality Model) scope, took approximately three hours. Most of the time was spent on actually typing the names of the traceable elements and the trace-links.

18 instances of traceable elements were registered in the database during the trial: seven Quality Model elements, four DV elements, four Design Model elements and three elements of type “Rationale and Assumptions”. 12 trace-links were recorded: three trace-links of type “Design Model Element to Design Model Element”, three trace-links of type “Design Model Element to DV Element”, one trace-link of type “Design Model Element to Rationale and Assumptions”, three trace-links of type “DV Element to Quality Model Element”, and two trace-links of type “Structure, Parameter or Semantics of DV Element Documented through Rationale and Assumptions”, were documented.

An extract of a screen shot of a trace-link report (obtained from the prototype tool) is shown by Figure 6. The report included: three out of three needed (i.e., actually existing, regardless if they are recorded in the trace-link database) “Design Model Element to Design Model Element” links,
three out of four needed “Design Model Element to DV Element” links, one out of one needed “Design Model Element to Rationale and Assumptions” link, three out of six needed “DV Element to Quality Model Element” links and one out of one needed “Structure, Parameter or Semantics of DV Element Documented through Rationale and Assumptions” link.

Best effort was made to document the appropriate trace-links without taking into consideration any knowledge of exactly which of them would be used when applying the change. The use of the trace-links along with the application of change on the prediction models took totally 20 minutes and resulted in the same predictions (change propagation paths and values of QCF estimates on the Availability DV), as in the original case study [3]. Without the guidelines and the trace-link report, the change application would have taken approximately double time for the same user.

All documented trace-links were relevant and used during the application of the change, and about 73% of the relevant trace-links could be retrieved from the prototype tool. Considering however the importance and the role of the retrievable trace-links, the percentage should increase considerably.

Although hyperlinks are included as meta-data in the user interface for element registration, an improved solution should include interfaces for automatic import of the element names from the prediction models, as well as user interfaces for easy (graphical) trace-link generations between the existing elements. This would also aid verification of the element names.

VI. WHY OUR SOLUTION IS A GOOD ONE

This section argues that the approach presented above fulfills the success criteria specified in Section III.

A. Success Criterion 1

The traceability scheme and the prototype tool capture the kinds of trace-links and traceable elements, specified in the Success Criterion 1. The types of trace-links and traceable elements as well as their properties, are specified in dedicated tables in the database of the prototype tool. This allows constraining the types of the trace-links and the types of the traceable elements to only the ones defined, or extending their number or definitions, if needed. The trace-links in the prototype tool are binary and unidirectional, as required by the traceability scheme. Macros and constraints can be added in the tool, to implement any additional logic regarding trace-links, traceable elements, or their respective type definitions and relations. The data properties (e.g. date, hyperlink or creator) required by the user interface, allow full traceability of the data registered in the database of the prototype tool.

B. Success Criterion 2

Searching based on user input, selectable values from a list of pre-defined parameters, or comparison of one or more database fields, are relatively simple and fully supported based on queries in MS Access. Customized reports can be produced with results of any query and show any information registered in the database. The report, an extract of which is presented in Section V, is based on a query of all documented trace-links and the related elements.

C. Success Criterion 3

The text-based fields for documenting the concrete instances of the traceable elements and the trace-links, allow level of detail selectable by the user. Only a subset of fields is mandatory for providing the necessary trace-link data. The optional fields in the tables can be used for providing additional information such as for example rationale, comments, links to external information sources, attachments, strength or dependency. There are no restrictions as to what can be considered as a traceable element, as long as it belongs to one of the element types defined by Figure 2. Similarly, there are no restrictions as to what can be considered as a trace-link, as long as it belongs to one of the trace-link types defined by Figure 2. The amount of information provided regarding the naming and the meta-data, are selectable by the user.

D. Success Criterion 4

Given the realism of the prediction models involved in the example, the size and complexity of the target system they address, the representativeness of the change applied on them, the simplicity of the prototype tool with respect to both the user interfaces and the notions involved, as well as the time spent on documenting the trace-links and using them, the application of the approach presented in Section V indicates the applicability of our solution on real-life applications of PREDIQT, with limited resources and by an average user (in the role of the analyst).

The predictions (change propagation paths and values of QCF estimates) we obtained during the application of our solution on the example were same as the ones from the original case study [3] (performed in year 2008) which the models stem from. Although the same analyst has been involved in both, the results suggest that other users should, by following PREDIQT guidelines and applying the prototype traceability tool, obtain similar results.

The time spent is to some degree individual and depends on the understanding of the target system, the models and the PREDIQT method. It is unknown if the predictions would have been the same (as in the original case study) for another user. We do however consider the models and the change applied during the application of the solution, to be representative due to their origins from a major real-life system. Still, practical applicability of our solution will be subject to future empirical evaluations.
VII. WHY OTHER APPROACHES ARE NOT BETTER IN THIS CONTEXT

This section evaluates the feasibility of other traceability approaches in the PREDIQT context. Based on our review of the approach-specific publications and the results of the evaluation by Galvao and Goknil [9] of a subset of the below mentioned approaches, we argue why the alternative traceability approaches do not perform sufficiently on one or more of the success criteria specified in Section III. The evaluation by Galvao and Goknil is conducted with respect to five criteria: 1) structures used for representing the traceability information; 2) mapping of model elements at different abstraction levels; 3) scalability for large projects in terms of process, visualization of trace information, and application to a large amount of model elements; 4) change impact analysis on the entire system and across the software development lifecycle; and 5) tool support for visualization and management of traces, as well as for reasoning on the trace-link information.

Almeida et al. [17] propose an approach aimed at simplifying the management of relationships between requirements and various design artifacts. A framework which serves as a basis for tracing requirements, assessing the quality of model transformation specifications, meta-models, models and realizations, is proposed. They use traceability crosstable for representing relationships between application requirements and models. Cross-tables are also applied for considering different model granularities and identification of conforming transformation specifications. The approach does not provide sufficient support for intra-model mapping, thus failing on our Success Criterion 1. Moreover, possibility of representing the various types of trace-links and traceable elements is unclear, although different visualizations on a cross-table are suggested. Tool support is not available, which limits applicability of the approach in a practical setting. Searching and reporting facilities are not available. Thus, it fails on our Success Criteria 1, 2 and 4.

Event-based Traceability (EBT) is another requirements-driven traceability approach aimed at automating trace-link generation and maintenance. Cleland-Huang, Chang and Christensen [18] present a study which uses EBT for managing evolutionary change. They link requirements and other traceable elements, such as design models, through publish-subscribe relationships. As outlined by [9], “Instead of establishing direct and tight coupled links between requirements and dependent entities, links are established through an event service. First, all artefacts are registered to the event server by their subscriber manager. The requirements manager uses its event recognition algorithm to handle the updates in the requirements document and to publish these changes as event to the event server. The event server manages some links between the requirement and its dependent artefacts by using some information retrieval algorithms.” The notification of events carries structural and semantic information concerning a change context. Scalability in a practical setting is the main issue, due to performance limitation of the EBT server [9]. Moreover, the approach does not provide sufficient support for intra-model mapping. Thus, it fails on our Success Criteria 1 and 4.

Cleland-Huang et al. [19] propose Goal Centric Traceability (GCT) approach for managing the impact of change upon the non-functional requirements of a software system. Softgoal Interdependency Graph (SIG) is used to model non-functional requirements and their dependencies. Additionally, a traceability matrix is constructed to relate SIG elements to classes. The main weakness of the approach is the limited tool support, which requires manual work. This limits both scalability in a practical setting and searching support (thus failing on our Success Criteria 4 and 2, respectively). It is unclear to what degree granularity of the approach would suffice the needs of PREDIQT.

Cleland-Huang and Schmelzer [20] propose another requirements-driven traceability approach that builds on EBT. The approach involves a different process for dynamically tracing non-functional requirements to design patterns. Although more fine-grained than EBT, there is no evidence that the method can be applied with success in a practical real-life setting (required through our Success Criterion 4). Searching and reporting facilities (as required through our Success Criterion 2) are not provided.

Many traceability approaches address trace maintenance. Cleland-Huang, Chang and Ge [21] identify the various change events that occur during requirements evolution and describe an algorithm to support their automated recognition through the monitoring of more primitive actions made by a user upon a requirements set. Mader and Gotel [22] propose an approach to recognize changes to structural UML models that impact existing traceability relations and, based on that knowledge, provide a mix of automated and semi-automated strategies to update the relations. Both approaches focus on trace maintenance, which is as argued in Section III, not among the traceability needs in PREDIQT.

Ramesh and Jarke [13] propose another requirements-driven traceability approach where reference models are used to represent different levels of traceability information and links. The granularity of the representation of traces depends on the expectations of the stakeholders [9]. The reference models can be implemented in distinct ways when managing the traceability information. As reported by [9], “The reference models may be scalable due to their possible use for traceability activities in different complexity levels. Therefore, it is unclear whether this approach lacks scalability with respect to tool support for large-scale projects or not. The efficiency of the tools which have implemented these meta-models was not evaluated and the tools are not the focus of the approach.” In PREDIQT context, the reference models are too broad, their focus is on requirements
traceability, and tool support is not sufficient with respect to searching and reporting (our Success Criterion 2).

We could however have tried to use parts of the reference models by Ramesh and Jarke [13] and provide tool support based on them. This is done by [23] in the context of product and service families. The authors discuss a knowledge management system, which is based on the traceability framework by Ramesh and Jarke [13]. The system captures the various design decisions associated with service family development. The system also traces commonality and variability in customer requirements to their corresponding design artifacts. The tool support has graphical interfaces for documenting decisions. The trace and design decision capture is illustrated using sample scenarios from a case study. We have however not been able to obtain the tool, in order to try it out in our context.

A modeling approach by Egyed [24] represents traceability information in a graph structure called a footprint graph. Generated traces can relate model elements with other models, test scenarios or classes [9]. Galvao and Goknil [9] report on promising scalability of the approach. It is however unclear to what degree the tool support fulfills our success criterion regarding searching and reporting, since semantic information on trace-links and traceable elements is limited.

Aizenbud-Reshef et al. [25] outline an operational semantics of traceability relationships that capture and represent traceability information by using a set of semantic properties, composed of events, conditions and actions [9]. Galvao and Goknil [9] state: the approach does not provide sufficient support for intra-model mapping; a practical application of the approach is not presented; tool support is not provided; however, it may be scalable since it is associated with the UML. Hence, it fails on our Success Criteria 1 and 2.

Limon and Garbajosa [26] analyze several traceability schemes and propose an initial approach to Traceability Scheme (TS) specification. The TS is composed of a traceability link dataset, a traceability link type set, a minimal set of traceability links, and a metrics set for the minimal set of traceability links [9]. Galvao and Goknil [9] argue that “The TS is not scalable in its current form. Therefore, the authors outline a strategy that may contribute to its scalability: to include in the traceability schema a set of metrics that can be applied for monitoring and verifying the correctness of traces and their management.” Hence, it fails with respect to scalability in a practical setting, that is, our criterion 4. Moreover, there is no tool support for the employment of the approach, which fails on our success criterion regarding searching and reporting.

Some approaches [27] [28] [29] that use model transformations can be considered as a mechanism to generate trace-links. Tool support with transformation functionalities is in focus, while empirical evidence of applicability and particularly comprehensibility of the approaches in a practical setting, is missing. The publications we have retrieved do not report sufficiently on whether these approaches would offer the searching facilities, the granularity of trace information, and the scalability needed for use in PREDIQT context (that is, in a practical setting by an end-user (analyst) who is not an expert in the tools provided).

VIII. CONCLUSION AND FUTURE WORK

Our earlier research indicates the feasibility of the PREDIQT method for model-based prediction of impacts of architectural design changes on system quality. The PREDIQT method produces and applies a multi-layer model structure, called prediction models, which represent system design, system quality and the interrelationship between the two.

Based on the success criteria for a traceability approach in the PREDIQT context, we put forward a traceability scheme. Based on this, a prototype tool which can be used to define, document, search for and represent the trace-links needed, is developed. We have argued that our solution offers a useful and practically applicable support for traceability in the PREDIQT context. The model application guidelines provided in Appendix 2 complement the prototype traceability tool and aim to jointly provide the facilities needed for a schematic application of prediction models.

Performing an analysis of factors such as cost, risk, and benefit and following the paradigm of value-based software-engineering, would be relevant in order to stress the effort on the important trace-links. As argued by [8], if the value-based paradigm is applied to traceability, cost, benefit, and risk will have to be determined separately for each trace according to if, when, and to what level of detail it will be needed later. This leads to more important artifacts having higher-quality traceability. There is a trade-off between the semantically accurate techniques on the one hand and cost-efficient but less detailed approaches on the other hand. Finding an optimal compromise is still a research challenge. Our solution proposes a feasible approach, while finding the optimal one is subject to further research.

Further empirical evaluation of our solution is also necessary to test its feasibility on different analysts as well as its practical applicability in the various domains which PREDIQT is applied on. Future work should also include standard interfaces and procedures for updating the traceable elements from the prediction models into our prototype traceability tool. As model application phase of PREDIQT dictates which trace-link information is needed and how it should be used, the current PREDIQT guidelines focus on the application of the prediction models. However, since the group of recorders and the group of users of traces may be distinct, structured guidelines for recording the traces during the model development should also be developed as a part of the future work.
ACKNOWLEDGMENT

This work has been conducted as a part of the DIGIT (180052/S10) project funded by the Research Council of Norway, as well as a part of the NESSoS network of excellence funded by the European Commission within the 7th Framework Programme.

REFERENCES


Phase 1: Target modeling
- Sub-phase 1: Characterization of the target and the objectives
- Sub-phase 2: Development of Quality Models
- Sub-phase 3: Mapping of Design Models
- Sub-phase 4: Development of Dependency Views

Phase 2: Verification of prediction models
- Sub-phase 1: Evaluation of prediction models
- Sub-phase 2: Fitting of prediction models
- Sub-phase 3: Approval of the final prediction models

Phase 3: Application of prediction models
- Sub-phase 1: Specification of a change
- Sub-phase 2: Application of the change on prediction models
- Sub-phase 3: Quality prediction

Figure 7. A simplified overview of the process of the PREDIQT method

APPENDIX 1: AN OVERVIEW OF THE PREDIQT METHOD

The PREDIQT method produces and applies a multi-layer model structure, called prediction models, which represent system relevant quality concepts (through “Quality Model”), architectural design (through “Design Model”), and the dependencies between architectural design and quality (through “Dependency Views”). The Design Model diagrams are used to specify the architectural design of the target system and the changes whose effects on quality are to be predicted. The Quality Model diagrams are used to formalize the quality notions and define their interpretations. The values and the dependencies modeled through the Dependency Views (DVs) are based on the definitions provided by the Quality Model. The DVs express the interplay between the system architectural design and the quality characteristics. Once a change is specified on the Design Model diagrams, the affected parts of the DVs are identified, and the effects of the change on the quality values are automatically propagated at the appropriate parts of the DV. This section briefly outlines the PREDIQT method in terms of the process and the artifacts. For further details on PREDIQT, see [1] [30] [2].

The process of the PREDIQT method consists of three overall phases. Each phase is decomposed into sub-phases, as illustrated by Figure 7. Based on the initial input, the stakeholders involved deduce a high level characterization of the target system, its scope and the objectives of the prediction analysis, by formulating the system boundaries, system context (including the usage profile), system lifetime and the extent (nature and rate) of design changes expected. Quality Model diagrams are created in the form of trees, by defining the quality notions with respect to the target system. The Quality Model diagrams represent a taxonomy with interpretations and formal definitions of system quality.
Data protection
QCF=0.94
Encryption
QCF=1.00
Authentication
QCF=0.95
Authorization
QCF=0.90
Other
QCF=0.90
EI=0.30
EI=0.25
EI=0.30
EI=0.15
EI=0.30
EI=0.15

Figure 8. Excerpt of an example DV with fictitious values

notions. The total quality of the system is decomposed into characteristics, sub-characteristics and quality indicators. The Design Model diagrams represent the architectural design of the system.

For each quality characteristic defined in the Quality Model, a quality characteristic specific DV is deduced from the Design Model diagrams and the Quality Model diagrams of the system under analysis. This is done by modeling the dependencies of the architectural design with respect to the quality characteristic that the DV is dedicated to, in the form of multiple weighted and directed trees. A DV comprises two notions of parameters:

1) EI: Estimated degree of Impact between two nodes, and
2) QCF: estimated degree of Quality Characteristic Fulfillment.

Each arc pointing from the node being influenced is annotated by a quantitative value of EI, and each node is annotated by a quantitative value of QCF.

Figure 8 shows an excerpt of an example DV with fictitious values. In the case of the Encryption node of Figure 8, the QCF value expresses the goodness of encryption with respect to the quality characteristic in question, e.g., security. A quality characteristic is defined by the underlying system specific Quality Model, which may for example be based on the ISO 9126 product quality standard [31]. A QCF value in a DV expresses to what degree the node (representing system part, concern or similar) is realized so that it, within its own domain, fulfills the quality characteristic. The QCF value is based on the formal definition of the quality characteristic (for the system under analysis), provided by the Quality Model. The EI value on an arc expresses the degree of impact of a child node (which the arc is directed to) on the parent node, or to what degree the parent node depends on the child node, with respect to the quality characteristic under consideration.

“Initial” or “prior” estimation of a DV involves providing QCF values to all leaf nodes, and EI values to all arcs. Input to the DV parameters may come in different forms (e.g., from domain expert judgments, experience factories, measurements, monitoring, logs, etc.), during the different phases of the PREDIQT method. The DV parameters are assigned by providing the estimates on the arcs and the leaf nodes, and propagating them according to the general DV propagation algorithm. Consider for example the Data protection node in Figure 8 (denoting: DP: Data protection, E: Encryption, AT: Authentication, AAT: Authorization, and O:Other):

$$QCF_{(DP \rightarrow E)} = QCF_{(E)} \cdot EI_{(DP \rightarrow E)} + QCF_{(AT)} \cdot EI_{(DP \rightarrow AT)} + QCF_{(O)} \cdot EI_{(DP \rightarrow O)} \quad \text{Eq. 1}$$

The DV-based approach constrains the QCF of each node to range between 0 and 1, representing minimal and maximal characteristic fulfillment (within the domain of what is represented by the node), respectively. This constraint is ensured through the formal definition of the quality characteristic rating (provided in the Quality Model). The sum of EIs, each between 0 (no impact) and 1 (maximum impact), assigned to the arcs pointing to the immediate children must be 1 (for model completeness purpose). Moreover, all nodes having a common parent have to be orthogonal (independent). The dependent nodes are placed at different levels when structuring the tree, thus ensuring that the needed relations are shown at the same time as the tree structure is preserved.

The general DV propagation algorithm, exemplified by Eq. 1, is legitimate since each quality characteristic DV is complete, the EIs are normalized and the nodes having a common parent are orthogonal due to the structure. A DV is complete if each node which is decomposed, has children nodes which are independent and which together fully represent the relevant impacts on the parent node, with respect to the quality characteristic that the DV is dedicated to.

The rationale for the orthogonality is that the resulting DV structure is tree-formed and easy for the domain experts to relate to. This significantly simplifies the parametrization and limits the number of estimates required, since the number of interactions between the nodes is minimized. Although the orthogonality requirement puts additional demands on the DV structuring, it has shown to represent a significant advantage during the estimation.

The “Verification of prediction models” is an iterative phase that aims to validate the prediction models, with respect to the structure and the individual parameters, before they are applied. A measurement plan with the necessary statistical power is developed, describing what should be evaluated, when and how. Both system-as-is and change effects should be covered by the measurement plan. Model fitting is conducted in order to adjust the DV structure and the parameters to the evaluation results. The objective of the “Approval of the final prediction models” sub-phase is to evaluate the prediction models as a whole and validate that they are complete, correct and mutually consistent after the fitting. If the deviation between the model and the new measurements is above the acceptable threshold after the fitting, the target modeling phase is re-initiated.

The “Application of the change on prediction models”
phase involves applying the specified architectural design change on the prediction models. During this phase, a specified change is applied to the Design Model diagrams and the DVs, and its effects on the quality characteristics at the various abstraction levels are simulated on the respective DVs. When an architectural design change is applied on the Design Model diagrams, it is according to the definitions in the Quality Model, reflected to the relevant parts of the DV. Thereafter, the DV provides propagation paths and quantitative predictions of the new quality characteristic values, by propagating the change throughout the rest of each one of the modified DVs, based on the general DV propagation algorithm. We have earlier developed tool support [3] (below referred to as the “DV tool”) based on MS Excel for simulation and sensitivity analysis of DVs.

APPENDIX 2: GUIDELINES FOR APPLICATION OF PREDICTION MODELS

In order to facilitate quality and correct use of prediction models, this section provides guidelines for application of the prediction models and the trace-link information, with the analyst as the starting point. Thus, unless otherwise specified, all the guidelines are directed towards the analyst. Overall guidelines for the “Application of prediction models” – phase (see Figure 7) are presented first, followed by detailed guidelines for each one of its sub-phases: ‘Specification of a change’, “Application of the change on prediction models” and “Quality prediction”, respectively.

Guidelines for the “Application of prediction models” – phase

Objective During this phase, a specified change is applied to the prediction models, and its effects on the quality characteristics at the various abstraction levels are simulated on the respective Dependency Views (DV). The simulation reveals which design parts and aspects are affected by the change and the degree of impact (in terms of the quality notions defined by the Quality Model).

Prerequisites The fitted prediction models are approved. The changes applied are assumed to be independent relative to each other. The "Quality prediction" sub-phase presupposes that the change specified during the "Specification of a change" sub-phase can be fully applied on the prediction models, during the "Application of the change on prediction models" sub-phase.

How conducted This phase consists of the three sub-phases:
1) Specification of a change
2) Application of the change on prediction models
3) Quality prediction

Input documentation Prediction models: Design Model diagrams, Quality Model diagrams and Dependency Views; Trace-links.

Output documentation Change specification; Pre- and post-change Design Model diagrams; DVs.

People that should participate
- Analysis leader (Required). Analysis leader is also referred to as analyst.
- Analysis secretary (Optional)
- Representatives of the customer:
  - Decision makers (Optional)
  - Domain experts (Required)
  - System architects or other potential users of PREDIQT (Required)

Modeling guideline
1) Textually specify the architectural design change of the system.
2) Modify the Design Model diagrams with respect to the change proposed. Modify the structure and the values of the prior parameters, on the affected parts of the DVs.
3) Run the simulation and display the changes on the Design Model diagrams and the DVs, relative to their original (pre-change) structure and values.

Guidelines for the “Specification of a change” sub-phase

Objective The change specification should clearly state all deployment relevant facts necessary for applying the change on the prediction models. The specification should include the current and the new state and characteristics of the design elements/properties being changed, the rationale and the assumptions made.

Prerequisites The fitted prediction models are approved.

How conducted Specify the change by describing type of change, the rationale, who should perform it, when, how and in which sequence of events. In case change specification addresses modification of specific elements of the Design Model diagrams or the DVs, the quality characteristics of the elements before and after the change have to be specified, based on the definitions provided by the Quality Model. The change specification has to be at the abstraction level corresponding to the abstraction level of a sufficient subset of the Design Model diagrams or DVs.

Input documentation Prediction models: Design Model, Quality Model, Dependency Views.

Output documentation Textual specification of a change.

Modeling guideline
1) Textually specify an architectural design change of the system represented by the approved prediction models.
2) Specify the rationale and the process related to the change deployment.

Guidelines for the “Application of the change on prediction models” sub-phase

Objective This sub-phase involves applying the specified change on the prediction models.
**Prerequisites** The change is specified. The specified change is, by the analyst and the domain experts, agreed upon and a common understanding is reached.

**How conducted** Detailed instructions for performing the six steps specified in “Modeling guideline”, are provided here.

1) This first step of relating the change to the Design Model diagram(s) and their elements is a manual effort. The analyst and the domain experts confirm that a common understanding of the specification has been reached. Then, they retrieve the diagrams and the respective elements of the Design Model and identify which elements are potentially affected by the change, with respect to the system quality in general. The identified elements are marked, and their post-change status specified. The status may be of three types: update, delete or add. The update may involve change of a property related to design or a quality characteristic. In the case of delete, the diagram element is marked and its new status is visible. In case of add, a new diagram element is introduced.

2) The trace-links between diagrams and diagram elements are (during the "Target modeling" phase) documented in the form of a database, which they can be retrieved from. Each one of the above identified Design Model diagrams and diagram elements (except the added ones) is searched in the existing trace-link database (created during the model development). The result displays the searched items being in the role of the origin or the target element, and all the elements that depend on them or that they are dependent on, respectively. The result also displays overall meta-data, e.g. the kinds of the trace-links and their rationale. The retrieved (linked) elements are, by the domain experts and the analyst, considered whether they are affected by the specified change. Depending on the contents of the change and the trace-link type and rationale, each diagram or element which, according to the search results is linked to the elements identified in the previous step, may be irrelevant, deleted or updated. The updated and the deleted elements are, on the diagrams, assigned the new (post-change) status and meta-data.

3) Search in the trace-link database for all the above identified elements which have been updated or deleted, and retrieve their trace-links to the DV model elements. Manually identify the overall DV model elements that may be affected by the change. For all the retrieved and manually identified DV model elements, retrieve from the trace-link database, their rationale for the DV structure and the node semantics. Consider whether the added design element models require new DV nodes. Manually modify the DV structure, based on the retrieved trace-link information.

4) The domain experts and the analyst manually verify the updated structure (completeness, orthogonality and correctness) of each DVs, with respect to the 1) quality characteristic definitions provided by the Quality Model, and 2) the modified Design Model.

5) The estimates of the prior parameters have to be updated due to the modifications of the Design Model and the DV structure. Due do the structural DV modification in the previous step, previously internal nodes may have become prior nodes, and the EIs on the arcs may now be invalid. New nodes and arcs may have been introduced. All the earlier leaf nodes which have become internal nodes, and all new internal nodes are assumed to automatically be assigned the function for the propagation model, by the DV tool. All the new or modified arcs and leaf nodes have to be marked so that the values of their parameters can be evaluated. Manually identify the overall unmodified arcs and leaf nodes whose values have may have been affected by the change. In the case of the modified arcs and leaf nodes, trace-links are used to retrieve the previously documented rationale for the estimation of the prior parameter values and node semantics. The parameter values on the new and the modified arcs and leaf nodes are estimated manually based on the Quality Model. Estimate the leaf node QCFs of a sub-tree prior to estimating the related EIs. The rationale is to fully understand the semantics of the nodes, through reasoning about their QCFs first. In estimating a QCF, two steps have to be undergone:

   a) interpretation of the node in question – its contents, scope, rationale and relationship with the Design Model, and

   b) identification of the relevant metrics from the Quality Model of the quality characteristic that the DV is addressing, as well as evaluation of the metrics identified.

When estimating a QCF the following question is posed (to the domain experts): “To what degree is the quality characteristic fulfilled, given the contents and the scope of the node?” The definition of the rating should be recalled, along with the fact that zero estimate value denotes no fulfillment, while one denotes maximum fulfillment.

In estimating an EI, two steps have to be undergone:

   a) interpretation of the two nodes in question, and

   b) determination of the degree of impact of the child node, on the parent node. The value is assigned relative to the overall EIs related to the same parent node, and with a consistent unit of measure, prior to being normalized. The normalized EIs on the arcs from the same parent node have to
sum up to one, due to the requirement of model completeness.

When estimating an EI the following question is posed (to the domain experts): “To what degree does the child node impact the parent node, or how dependent is the parent node on child node, with respect to the quality characteristic that the DV is dedicated to?”

The definition of the quality characteristic provided by its Quality Model, should be recalled and the estimate is provided relative to the impact of the overall children nodes of the parent node in question. Alternatively, an impact value is assigned using the same unit of measure on all arcs of the sub-tree, and normalized thereafter.

Once one of the above specified questions is posed, depending on the kind of the DV parameter, the domain expert panel is asked to provide the estimate with an interval so that the correct value is within the interval with a probability given by the confidence level [30].

6) Manually verify the updated prior parameter values, so that the relative QCF values are consistent to each other and the rest of the estimates, and so that EIs on the arcs from a common parent sum up to one.

If the specified change can be fully applied, it is within the scope of the prediction models, which is a prerequisite for proceeding to the next sub-phase. Otherwise, the modifications are canceled and the change deemed not predictable by the models as such.

**Input documentation** Prediction models: Design Model, Quality Model, Dependency Views, change specification, trace-links.

**Output documentation** Design Model and DVs modified with respect to the change.

**Modeling guideline**

1) Relate the specified change to manually identifiable Design Model diagram(s) and their elements.

2) Use the trace-links to identify the affected parts (diagrams and diagram elements) of the Design Model. Apply the change by modifying (updating, deleting or adding) the identified affected parts of the Design Model.

3) Use the trace-links to identify the affected parts (nodes and dependency links) of each DV, by retrieving the traces from the modified and the deleted parts of the Design Model to the DVs, as well as the rationale for the DV structure and the node semantics. Modify the structure of the affected parts of the DVs.

4) Manually verify the updated structure (completeness, orthogonality and correctness) of the DVs, with respect to the Quality Model and the modified Design Model.

5) Use trace-links to identify the documented rationale for the estimation of the prior parameter values. Manually identify the overall prior parameters which have been affected by the change. Use Quality Model to modify the values of the affected prior parameters (i.e., Estimated Impact (EI) and leaf node Quality Characteristic Fulfillment (QCFs)).

6) Manually verify the updated prior parameter values (that QCFs are consistent relative to each other and that EIs on the arcs from a common parent sum up to one).

**Guidelines for the “Quality prediction” sub-phase**

**Objective** The propagation of the change throughout the rest of each one of the modified DVs, is performed. The propagation paths and the modified parameter values are obtained.

**Prerequisites** The specified change is within the scope of and fully applied on the prediction models.

**How conducted** Use the DV tool support to propagate the change. The tool explicitly displays the propagation paths and the modified parameter values, as well as the degrees of parameter value change. Obtain the predictions, in terms of the propagation paths and the parameter value modification. The result must explicitly express the changes with respect to the pre-change values. The propagation of the change throughout each one of the modified DVs, is performed based on the general DV propagation model, according to which the QCF value of each parent node is recursively calculated by first multiplying the QCF and EI value for each closest child and then summing up these products. Such a model is legitimate since each quality characteristic DV is complete, the EIs are normalized and the nodes having a common parent are orthogonal (with respect to the quality characteristic that the DV is dedicated to) due to the structure. The root node QCF values on the quality characteristic specific DVs represent the system-level rating value of the quality characteristic that the DV is dedicated to. If the predicted parameter values are beyond a pre-defined uncertainty threshold, the modifications are canceled and the change deemed not predictable by the input data and the models as such.

**Input documentation** DVs.

**Output documentation** The change is propagated throughout the DVs, based on the DV propagation model. Propagation paths and parameter value changes (relative to the original ones) are displayed.

**Modeling guideline**

1) Run the simulation on the DV tool, in order to obtain the change propagation paths and the modified QCF values of the affected non-leaf nodes of the DVs.

2) Display the changes performed on the Design Model and the DVs (structure and the prior parameter values).