

A Framework for Analyzing and Monitoring the Impact of Dependencies on Quality

Doctoral Dissertation by

Olav Skjelkvåle Ligaarden

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

October 2012

© Olav Skjelkvåle Ligaarden, 2013

*Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo
No. 1290*

ISSN 1501-7710

All rights reserved. No part of this publication may be
reproduced or transmitted, in any form or by any means, without permission.

Cover: Inger Sandved Anfinsen.
Printed in Norway: AIT Oslo AS.

Produced in co-operation with Akademika publishing.
The thesis is produced by Akademika publishing merely in connection with the
thesis defence. Kindly direct all inquiries regarding the thesis to the copyright
holder or the unit which grants the doctorate.

Abstract

In today's society, we are dependent on a number of services provided by interconnected systems. These services may be anything from electricity to services provided by social media platforms. Interconnected systems are challenging to analyze from a quality perspective in general and from a security perspective in particular. The systems depend on each other through services. Thus, the quality of services provided by one system is often directly linked to the quality of services provided by another. Moreover, the systems may be under different managerial control and within different jurisdictions, and the systems may evolve rapidly in a manner that may be difficult to predict. All of this makes it challenging to assess risk to the quality of services.

In this thesis we present a framework for analyzing and monitoring the impact of dependencies on quality. More specifically, the framework should be used in the context of interconnected systems to analyze and monitor the impact of service dependencies on quality of services. The framework is the result of the integration of three artifacts: (1) a method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators; (2) a method for capturing and monitoring the impact of service dependencies on the quality of provided services; and (3) an architectural pattern for constructing enterprise level monitoring tools based on indicators. The three artifacts may be viewed as contributions on their own, since they can be used independently of each other. In addition, the thesis contributes in terms of two industrial case studies: (1) an empirical study on trust-based decisions in interconnected systems; and (2) an empirical study on the design of indicators for monitoring risk. The industrial case studies have mainly been carried out to support the development of the artifacts, but since the industrial case studies also provide insight into issues of a more general nature, they may be seen as contributions on their own.

Acknowledgements

First of all, I would like to express my deepest gratitude to my main supervisor Ketil Stølen for all his encouragement and involvement. I thank him for sharing his extensive knowledge and passion for science, for all the interesting discussions, for everything I have learnt from him, and for all the useful advice and feedback he has provided on my work. The numerous hours he has spent on supervising my work, far exceeds what can reasonably be expected from any supervisor. He has been an excellent supervisor, and I will without doubt recommend Ketil Stølen as a supervisor to anyone who wants to pursue a master's degree or a PhD within his fields of expertise.

I also wish to express my gratitude to my second supervisor Birger Møller-Pedersen for all the valuable advice, feedback, and support I have received from him during his participation in the DIGIT project.

I am grateful to Atle Refsdal who co-authored all of the papers in this thesis. I would like to thank him for the collaboration and for all the feedback, support, and advice he has provided.

I have also co-authored papers with Tormod Vaksvik Håvaldsrud, Mass Soldal Lund, Fredrik Seehusen, Jon Ølnes, and Per Myrseth. I wish to thank them all for their collaboration and support.

I would also like to thank guest scientists in the DIGIT and EMERGENCY projects: Fabio Massacci, Christian Damsgaard Jensen, Audun Jøssang, late William Winsborough, Jonas Landgren, and Isabelle Simplot-Ryl. I am also thankful to the members of the DIGIT advisory board. Both the guest scientists and the members of the advisory board have on a number of occasions provided useful feedback and advice.

I have on many occasions presented my work to Aida Omerovic, André Alexander-Haug, Erik Gøsta Nilsson, Amela Karahasanovic, Bjørnar Solhaug, Gyrd Brændeland, and Heidi Elisabeth Iuell Dahl. Thanks for the many interesting discussions and for all the useful feedback and advice you have provided.

My thanks also go to the people of the Department of Networked Systems and Services at SINTEF ICT and research director Bjørn Skjellaug for providing a good work environment. I would also like to thank the many members of the administrative and academic staff at the University of Oslo for helping me whenever I needed assistance.

I would like to thank all my friends for taking my mind off work, for all the joyful time we spend together, and for your support. I am also grateful for all the love and support I have received from my parents, Lars Ligaarden and Tove Randi Skjelkvåle, my brother Lars Anton Skjelkvåle Ligaarden, and my sister Ingeborg Skjelkvåle Ligaarden. I would also like to thank my sister for reading and commenting on my thesis.

The research on which this thesis reports has been carried out within the DIGIT project (180052/S10), funded by the Research Council of Norway, and the MASTER and NESSoS projects, both funded from the European Community's Seventh Frame-

work Programme (FP7/2007-2013) under grant agreements FP7-216917 and FP7-256980, respectively.

List of original publications

1. Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen. **ValidKI: A method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators.** *In International Journal on Advances in Intelligent Systems*, 5(1-2), pp. 175–193, IARIA, 2012.
*The first version of this paper received a **best paper award** at the First International Conference on Business Intelligence and Technology (BUSTECH'2011).*
2. Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen. **Using indicators to monitor risk in interconnected systems: How to capture and measure the impact of service dependencies on the quality of provided services.** *Chapter in the book “IT Security Governance Innovations: Theory and Research,”* D. Mellado, L. E. Sánchez, E. Fernández-Medina, and M. Piattini (eds.), pp. 256–292, IGI Global, 2012.
3. Olav Skjelkvåle Ligaarden, Mass Soldal Lund, Atle Refsdal, Fredrik Seehusen, and Ketil Stølen. **An architectural pattern for enterprise level monitoring tools.** *In Proceedings of 2011 IEEE International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA'2011)*, IEEE Computer Society, 2011.
4. Tormod Vaksvik Håvaldsrud, Olav Skjelkvåle Ligaarden, Per Myrseth, Atle Refsdal, Ketil Stølen, and Jon Ølnes. **Experiences from using a UML-based method for trust analysis in an industrial project on electronic procurement.** *In Journal of Electronic Commerce Research*, 10(3-4), pp. 441–467, Springer, 2010.
5. Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen. **Experiences from using indicators to validate expert judgments in security risk analysis.** *In Proceedings of Third International Workshop on Security Measurements and Metrics (MetriSec'2011)*, IEEE Computer Society, 2011.

The publications 1–5 are available as Chapters 9–13 in Part II of this thesis. In the case of publications 1, 2, and 5, we have included the full technical reports which are extended, slightly revised versions of the published papers.

Contents

Abstract	iii
Acknowledgements	v
List of original publications	vii
Contents	ix
List of figures	xiii
I Overview	1
1 Introduction	3
1.1 Objective	4
1.2 Contribution	4
1.3 Organization	6
2 Problem characterization	9
2.1 Conceptual clarification	9
2.1.1 System and service	9
2.1.2 Quality and quality of service	9
2.1.3 Service dependency	10
2.1.4 Indicator and metric	10
2.1.5 Trust	10
2.1.6 Architectural pattern	11
2.2 Success criteria	11
2.2.1 Framework for analyzing and monitoring the impact of dependencies on quality	12
2.2.2 Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators	12
2.2.3 Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services	13
2.2.4 Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators	13

3	Research method	15
3.1	The standing of computer science	15
3.2	Classical versus technology research	16
3.3	Strategies for evaluation	18
3.4	Our research method	19
4	State of the art	23
4.1	Indicators	24
4.1.1	Measurement of the fulfillment of business objectives	24
4.1.2	Specification of the design and deployment of indicators	24
4.1.3	Validation of indicators	25
4.2	Dependencies	26
4.2.1	Modeling and analysis of dependencies in general	26
4.2.2	Modeling and analysis of service dependencies	27
4.3	Risk analysis	28
4.3.1	Asset-based risk analysis methods	28
4.3.2	Analysis of risk in the context of dependencies	28
4.3.3	Validation of expert judgments in risk analysis	29
4.3.4	Indicator-based risk analysis	30
4.4	Monitoring	30
4.4.1	Monitoring of quality	30
4.4.2	Dynamic risk monitoring	31
4.4.3	Patterns and the implementation of monitoring tools	32
4.5	Trust	33
4.5.1	Estimating the trustworthiness of external services	33
4.5.2	Trust-based decisions in interconnected systems	34
5	Summary of contribution	37
5.1	The ValidKI Framework	38
5.2	Overview of artifacts	40
5.2.1	Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators	40
5.2.2	Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services	41
5.2.3	Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators	42
5.3	Overview of industrial case studies	43
5.3.1	Empirical study on trust-based decisions in interconnected systems	43
5.3.2	Empirical study on the design of indicators for monitoring risk	44
6	Overview of research papers	47
6.1	Paper A: ValidKI: A method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators	47
6.2	Paper B: Using indicators to monitor risk in interconnected systems: How to capture and measure the impact of service dependencies on the quality of provided services	48

6.3	Paper C: An architectural pattern for enterprise level monitoring tools .	49
6.4	Paper D: Experiences from using a UML-based method for trust analysis in an industrial project on electronic procurement	49
6.5	Paper E: Experiences from using indicators to validate expert judgments in security risk analysis	50
7	Discussion	51
7.1	Fulfillment of the success criteria	51
7.1.1	Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators	51
7.1.2	Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services	62
7.1.3	Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators	70
7.1.4	Framework for analyzing and monitoring the impact of dependencies on quality	76
7.2	How the artifacts relate to and extend the state of the art	77
7.2.1	Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators	77
7.2.2	Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services	78
7.2.3	Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators	78
8	Conclusion	81
8.1	What has been achieved	81
8.2	Directions for future work	82
	Bibliography	85
II	Research Papers	97
9	Paper A: ValidKI: A method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators	99
10	Paper B: Using indicators to monitor risk in interconnected systems: How to capture and measure the impact of service dependencies on the quality of provided services	149
11	Paper C: An architectural pattern for enterprise level monitoring tools	277
12	Paper D: Experiences from using a UML-based method for trust analysis in an industrial project on electronic procurement	289

13 Paper E: Experiences from using indicators to validate expert judgments in security risk analysis	319
---	------------

List of Figures

3.1	The main steps of the method for technology research (adopted from [1])	17
3.2	The main steps of the method for classical research (adopted from [1]) .	18
3.3	Our research method (adopted from [2] and modified)	20
3.4	How the artifacts and the framework are related to the empirical studies	22
5.1	Relations between the ValidKI Framework and Artifacts 1–3	38
5.2	The ValidKI Framework	39

Part I

Overview

Chapter 1

Introduction

Over time we have gradually become more and more dependent on services that are provided by interconnected systems. Such systems may, for instance, be found in power grids or in the Internet. They provide services like electricity, communication services, etc. The importance of the quality of these services in general and the security in particular is not new. In 1988, the Morris worm [3] infected about 4% of the approximate Internet population of 60000 computers. Much has however changed since the early days of the Internet. Since then, the Internet and other networks have increased much in size and become more interconnected, while incidents with respect to interconnected systems have increased both with respect to number and severity [4]. Moreover, the realization of the Internet of Things [5] will lead to even more interconnected networks. The Internet of Things refers to a world where physical objects and beings have virtual components that can produce and consume services. Such extreme interconnection will in particular result in new challenges for the security of services [6].

Despite of the importance of quality in general, security in particular, and the interconnected systems that surround us, there is often a lack of understanding of the interconnections' potential effect on quality of services. On January 25, 2003, the David-Besse nuclear power plant in Ohio was infected by a SQL slammer worm [7] due to a network connection that circumvented the firewall. The infection resulted in the internal network being overloaded, which again resulted in the unavailability of crucial control systems for about five hours. Although the operators were burdened by these losses, the plant was not affected because of analogue backup systems which remained unaffected. Later the same year, the so-called "Northeast blackout" [8] occurred. This blackout left 50 million people in North America without electrical power and affected other critical infrastructures such as transportation, communication, and water supply. A major contributing factor to the incident was a software bug, while the severe consequences were the result of the many interconnections.

Interconnected systems are often so-called system of systems (SoS). An SoS may be thought of as a kind of "super system" comprising a set of interconnected systems that work together towards some common goal. The common goal may be as simple as enabling all the individual systems to achieve their capabilities, or to construct a set of new capabilities not achievable by the individual systems alone.

Interconnected systems, such as SoS, are challenging from a quality perspective for the following reasons:

1. The services provided by one system may rely on services provided by other

systems, resulting in so-called service dependencies. Changes in the quality attributes of one service may easily cause the quality attributes of its dependent services to change as well. This means that in order to capture the impact of risk to quality of services provided by one system, we not only need to capture the risks arising in the system in question, but also risks which are solely or partially due to dependencies on other services.

2. The systems may be under different managerial control and within different jurisdictions. For the systems that are outside our control, we have limited knowledge of their risks, structure, and behavior. Thus, we need means for capturing the impact of service dependencies involving systems for which we have insufficient information on risk.
3. Such a large number of systems, controlled and operated by different parties, evolve rapidly in a manner that may be difficult to predict. Thus, there is a need for updating the risk picture as the interconnected systems change with values based on observable properties of the interconnected systems. It is however not trivial to achieve this. The data from which the updates are calculated may be associated with many different sources of uncertainty. Thus, the validity of the data must be taken into account in order to ensure the correctness of the risk picture.

Traditional approaches to risk analysis such as [9–11] lack capabilities for addressing these challenges in a satisfactory manner. Moreover, a critical infrastructure may often be thought of as a set of interconnected systems that interact by the use of services. Hence, research on critical infrastructure protection is relevant in this thesis. In [12–14], state of the art on critical infrastructure protection are presented. With respect to the above mentioned challenges, the approaches are either not relevant at all or they lack capabilities for addressing them in a satisfactory manner. Based on all of this, we see the need for new artifacts for addressing the above mentioned challenges.

1.1 Objective

The purpose of this thesis has been to develop a framework for analyzing and monitoring the impact of dependencies on quality. More specifically, the framework should be useful in the context of interconnected systems to analyze and monitor the impact of service dependencies on quality of services. The overall objective has been to: *develop a framework that is:*

1. *well-suited to analyze the impact of service dependencies on quality of services;*
2. *well-suited to support the set-up of monitoring of the impact of service dependencies on quality of services; and*
3. *applicable in an industrial context within acceptable effort.*

1.2 Contribution

The framework is the result of the development and integration of three artifacts. Since the artifacts may be employed and used in practice independently of each other, they

may be seen as contributions on their own. Besides contributing in terms of new artifacts, the thesis also contributes in terms of empirical results from two industrial case studies. The industrial case studies were mainly carried out to support the development of the artifacts, but since they also provide insight into issues of a more general nature, they may be seen as contributions on their own. The main contributions of this thesis are: (1) a method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators; (2) a method for capturing and monitoring the impact of service dependencies on the quality of provided services; (3) an architectural pattern for constructing enterprise level monitoring tools based on indicators; (4) an empirical study on trust-based decisions in interconnected systems; and (5) an empirical study on the design of indicators for monitoring risk.

1. *Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators:* The method takes business objectives focusing on quality as input, and delivers valid indicators as output. By valid indicators we mean that the indicators measure to what extent the business or relevant part thereof fulfills the business objectives. The method also results in deployment and design specifications for the different indicators. The deployment specifications document how sensors for gathering the data needed in the calculation of the indicators should be deployed in the relevant part of business, while the design specifications document how the indicators should be calculated based on data provided by the sensors. These specifications may be used to implement ICT-supported monitoring of the indicators.

Analysts will manage the application of the method and document its results, while domain experts will participate during the application of the method. The domain experts are supposed to communicate their knowledge in such a way that correct models are achieved.

2. *Method for capturing and monitoring the impact of service dependencies on the quality of provided services:* The method is used for capturing the impact of service dependencies on risk to the quality of provided services in interconnected systems, and for setting up monitoring of selected risks by the use of indicators for the purpose of providing a dynamic risk picture for the provided services. The result of applying the method is a risk picture that captures the impact of services dependencies on the quality of the provided services. The risk picture is parameterized by indicators, each defined by design and deployment specifications. These specifications may be used in the implementation of a risk monitor.

Analysts will manage the application of the method and document its results, while domain experts will participate during the application of the method. The domain experts are supposed to communicate their knowledge in such a way that correct models are achieved.

3. *Architectural pattern for constructing enterprise level monitoring tools based on indicators:* The pattern serves as a basis for constructing enterprise level monitoring tools based on indicators. These are tools that: collect low-level indicators from the ICT infrastructure or similar; aggregate the low-level indicators into high-level indicators, useful at the enterprise level; and present the high-level

indicators in a way that is understandable to the intended users. The pattern structures an enterprise level monitoring tool into a set of components, and it captures features that are general to a broad class of enterprise level monitoring tools.

The architectural pattern will be employed by developers of ICT-based enterprise level monitoring tools.

4. *Empirical study on trust-based decisions in interconnected systems*: The empirical study was conducted as part of an industrial project focusing on the use of a UML-based trust analysis method to model and analyze a public eProcurement system (used by public authorities to award contracts to economic operators). This system makes use of a Validation Authority (VA) service for validating electronic IDs and digital signatures. The goal of the trust analysis was to obtain a better understanding of the potential usefulness of a VA service for supporting trust-based decisions in systems which rely on electronically signed documents. The study gave strong indications that the trust analysis method is feasible in practice.
5. *Empirical study on the design of indicators for monitoring risk*: The empirical study was integrated in a commercial security risk analysis conducted in 2010. In this analysis, indicators were designed for the purpose of validating likelihood estimates obtained from expert judgments. The main result from the empirical study was the identification of several challenges related to the design of indicators for monitoring security risks.

1.3 Organization

The thesis is structured into two main parts. Part I provides the context and an overall view of the work, while Part II contains the research papers. Each of the papers is self-contained and can therefore be read separately. We have structured Part I into eight chapters:

Chapter 1 – Introduction provides the background and motivation for the thesis, brief explanations of the objective and contributions, and the structure of the thesis.

Chapter 2 – Problem characterization clarifies the interpretation of core concepts used throughout the thesis and refines the overall objective into success criteria that the framework and the three artifacts must fulfill.

Chapter 3 – Research method presents the research method used in the thesis work.

Chapter 4 – State of the art provides an overview of work related to the research presented in the thesis.

Chapter 5 – Summary of contribution presents the framework and provides an overview of the five contributions.

Chapter 6 – Overview of research papers provides an overview of the papers resulting from the research.

Chapter 7 – Discussion discusses to what extent the success criteria has been fulfilled and how our artifacts relate to and extend the state of the art.

Chapter 8 – Conclusion summarizes the work and discusses different directions for future work.

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 core concepts used throughout the thesis. In Section 2.2 we present success criteria that should be fulfilled in order to successfully accomplish the research objective. Section 2.2 is divided into four sub-sections. In Section 2.2.1 we present success criteria for the framework for analyzing and monitoring the impact of dependencies on quality, while in Sections 2.2.2–2.2.4 we present success criteria for the three artifacts.

2.1 Conceptual clarification

This section characterizes the main terminology used throughout the thesis.

2.1.1 System and service

As already explained in Chapter 1, the framework should be useful in the context of interconnected systems. Both computerized and non-computerized systems are addressed in this thesis. In our context, a system is characterized by a set of components which interact and operate as a whole. Based on this, we define system as “*a group of interacting, interrelated, or interdependent elements forming a complex whole*” [15].

In our context, systems are interconnected if they interact by the use of services. The different systems act as providers and/or consumers of services, where each service represents the exchange of some commodity (electricity, information, etc.). Moreover, we limit each service to have one provider and one consumer. Based on the above, we end up with the following definition for service: “*A service is provided by a system and consumed by a system, and it represents the exchange of some commodity.*”

2.1.2 Quality and quality of service

The framework has a strong focus on quality and quality of service. Generally, quality is concerned with the degree to which relevant non-functional requirements are fulfilled. In [16], quality is defined as “*the degree to which a system, component, or process meets specified requirements,*” while [17] defines quality as “*the ability of a product, service, system, component, or process to meet customer or user needs, expectations, or requirements.*” Based on the two definitions above, we define quality as “*the degree to which*

a system, service, component, or process meets specified non-functional requirements.” Moreover, based on this definition we define quality of service as *“the degree to which a service meets specified non-functional requirements.”*

2.1.3 Service dependency

As already explained in Chapter 1, the framework has been developed to analyze and monitor the impact of service dependencies on quality of services. A service may require other services in order to be provided with the required quality. A service dependency describes a relationship between a service provided by a system and services this system requires from its environment to provide the service in question. In this thesis, we consider a service to be dependent on other services if *“a change in the quality of the latter may lead to a change in the quality of the former.”*

2.1.4 Indicator and metric

In this thesis, the impact of service dependencies on quality of services is monitored by the use of indicators. Hammond et al. defines indicator as *“something that provides a clue to a matter of larger significance or makes perceptible a trend or phenomenon that is not immediately detectable”* [18]. For example, a drop in barometric pressure may signal a coming storm, while an unexpected rise in the traffic load of a web server may signal a denial of service attack in progress. Thus, the significance of an indicator extends beyond what is actually measured to a larger phenomenon of interest.

Indicators are closely related to metrics. In [17], metric is defined as *“a quantitative measure of the degree to which a system, component, or process possesses a given attribute,”* while it defines attribute as *“the specific characteristic of the entity being measured.”* For the web server mentioned above, an example of an attribute may be availability. An availability metric may again act as an indicator for denial of service attacks, if we compare the metric with a baseline or expected result [19]. As we can see, metrics are not that different from indicators. For that reason, indicators and metrics are often used interchangeably in the literature.

It should also be noticed that indicators are often referred to as key indicators in a business context. Here, the key indicators are used to measure to what extent business objectives/goals are fulfilled. In Paper A (presented in Chapter 9), we refer to indicator as key indicator, while in the rest of the thesis we only use the term indicator.

2.1.5 Trust

In the context of the thesis, trust is relevant for reasoning about third-party service dependencies. Inspired by [20, 21], [22] defines trust as *“the subjective probability by which an actor (the trustor) expects that another entity (the trustee) performs a given transition on which its welfare depends.”* In other words, trust is the belief of a trustor that a trustee will perform a specific transaction on which the welfare of the trustor depends. For instance, the operator of a system may have a certain amount of trust in the ability of an operator of another system to deliver a service according to requirements. The level of trust may vary from 0 (complete distrust) to 1 (complete trust).

2.1.6 Architectural pattern

In this thesis, we present an architectural pattern for constructing enterprise level monitoring tools based on indicators. A pattern captures the essence of solving a recurring problem. It is a description or template for how to solve a problem that can be used in many different situations. In software engineering, patterns are best known as design patterns [23]. These patterns are used for solving recurring design problems. In [17], design pattern is defined as “*a description of the problem and the essence of its solution to enable the solution to be reused in different settings.*” In this thesis we focus on architectural patterns [24]. The difference between a design pattern and an architectural pattern is that design patterns describe design solutions at the object/class level, while architectural patterns describe design solutions at the architectural level, e.g., design of components and their relationships.

2.2 Success criteria

In Chapter 1 we outlined the problem area that motivates the work of this thesis, and we argued that there is a need for a framework for the analysis and monitoring of the impact of service dependencies on quality of services in the context of interconnected systems. As explained in Chapter 1, the overall objective has been to: *develop a framework that is:*

1. *well-suited to analyze the impact of service dependencies on quality of services;*
2. *well-suited to support the set-up of monitoring of the impact of service dependencies on quality of services; and*
3. *applicable in an industrial context within acceptable effort.*

The framework is the result of the development and integration of three artifacts:

1. Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators
2. Method for capturing and monitoring the impact of service dependencies on the quality of provided services
3. Architectural pattern for constructing enterprise level monitoring tools based on indicators

As already explained in Chapter 1, each of the three artifacts may be viewed as contributions on their own. In Chapter 5, we relate the three artifacts to each other, and we summarize their contributions to the overall objective. The main hypothesis for each of the artifacts is that it fulfills its intended purpose (as elaborated below), and that it is feasible to be used by its intended users. As explained in Chapter 1, the artifacts target the following different types of user groups:

- The target group of Artifact 1 is the analyst and the domain experts.
- The target group of Artifact 2 is the analyst and the domain experts.

- The target group of Artifact 3 is the developer of ICT-based enterprise level monitoring tools.

For the framework and each of the artifacts, we have identified a set of success criteria that the framework/artifact should fulfill. These are presented in Sections 2.2.1–2.2.4.

2.2.1 Framework for analyzing and monitoring the impact of dependencies on quality

The framework is the result of integrating the three artifacts. The purpose of the framework is to: (1) analyze the impact of service dependencies on quality of services; and (2) support the set-up of monitoring of the impact of service dependencies on quality of services. Hence, the following overall success criterion:

Success criterion 1 *The framework fulfills its intended purpose.*

2.2.2 Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators

The purpose of Artifact 1 is to facilitate the design and assessment of indicators to be used in ICT-supported monitoring of the fulfillment of business objectives focusing on quality.

Success criterion 2 *The application of the method results in indicators that measure correctly to what extent the business objectives received as input are fulfilled.*

Indicators are not suitable for measuring the fulfillment of business objectives if they cannot measure correctly to what extent the business objectives are fulfilled. The use of such indicators may lead to bad business decisions, which again can harm the company. To ensure that suitable indicators are designed, we need to evaluate whether the designed indicators measure correctly to what extent the business objectives are fulfilled.

Success criterion 3 *The application of the method results in specifications of indicators that are well-suited for ICT-based monitoring.*

The main output from the application of the method is specifications of the indicators. These specifications are to be used to implement ICT-based monitoring of the indicators. Thus, the specifications need to be well-suited for implementing ICT-based monitoring.

Success criterion 4 *The method is applicable in an industrial context within acceptable effort.*

In order for the method to be useful, it must be possible to apply it in an industrial context within acceptable effort.

2.2.3 Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services

The purpose of Artifact 2 is to facilitate the capture of the impact of service dependencies on risk to the quality of provided services in interconnected systems, and to facilitate the set-up of monitoring of selected risks by the use of indicators for the purpose of providing a dynamic risk picture for the provided services.

Success criterion 5 *The application of the method results in specifications of indicators that correctly capture and measure the impact of service dependencies on the quality of provided services.*

The main output from the application of the method is specifications of indicators for monitoring risk to quality of provided services. These specifications need to correctly capture and measure the impact of service dependencies on the quality of provided services in order to be useful.

Success criterion 6 *The application of the method results in specifications for the deployment and design of indicators that are sufficient for setting up risk monitoring based on indicators.*

In order to provide a dynamic risk picture for the provided services, the method must result in specifications for the deployment and design of indicators that are sufficient for setting up risk monitoring based on indicators.

Success criterion 7 *The method is applicable in an industrial context within acceptable effort.*

In order for the method to be useful, it must be possible to apply it in an industrial context within acceptable effort.

2.2.4 Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators

The purpose of Artifact 3 is to serve as a basis for implementing enterprise level monitoring tools based on indicators.

Success criterion 8 *The architectural pattern serves as a basis for building monitoring tools based on indicators within a wide range of domains and enterprises.*

It should be possible to use the architectural pattern as a basis for building monitoring tools based on indicators within a wide range of domains and enterprises. In order for this to be possible, the pattern must capture features that are general to a

broad class of enterprise level monitoring tools.

Success criterion 9 *The architectural pattern facilitates modularity and reuse.*

We have identified two desirable properties for monitoring tools resulting from the application the architectural pattern. The first desirable property is that the application of the pattern should result in tools that are modular, while the second desirable property is that it should be possible to build tools that can be reused when building other monitoring tools.

Chapter 3

Research method

In this chapter we present our research method. We start by three introductory sections. In Section 3.1 we discuss the standing of computer science as a science and describe its overall research method. In Section 3.2 we relate classical and technology research as defined by Solheim and Stølen. In Section 3.3 we describe different strategies to evaluate the research results. Thereafter, in Section 3.4 we go on to describe the research method used in this thesis. We emphasize in particular the role of empirical studies in the invention and evaluation of the three artifacts.

3.1 The standing of computer science

Computer science is a relatively young discipline compared to classical sciences such as mathematics, physics, astronomy, etc., which date back to the ancient Greeks. Moreover, computer science has a unique standing with respect to theory and practice compared to the classical sciences. While the focal point of classical sciences is more on the *what* than *how*, many advances in the history of computer science have been driven by explaining *how* something can be achieved through the interaction between theory and the technology that realizes it [25, 26]. For this reason, [25] states that the science and engineering aspects of computer science are much closer than in many other disciplines.

With computer science being such a young discipline and closely connected to engineering, its qualification as a science has been widely debated [25, 27–30]. Abelson and Sussman [30] claim that computer science is not a science and that its significance has little to do with computers. The authors do not explain what qualifies as a science or why computer science does not qualify as one. They do however relate computation to mathematics by describing mathematics as the framework for dealing precisely with notions of *what is*, while they describe computation as the framework for dealing precisely with notions of *how to*.

Similarly, Brooks [29] claims that computer science is not a science but merely an engineering discipline. According to Brooks, computer scientists are engineers since they study in order to build things such as computers, algorithms, software systems, etc. The scientist, on the other hand, is concerned with the discovery of facts and laws. The scientist does only build things that are needed for supporting his/hers studies.

Even though we find some objections to the classification of computer science as a science, there is a widely established agreement that computer science is in fact a

science, because of its many similarities with the classical sciences. Denning [27] describes computer science as the blend of mathematics, science, and engineering. Moreover, Denning criticizes the ones who object to computer science being a science on the grounds that man-made objects (technologies) are studied. According to Denning, computer science is about studying natural and artificial information processes. To study these processes, computer science relies on the same method as the classical sciences; namely the scientific method, or what we often refer to as the hypothetico-deductive method: (1) recognition of a theoretical problem; (2) proposal of conjectured solution to the problem (hypothesis); (3) critical testing of the hypothesis; and (4) conclusion: retain hypothesis if test is positive, otherwise reject hypothesis as falsified and possibly devise new tests/hypotheses/problems [31]. According to Denning, there are many examples of the usage of the scientific method within computer science. One example is that software engineering researchers hypothesize models for how programming is done and how effects arise. These models go through testing where the researchers seek to understand which models work well and how to use them to create better programs with fewer defects.

Tichy [32] shares Denning's view on computer science being the study of information processes. According to Tichy, the applicability of the scientific method is as relevant in computer science as it is in for instance physics. It does not make any difference that the subject of inquiry is information instead of energy or matter. In both cases we would need to observe a phenomenon, formulate explanations and theories, and test them.

3.2 Classical versus technology research

Solheim and Stølen [1] distinguish between classical and technology research. For both classical and technology research we start with an overall hypothesis on the form “ B solves the problem A .” In the case of classical research, A is the need for forming a new or improved theory about a real-world phenomenon, while B is the new theory. The real-world phenomenon addressed may take place in nature, space, the human body, society, etc. In its simplicity, classical research is concerned with seeking new knowledge about the real world. On the other hand, in technology research A is the need for a new or improved human-made object, i.e., artifact, while B is this artifact. The technology researcher strives to manufacture artifacts which are better than those that already exist. The result of conducting technology research may for instance be a new or improved material, medicine, algorithm, software engineering method, etc.

Solheim and Stølen [1] argue that despite the differences in the problems addressed and the solutions sought after, classical and technology research have a lot in common and follow the same principal steps for finding a solution to the research problem. Moreover, Solheim and Stølen claim that technology research should be conducted in accordance with the hypothetico-deductive method of classical research. In technology research, the development of an artifact is motivated by a need. The overall hypothesis of technology research is that the artifact satisfies the need. A common way to evaluate the satisfaction of the overall hypothesis is to formulate a set of predictions. The falsification of a prediction results in the rejection the overall hypothesis. Hence, the predictions serve as a basis for gathering evidence on the validity of the overall hypothesis.

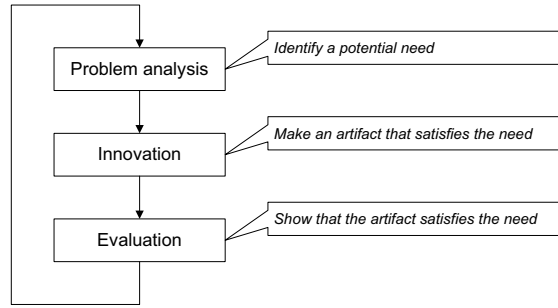


Figure 3.1: The main steps of the method for technology research (adopted from [1])

Technology research, like classical research, is driven by an iterative process [1]. In Figures 3.1 and 3.2 (both adopted from [1]) we have summarized the processes of technology research and classical research. The main steps of the technology research method are as follows:

1. *Problem analysis* – The researcher identifies a potential need for a new or improved artifact by interacting with potential users and other stakeholders. During this step, the researcher expresses the satisfaction of the potential need through a set of success criteria.
2. *Innovation* – The researcher tries to manufacture an artifact that satisfies the success criteria. The overall hypothesis is that the artifact satisfies the success criteria, or more precisely the potential need.
3. *Evaluation* – Based on the success criteria, the researcher formulate a set of predictions about the artifact and evaluate whether these predictions come true. If the evaluation results in a positive outcome, then the researcher may argue that the artifact satisfies the potential need.

Moreover, the main steps of the classical research method are as follows:

1. *Problem analysis* – The researcher identifies a need for a new or better theory. The need is either due to the lack of a theory or a deviation between present theory and reality.
2. *Innovation* – The researcher suggests a new explanation. The researcher works according to the overall hypothesis that the new explanation agrees with reality.
3. *Evaluation* – The researcher checks whether the hypothesis is true by performing observations. Based on the hypothesis, the researcher formulates predictions and checks whether the predictions come true. If the observations verify the predictions, the researcher can argue that the new explanation agrees with reality.

Technology research is similar to technology development. In both technology research and technology development, artifacts are developed to satisfy potential needs. What distinguishes technology research from technology development is that: (1) the

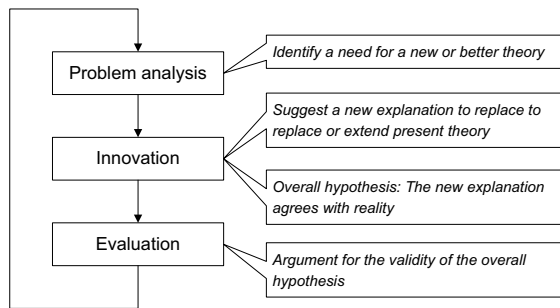


Figure 3.2: The main steps of the method for classical research (adopted from [1])

development of the artifact results in new knowledge; (2) the new knowledge is of interest to others; and (3) the development of the artifact is documented in such a way that it enables others to repeat and verify the development of the artifact [1].

Technology research is also closely related to design science. This paradigm and the behavioral-science paradigm characterize much of the research in the information systems discipline [33]. Moreover, the paradigm has its roots in engineering and the sciences of the artificial [34]. In information systems research, design science is used to create and evaluate IT artifacts intended to solve problems related to some aspect of the design of an information system. Design science differs from routine design or system development by contributing with new knowledge in the form of foundations and methodologies for design.

3.3 Strategies for evaluation

As mentioned in Section 3.2, evaluation is to find out if the predictions are true. Regardless of whether classical or technology research is being conducted, different strategies may be used to gather the evidence necessary to test the predictions. According to McGrath [35], when you gather evidence, you are always trying to maximize three things: (*A*) generality – results that are valid across populations; (*B*) precision – precise measurements; and (*C*) realism – the evaluation is performed in environments similar to reality. Different strategies all have their strengths and weaknesses with respect to *A*, *B*, and *C*. According to McGrath, the eight most common strategies are:

- *Field studies* refer to efforts to make direct observations of ongoing systems, while interfering with the systems as little as possible.
- *Laboratory experiments* are attempts from a researcher to observe systems in a context where the researcher may control and isolate the variables whose effects are to be examined.
- *Field experiments* are field studies with one major difference; the deliberate manipulation of the variables whose effects are to be examined.
- *Experimental simulations* are conducted in a laboratory setting. The researcher makes an effort to create a system that is like some class of natural occurring systems. The system is artificial in the sense that it is only created for the study.

- *Sample survey*, or just *survey*, are efforts to obtain information from a broad and carefully selected group of actors. The information is often given in the form of verbal responses to a set of questions.
- *Judgment studies*, or what may also be referred to as *qualitative interviews*, are efforts to obtain information from a small set of actors. The information obtained tends to be more precise than information obtained from sample surveys, but cannot be generalized in the same way.
- *Formal theory* is a theoretical approach where evidence is gathered by the use of argumentation based on logical reasoning.
- *Computer simulations* is another theoretical approach where attempts are made to model a specific real life system or class of systems.

None of the strategies is able to maximize A , B , and C simultaneously. Laboratory experiments score high on precision, while field studies have the greatest realism. Moreover, formal theory and sample surveys deliver the greatest generality. The solution must therefore be to choose several strategies that complement each other. When choosing strategies to evaluate the predictions, the researcher needs to consider a number of aspects [1]:

- *Is the strategy feasible?* Time, cost, and the availability of resources, such as individuals to participate in the evaluation, are important aspects to consider when selecting a strategy.
- *How to ensure that a measurement really measures the property it is supposed to measure?* The property to be measured needs to be isolated, and different factors that may influence the measurement need to be accounted for. The researcher also needs to take into account the nature of the property, i.e., whether it is qualitative, quantitative, or formal, when selecting the strategy to be used.
- *What is needed to falsify the prediction?* The strategy selected is nothing worth if it cannot cause the prediction to be rejected.

3.4 Our research method

The overall objective of the thesis has been to develop a framework for analyzing the impact of service dependencies on quality of services, and to support the set-up of monitoring of the impact of service dependencies on quality of services. As previously mentioned, the framework integrates three artifacts. Each of the three artifacts also serves a purpose outside the context of the framework. Thus, the artifacts have been developed in such a way that they can be employed and used in practice independently of each other. To develop the artifacts, the technology research method described in Section 3.2 was applied. The result of applying this highly iterative method was that the framework, artifacts, and success criteria were constantly improved as the evaluation provided us with new insight into the research problems.

Figure 3.3 describes the process that was followed in our thesis work. The figure also shows how this process relates to the process depicted in Figure 3.1. For each artifact, we first identified success criteria. Both the usage of the artifact in the framework and

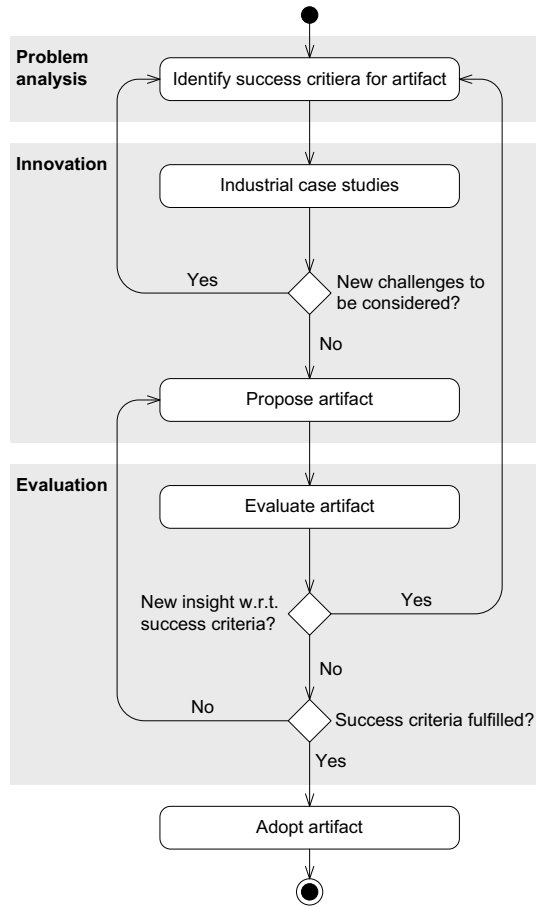


Figure 3.3: Our research method (adopted from [2] and modified)

its independent usage were taken into account when identifying the success criteria. The next step was to conduct industrial case studies. Besides providing valuable input to the development of the framework and the artifacts, the two case studies may also be seen as contributions on their own, since they provide insight into issues of a more general nature. The following industrial case studies were conducted:

- *Study 1*: Empirical study on trust-based decisions in interconnected systems
- *Study 2*: Empirical study on the design of indicators for monitoring risk

Study 1 and *Study 2* were conducted for the purpose of identifying challenges with respect to the use of trust-based decisions in interconnected systems and design of indicators for monitoring risk, respectively. In the innovation phase, each artifact was developed with respect to the success criteria. And finally, the artifact was evaluated with respect to the success criteria.

The following three artifacts have been developed by following the process in Figure 3.3:

- *Artifact 1*: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators
- *Artifact 2*: Method for capturing and monitoring the impact of service dependencies on the quality of provided services
- *Artifact 3*: Architectural pattern for constructing enterprise level monitoring tools based on indicators

The three artifacts have been checked against the literature. To evaluate *Artifact 1* and *Artifact 2*, the following two case studies were conducted:

- *Study 3*: Case study for evaluating the method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators
- *Study 4*: Case study for evaluating the method for capturing and monitoring the impact of service dependencies on the quality of provided services

Both *Study 3* and *Study 4* are the results of development of large, realistic examples for the purpose of evaluating the success criteria associated with *Artifact 1* and *Artifact 2*, respectively. Both case studies cover all the steps of the two methods from start to finish. Thus, the entire method was considered in both cases.

Artifact 3 (the architectural pattern) is a generalization of the architecture of the CORAS risk monitor [36] that was developed in the MASTER¹ [37] research project. The artifact has been evaluated based on experiences from the MASTER project, and by arguing for its ability to serve as a basis for building monitoring tools based on indicators within a wide range of domains and enterprises.

Figure 3.4 shows how the artifacts and the framework are related to the four case studies. The two industrial case studies, *Study 1* and *Study 2*, identified challenges that have been taken into consideration when identifying the challenges to be addressed by the artifacts and the framework. In particular, the main result from *Study 1* is the need for reasoning about trust in interconnected systems. This challenge has been addressed by *Artifact 2*. The main results from *Study 2* is the need for valid indicators when monitoring risk and domain knowledge for systems with dependencies. The former challenge has been addressed by *Artifact 1*, while the latter result has mainly been used in the development of *Artifact 2*. As already explained, *Study 3* and *Study 4* have been used to evaluate *Artifact 1* and *Artifact 2*, respectively. On the other hand, the development of the two case studies has also resulted in new challenges being identified. These challenges have been addressed during the development of the two artifacts.

¹According to [37], the MASTER (Managing Assurance Security and Trust for sERvices) project aimed at “providing methodologies and infrastructures that facilitate the monitoring, enforcement, and audit of quantifiable indicators on the security of a business process, and that provide manageable assurance of the security levels, trust levels and regulatory compliance of highly dynamic service-oriented architecture in centralized, distributed (multidomain), and outsourcing contexts.”

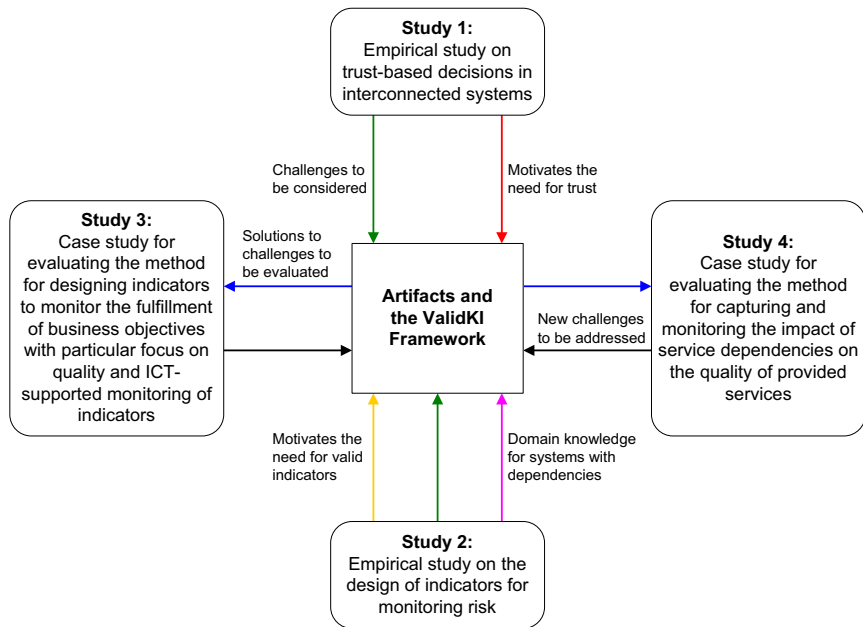


Figure 3.4: How the artifacts and the framework are related to the empirical studies

Chapter 4

State of the art

In this chapter we provide an overview of the state of the art of relevance to the contributions of this thesis. In the following we motivate the structure of this chapter.

The main result of Artifact 1 is a method for specifying the design and deployment of indicators to be used in ICT-supported monitoring of the fulfillment of business objectives focusing on quality. In Section 4.1.1 we present state of the art of relevance to measuring the fulfillment of business objectives. Section 4.1.2 presents state of the art related to the specification of design and deployment of indicators, while state of the art related to the validation of indicators is presented in Section 4.1.3. Moreover, state of the art of relevance to the monitoring of quality is presented in Section 4.4.1. In all four sections we put particular emphasis on the issue of quality.

The main result of Artifact 2 is a method for capturing the impact of service dependencies on risk to quality of provided services in interconnected systems. This includes facilitating monitoring of selected risks by the use of indicators to offer a dynamic risk picture for the provided services. The method relies on modeling and analysis of service dependencies in order to capture their impact on risk. In Sections 4.2.1 and 4.2.2 we provide an overview of state of the art related to the modeling and analysis of dependencies in general and service dependencies in particular, respectively. The method also provides a solution for estimating the trustworthiness of services provided by third-parties. Related state of the art is presented in Section 4.5.1. To capture the impact of service dependencies on risk to quality, the method employs an asset-based risk analysis approach. In Section 4.3.1 we present approaches of this kind, while in Section 4.3.2 we direct our attention to state of the art of relevance to the analysis of risk in the context of dependencies in general and service dependencies in particular. As already explained, the method is also used for setting up monitoring of selected risks to quality. Thus, in Section 4.4.2 we discuss state of the art of relevance to dynamic risk monitoring.

The main result of Artifact 3 is an architectural pattern that serves as a basis for implementing enterprise level monitoring tools based on indicators. In Section 4.4.3 we present state of the art related to patterns and the implementation of monitoring tools.

The first industrial case study investigates the use of trust to reason about the behavior of systems/actors in cases where the behavior of these systems/actors may result in risks and/or opportunities. In Section 4.5.2 we discuss state of the art related to trust-based decisions in interconnected systems.

The second industrial case study investigates the use of indicators to validate likeli-

hood estimates based on expert judgments in security risk analysis. In Section 4.3.3 we present state of the art related to the validation of expert judgments used in risk analysis, while in Section 4.3.4 we focus on state of the art related to the use of indicators in risk analysis.

4.1 Indicators

4.1.1 Measurement of the fulfillment of business objectives

There exist numerous approaches for measuring business performance. Some of these are presented in [38]. Regardless of the approach being used, the organization must translate their business objectives/goals into a set of key performance indicators in order to measure performance. An approach that is widely used [39] is balanced scorecard [40]. This approach translates the company's vision into four financial and non-financial perspectives. For each perspective, a set of business objectives (strategic goals) and their corresponding key performance indicators are identified. However, the implementation of a balanced scorecard is not necessarily straight forward. In [41], Neely and Bourne identify several reasons for the failure of measurement initiatives such as balanced scorecards. One problem is that the identified measures do not measure fulfillment of the business objectives, while another problem is that measures are identified without putting much thought into how the data must be extracted in order to compute the measures.

There exist a number of frameworks and best practice approaches that are supported by metrics/indicators for measuring the achievement of goals. One example is COBIT (Control Objectives for Information and related Technology) [42], which is a framework for IT management and IT governance. The framework provides an IT governance model that helps in delivering value from IT and understanding and managing the risks associated with IT. In the governance model, business goals are aligned with IT goals, while metrics, in the form of leading and lagging indicators [43], and maturity models are used to measure the achievement of the IT goals. Another example is Val IT [44], which is a framework that is closely aligned with and that complements COBIT. It consists of a set of guiding principles and a number of processes and best practices for measuring, monitoring, and optimizing the realization of business value from investment in IT.

ITIL [45–49] (Information Technology Infrastructure Library) provides best practice guidance for IT service management for the purpose of aligning IT services with business needs. The current version of ITIL (version 3) provides a holistic perspective of the full life cycle of services by covering the entire IT organization and all supporting components needed to deliver services to customers. Under ITIL guidelines, services are designed to be measurable. To support measurement, ITIL comes with a number of metrics/indicators.

4.1.2 Specification of the design and deployment of indicators

A number of standards and guides provide guidance on the design of indicators/metrics. The ISO/IEC 27004 [50] standard provides guidance on the development and use of measures and measurement for assessing information security management systems and controls, as specified in ISO/IEC 27001 [51]. The appendix of the standard also

suggests security metrics which have been selected to align with ISO/IEC 27002 [52]. Moreover, the NIST Performance Measurement Guide for Information Security [53] provides guidance on the development, selection, and implementation of suitable measures for information security. It also comes with a number of candidate measures for measuring information security.

The Goal-Question-Metric [54,55] (GQM) is an approach for measuring the achievement of goals. Even though GQM originated as an approach for measuring achievement in software development, it can also be used in other contexts where the purpose is to measure achievement of goals. In GQM, business goals are used to drive the identification of measurement goals. These goals do not necessarily measure the fulfillment of the business goals, but they should always measure something that is of interest to the business. Each measurement goal is refined into questions, while metrics are defined for answering the questions. The data provided by the metrics are interpreted and analyzed with respect to the measurement goal in order to conclude whether it is achieved or not.

An approach that is closely related to GQM is the Goal-Question-Indicator-Measurement [56,57] (GQ(I)M) approach. In this approach, we start by identifying business goals that we break down to manageable sub-goals. The approach ends with a plan for implementing measures and indicators that support the goals.

In [58], Popova and Sharpanskykh present a framework for modeling performance indicators within a general organization modeling framework. Two of the main contributions of this framework are the formalization of the concept of a performance indicator, and the formalization of the relationships between performance indicators. In [59], the same authors present a formal framework for modeling goals based on performance indicators. This formal framework is also used within the general organization modeling framework mentioned above. To enable evaluation of organizational performance, the framework defines mechanisms for establishing goal satisfaction. By using the frameworks presented in [58,59], goal and performance indicator structures can be modeled. The goal structure is given in the form of a hierarchy. It shows how different goals are related and how goals are refined into sub-goals. It also defines how goal satisfaction should be propagated through the hierarchy. On the other hand, the performance indicator structure shows the different relationships that exist between the performance indicators. It may for instance show that one indicator is the aggregation of other indicators, or that the value of one indicator influences the value of another. In order to identify goals and performance indicators and to create the two structures, both frameworks rely on organizational documents and expert knowledge. The frameworks also provide mechanisms for checking consistency of and correspondence between the goal and the performance indicator structures.

4.1.3 Validation of indicators

No specific method, beyond reviews, is specified for validating whether the correct questions/sub-goals and metrics/indicators have been identified by the use of GQM or GQ(I)M. In the case of GQM, the different kinds of data used for computing the metrics should be checked for correctness, completeness, and consistency [55].

In the software engineering literature, the validity of metrics that measure attributes of software products is an important topic (see e.g., [60–62]). Even though this literature targets software engineering, it is still relevant to different extents in other domains

that also focus on the validity of measurements/metrics/indicators. Even though the validity of software engineering metrics have received a lot of attention, no agreement have been reached upon what constitutes a valid metric [63]. In [63], Meneely et al. present a systematic literature review of papers focusing on validation of software engineering metrics. The literature review began with 2288 papers, which were later reduced to 20 papers. From these 20 papers, the authors extracted and categorized 47 unique validation criteria. The authors argue that metric researchers and developers should select criteria based on the intended usage of the metric when validating it.

A number of the software metrics validation approaches advocate the use of measurement theory [64–66] in the validation (see e.g., [67–69]). Measurement theory is a branch of applied mathematics that is useful in measurement and data analysis. The fundamental idea of this theory is that there is a difference between measurements and the attribute being measured. Thus, in order to draw conclusions about the attribute, there is a need to understand the nature of the correspondence between the attribute and the measurements. In [70], Morali and Wieringa present an approach that relies on measurement theory for the validation of indicators. More specifically, the approach uses measurement theory to validate the meaningfulness of IT security risk indicators.

Measurement theory has been criticized of being too rigid and restrictive in a practical measurement setting. Briand et al. [68] advocate a pragmatic approach to measurement theory in software engineering. The authors show that even if their approach may lead to violations of the strict prescriptions and proscriptions of measurement theory, the consequences are small compared to the benefits. Another approach that takes a pragmatic approach to measurement theory is [69]. Here, the authors propose a framework for evaluating software metrics. The applicability of the framework is demonstrated by applying it on a bug count metric.

4.2 Dependencies

4.2.1 Modeling and analysis of dependencies in general

Dependencies are often modeled by the use of directed graphs. In these graphs, vertices represent different entities, while edges represent dependencies between the entities. The direction of an edge specifies the direction of the dependency. There exist a number of graph-based approaches that can be used in the modeling of dependencies. One example is semantic networks [71], which is a technique for knowledge representation. The main idea behind semantic networks is that knowledge is often best understood as a set of concepts that are related to each other. A semantic network is given in the form of a graph, where vertices represent concepts, while directed labeled edges connect the concepts. The meaning of a concept is defined by its semantic relations to other concepts. With a semantic network we can show which concepts that depend on each other and why.

In [72], Cox et al. present an approach to dependency analysis that relies on conceptual graphs [73], which is a formalism for knowledge representation. In [72], the conceptual graphs are used to represent, characterize, and analyze dependencies between the entities of a model. Cox et al. claim that the use of conceptual graphs makes it easier to model dependencies at multiple levels of detail and when only partial information is available.

Another approach to dependency modeling and analysis is block diagrams [74, 75]. These diagrams are often used in reliability assessments. The diagrams show how components of a system are parallel or serial, and thereby identify possible weak points in the form of dependencies.

4.2.2 Modeling and analysis of service dependencies

There exist a number of approaches that focus on the modeling and analysis of service dependencies and their applications in different domains. In [76], Ensel and Keller present an XML-based model for specifying service dependencies in distributed systems. The purpose of the model is to facilitate information sharing between the different systems. In [77], a UML-based service dependency model for ad hoc collaborative systems is presented. In such systems, different computing devices participate in collaborative applications by using and offering services to each other. Besides handling static aspects of service dependencies, the modeling approach also handles dynamic aspects of service dependencies due to the possibility of devices coming and going. Another approach that also takes dynamic aspects of service dependencies into account is presented in [78]. This approach provides a service dependency classification for system management analysis. It traces the flow of dependency information from the design to the run time stages of services. It relies on two types of models: a function model that defines generic service dependencies, and a structural model containing detailed information on the software components realizing the services.

In [79], Ding and Sha present a dependency algebra, consisting of a formal theoretical framework and a prototype toolkit, for service dependency management in real-time systems. The algebra can be used to ensure that critical components only use and do not depend on services provided by non-critical components. By ensuring that critical services do not depend on non-critical services, the system will not be brought down by minor failures in non-critical services. The algebra also offers a metric for measuring the strength of dependencies.

There are also approaches that address service dependencies with respect to security. Such approaches take into account that the dependencies have security implications, i.e., that a change in the security state of a component providing a service implies a change in the security state of the component requiring the service. One such approach is [80]. This approach is used for constructing formal models of services dependencies in information systems. The constructed dependency models are to be used in security policy-based management. More precisely, the dependency models are used to find enforcement points for security rules, which then support countermeasure deployment, and for computing the impact of attacks and countermeasures that propagate over the information system. A related approach is [81], which presents a framework for risk assessment of information infrastructures. The framework generalizes the notion of dependency with respect to security attributes, such as confidentiality, integrity, and availability. These security dependencies are used to describe relationships between components, to discover attack strategies, and to define risk mitigation plans.

Service dependencies are also used in fault analysis [82] and dependability analysis [83], as well as in analyses targeting critical infrastructures. A number of the approaches that address critical infrastructures use graph-based models to model and analyze service dependencies (see e.g., [84–87]). Moreover, a number of the approaches focus primarily on the consequences of infrastructure services not being provided. One

such approach is [86]. This approach is used to create models of infrastructure systems and their interactions. The models are used in computer simulations where the main purpose is to investigate how the functionality of infrastructure systems and interconnections react to different attack scenarios (“what if” scenarios where one or two systems are removed), and how mechanisms for strengthening the underlying dependency graph can be used.

4.3 Risk analysis

4.3.1 Asset-based risk analysis methods

According to [88], risk analysis is the process to comprehend the nature of risk and to determine the level of risk, whereby a risk is the effect of uncertainty on objectives. For an organization, this effect of uncertainty on objectives may be the result of internal and/or external factors, while the effect may be either positive or negative, since uncertainty is a deviation from the expected.

An asset is something to which a party assigns value and hence for which the party requires protection. Asset-based risk analysis methods focus on identifying the assets that requires protection, and then assess how they may be protected from threats and risks. There exist a number of well-established methods for asset-based risk analysis. Examples include: Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) [9]; CCTA Risk Analysis and Management Method (CRAMM) [10]; (Microsoft) Threat Modeling [89]; and CORAS [11]. In the following we focus on CORAS, since it is the method most relevant for Artifacts 1 and 2.

The CORAS approach for model-based risk analysis consists of the CORAS language for risk modeling; the CORAS method which is a step-by-step description of the risk analysis process, and that comes with a detailed guideline for constructing CORAS diagrams; and the CORAS tool which is used for documenting, maintaining, and reporting risk analysis results in the form of CORAS diagrams.

The CORAS method is structured into eight steps: (1) preparation for the analysis; (2) customer presentation of target; (3) refining the target description using asset diagrams; (4) approval of target description; (5) risk identification using threat diagrams; (6) risk estimation using threat diagrams; (7) risk evaluation using risk diagrams; and (8) risk treatment using treatment diagrams.

The CORAS approach is based on the ISO 31000 standard [88] which is preceded by the AS/NZS 4360 standard [90]. This results in the CORAS method being very similar to other methods that are based on these or similar standards. What really distinguishes the CORAS method from many other methods is its strong focus on the modeling of the target and the risks, for the purpose of supporting communication, documentation, and analysis of the risk analysis results.

4.3.2 Analysis of risk in the context of dependencies

Markov analysis [91] is a stochastic mathematical analysis method that is well-suited for assessing the reliability of systems with component dependencies. In Markov analysis, the system is considered as a number of states, from perfect operation to no operation at all. A Markov model is used to describe the states and the state transitions. The states and the transitions are modeled graphically, and statistical calculations are

used for determining the likelihoods of the different state transitions. The most important weakness of Markov analysis is the high workload and complexity resulting from analyzing large systems.

Dependent CORAS [11] is an approach for modular risk modeling, which can be used to document and reason about risk in the context of dependencies. It extends the CORAS risk modeling language with facilities for documenting and reasoning about risk analysis assumptions. It was motivated by the need to deal with mutual dependencies in risk analysis of systems of systems. By employing dependent CORAS we may document risk separately for the individual systems. In addition, we document the risk analysis assumptions for the different systems. A risk analysis assumption documents how a threat scenario or an unwanted incident of one system may lead to a threat scenario or an unwanted incident of another system. These assumptions are due to some form of dependencies, not necessarily service dependencies, between the different systems. The different risk models may be combined in the end, if the dependencies between them are well-founded, i.e., not circular.

There exist several approaches from the safety domain that apply component-based hazard analysis to describe fault propagation in systems containing dependent components. Giese et al. [92, 93] present a method for compositional hazard analysis of components described in the form of restricted UML [94] component and deployment diagrams. This method applies Fault Tree Analysis [95] (FTA) to describe hazards and the combination of components that causes them. For each component, incoming, outgoing, and internal failures are described, as well as the dependencies between the different failures. The dependency information is used to capture the propagation of failures by combining failure information of the different components. It should be noticed that the method of Giese et al. only considers failures caused by software and/or hardware. Thus, human failures, either accidental or deliberate, are not considered.

In [96], another approach to component-based hazard analysis is described. The approach extends, automates, and integrates well-established risk analysis techniques such as Functional Failure Analysis [97] (FFA), Failure Mode Effect Analysis [98] (FMEA), and FTA. A specialized version of FMEA is applied to describe component output failures of the individual components. The causes of the output failures are described as a logical combination of internal malfunctions of the component or deviations of the component's inputs. Based on the results from the FMEA analyses, fault trees for the components are constructed. The approach synthesizes the individual fault trees in order to describe the fault propagation in the system.

Kaiser et al. [99] present a component concept for FTA. They divide a fault tree into so-called fault tree components. Each fault tree component has incoming and outgoing ports. These ports are used to connect the different fault tree components into a system fault tree. A major advantage of the approach is the ability to reuse fault tree components.

4.3.3 Validation of expert judgments in risk analysis

In [100], Otway and von Winterfeldt describe two different types of processes for making expert judgments in risk analyses; informal and formal processes. Informal processes are implicit, unstructured, and undocumented, while the formal processes are explicit, structured, and documented. In general, formal processes focus more on detecting and resolving biases than informal processes. On the other hand, formal processes are often

time- and cost-consuming, have lack of flexibility, and may result in possible loss of creativity due to the formalism.

Many approaches for the elicitation of expert judgments focus on the aggregation of expert judgments in order to achieve expert judgments of good quality. There are two classes of aggregation methods: behavioral and mathematical. Behavioral approaches (see e.g., [101–103]) focus on negotiation in order to achieve a consensus, while mathematical approaches (see e.g., [104]) are rule or formula based.

4.3.4 Indicator-based risk analysis

In [105], Refsdal and Stølen present an approach to risk monitoring where risk values are calculated from measurable indicators. The approach consists of three main steps. In the first step, a risk analysis of the system is performed, and risks to be monitored are identified. The risk analysis provides information about how threats may exploit vulnerabilities to initiate threat scenarios leading to the risks. In the second step, relevant measurable indicators are identified for risks or vulnerabilities, threat scenarios, etc., leading up to the risks. Functions for calculating likelihood, consequence, and risk values based on indicators are identified in the third step. In [105], the authors also provide guidelines for how to evaluate the internal consistency of the dynamic risk picture. The authors also present a view on how to measure confidence in the dynamic risk picture, based on the discovered internal inconsistencies.

In [106], Baker et al. propose an approach which uses measurable, real-world metrics to improve information security risk assessment and decision making. The main motivation behind the approach is the inability of business leaders to identify the most effective information security strategies for limiting organizational loss. In particular, the approach aims to do the following three things: allow accurate measurement and tracking of threats; enable determination of the impact of loss of successful threats; and aid in evaluating the effectiveness and return on investment of countermeasures.

Breier and Hudec [107] present an approach which uses metrics as an instrument for security risk assessment. In the paper, metrics are used for evaluating the fulfillment of security control objectives. The authors propose a mathematical model based on metrics for evaluating the security control objectives.

4.4 Monitoring

4.4.1 Monitoring of quality

Quality monitoring is often concerned with the monitoring of quality of service (QoS). A large number of approaches focus on the monitoring of service level agreement (SLA) fulfillment. SLAs are used to establish a contract between service providers and consumers concerning quality of service parameters. One such approach is presented in [108]. The paper addresses the problem of service providers that deviate from the SLAs when providing web services. QoS monitoring is necessary for measuring the fulfillment of the SLAs. The problem is that neither the service provider nor the consumer can be trusted when it comes to monitoring. The paper presents QoS monitoring mechanisms that are based on feedback from the consumers. The consumers are running the monitoring code, and they report periodically feedback to a trusted reputation mechanism (RM), which estimates the delivered QoS for the different providers based

on the reports. Providers that do not fulfill their SLAs are penalized by the RM. The RM also applies incentives for the consumers to report honestly.

In [109], Wang et al. present a QoS management framework and QoS management services, including monitoring and diagnostics, for service level management in networked enterprise systems. The monitoring service does not only monitor the fulfillment of SLAs, it also monitors the health of the systems responsible for providing the services. The diagnostics service performs analyses and QoS related diagnostics based on data provided by the monitoring service. If SLA violations or system degradations are detected by the diagnostics service, adaption mechanisms are activated in order to maximize the systems' ability to meet QoS parameters specified in the SLAs.

Monitoring is also employed within business intelligence, often for the purpose of measuring achievement of business objectives. Data [110] and process [111] mining tools are two types of tools that do some sort of monitoring. Within business intelligence, data mining uses techniques from statistics and artificial intelligence to identify interesting patterns in often large sets of business data, while process mining is used to extract information about business processes by the use of event logs. Another type of tools that rely on monitoring is business performance management [112] tools. These tools are used to monitor, control, and manage the implementation of business strategies.

4.4.2 Dynamic risk monitoring

In [113], Trad et al. present a distributed system monitoring application called Factors Estimation System (FES), which is an application that can be used for proactive monitoring of information system risk and quality. By monitoring different sources to problems, the application can detect problems before they become critical. When problems are detected, reports and alarms are generated. Another tool for monitoring, called MASTER ESB, is presented in [114]. This tool is used to monitor compliance with access and usage policies in a system. The tool monitors low-level evidence data that is aggregated into meaningful evidence on how different parties comply with the policies. This evidence is then evaluated, and actions against compliance violations may be taken.

NIST [115] provides a guideline for information security continuous monitoring (ICSM). NIST defines ICSM as “*maintaining ongoing awareness of information security, vulnerabilities, and threats to support organizational risk management decisions.*” In this context, ongoing means that “*security controls and organizational risks are assessed and analyzed at a frequency sufficient to support risk-based security decisions to adequately protect organizational information.*” The purpose of the guideline is to assist organizations in the development of an ICSM strategy and the implementation of an ICSM program. The guideline describes the fundamentals of ongoing monitoring of information security in support of risk management, and it describes the process of ICSM, including implementation guidelines.

Risk monitoring is also central in approaches that focus on the protection of critical infrastructures (see e.g., [116,117]), as well as in approaches that focus on the protection of computerized networks (see e.g., [118,119]).

4.4.3 Patterns and the implementation of monitoring tools

The two main categories of patterns within software engineering are design patterns [23] and architectural patterns [24]. As already mentioned in Section 2.1.6, the difference between a design pattern and an architectural pattern is that design patterns describe design solutions at the object/class level, while architectural patterns describe design solutions at the architectural level, e.g., design of components and their relationships. There exist a number of different templates for describing patterns (see e.g., [23, 24]). It may of course be discussed what classifies as a good description of a pattern, but in general the description must capture the essence of solving the recurring problem in such a way that the pattern is easy to learn, compare, and use [23, 24].

In this thesis we focus on architectural patterns. The Model-View-Controller (MVC) pattern [24, 120] is one of the best-known examples of architectural patterns. MVC divides an interactive application into three main components: model, view, and controller. The model encapsulates core data and functionality, while views and controllers together comprise the user interface of the application. Each view displays data obtained from the model in a specific way, while the controllers are used to handle user input. The MVC pattern makes it easy to change the user interface of an interactive application, since the model is independent of the user interface.

Design patterns that specifically target the building of software health monitoring applications are described in [121]. The paper focuses on design patterns for sensors collecting information about the internal state and operation of software, and how this information can be combined into software health indicators describing different aspects of software health.

The tool framework called Mozart [122] uses a model driven approach to create monitoring applications that uses key performance indicators (KPIs). The framework mines KPIs from a data warehouse and builds an initial KPI net. In this net there will be KPIs that are central for reaching a goal. These KPIs are identified by the use of a goal model. There will also be other KPIs that can directly or indirectly influence the KPIs that are central for reaching the goal. In the next step, the framework makes use of the goal model and historical data on the different KPIs to discover how the different KPIs correlate with each other. This step results in a new KPI net. The new net contains dependencies that specify the correlations between the different KPIs. The next step is then to discover which of the dependency chains in the KPI net that are most influential for monitoring the achievement of the goal. After having identified these chains, a monitor model is constructed. This model can be transformed into a monitor application.

In [123], the design and implementation of a performance monitoring tool for clustered streaming media server systems is presented. The tool focuses on monitoring resources such as CPU utilization, memory usage, disk usage, and network bandwidth. In addition to the tool presented in [123], there are also commercial monitoring solutions that focus on similar monitoring tasks. Examples include SolarWinds ipMonitor [124] (for monitoring network devices, servers, and applications) and the IBM Tivoli Monitoring software [125] (for monitoring operating systems, databases, and servers in distributed and host environments).

4.5 Trust

4.5.1 Estimating the trustworthiness of external services

In [117, 126], the challenge of security risk assessment in interdependent critical infrastructures is addressed. To assess risk, a critical infrastructure operator needs information about services provided by its own infrastructure, as well as information about services provided by infrastructures that its own infrastructure depends on. Thus, in the approach of [117, 126], risk information is shared between the different interdependent critical infrastructures. The problem is that information provided by one critical infrastructure may be inaccurate. Such information will again affect the correctness of the risk assessment results. To tackle this problem, [117, 126] use trust indicators to classify how accurate the exchanged information is. Information that is not trusted may be given a low weight during the risk assessment or it may be discarded.

Subjective logic [127, 128] is another approach that can be used to deal with uncertainty. It is a probabilistic logic that captures uncertainty about probability values explicitly. The logic operates on subjective belief about the world. Different actors have different subjective beliefs, and these beliefs are associated with uncertainty. The approach makes it possible, for example, to calculate to what degree an actor believes that a service will be provided based on the actor's beliefs about services that the service in question depends on, or to calculate the consensus opinion of a group of actors. Subjective logic deals strictly with the actors' beliefs and reasoning, and does not address the question of how their beliefs affect their behavior. The belief calculus of subjective logic can be applied in risk analysis to capture the uncertainty associated with such analysis, as shown in [129]. This is achieved by using subjective beliefs about threats and vulnerabilities as input parameters to the analysis. Through application of the belief calculus, the computed risk assessments provide information about the uncertainty associated with the result of the analysis.

A Bayesian network [130] is a directed acyclic graph consisting of nodes with states and edges describing causal relationships between nodes. Each node is characterized by a probability distribution over its possible states, where these probabilities depend on the probabilities of the states of its parents. The probabilities of nodes are changed by the gathering of new evidence. For any changes of the probabilities of nodes, the effects both forwards (towards child nodes) and backwards (towards parent nodes) may be computed.

In [131], Wu et al. present a Bayesian network based QoS assessment model for web services. The purpose of the model is to predict whether different service providers can deliver a service that satisfies service consumers' QoS requirements. The model is trained by computing the compliance between consumers' QoS requirements and the QoS of the delivered service. The capability of the service to deliver the correct QoS is inferred based on the compliance values, and the Bayesian network is updated by using the inference outcome. In [132], Melaye and Demazeau propose a Bayesian dynamic trust model for determining an agent's trust in another agent. The model can for instance be used to determine whether an agent delivers a service with the expected quality. The notion of trust is formalized by using a Bayesian network that is structured into three layers. The top layer is the trust level, while the second and third level represent basic beliefs and belief sources, respectively. A Bayesian Kalmar filter is used to capture dynamic aspects of trust, e.g., changes in beliefs. In [133], a Bayesian

network trust model for peer-to-peer networks is proposed. A peer can use trust to assess other peers' ability to provide services with the expected quality. In [133], a peer calculates trust by the use of Bayesian networks. The peer maintains a Bayesian network for each peer that it has received services from. After each service interaction, the peer updates the Bayesian network based on the quality of the provided service. The trust values calculated by the Bayesian networks are used to rank the different service providers.

Fuzzy Logic [134] is a form of reasoning for computers that is very close to human reasoning. It is used to draw conclusions from uncertain, vague, ambiguous, or imprecise information. Fuzzy logic is therefore suitable for reasoning about trust, since trust assessments are often based on such information. In Fuzzy logic, we first perform a fuzzification by gathering crisp numerical values that are assigned to fuzzy sets by the use of fuzzy linguistic variables, fuzzy linguistic terms, and membership functions. A crisp numerical value can be the member of more than one fuzzy set. The degree of membership in a set ranges between 0 and 1 in fuzzy logic. Thus, a crisp value that represent some assessment of service quality can have a membership degree of 0.1 in the set "low" (quality) and a membership degree of 0.8 in the set "medium" (quality). After the fuzzification has been conducted, the fuzzy values are used to evaluate a set of IF ... THEN ... rules. A rule can for instance say: IF quality is low THEN trust is low. Afterwards, the results from the evaluation of the rules are aggregated. The last step is to perform a defuzzification by mapping the aggregation results to crisp numerical values. Such a crisp numerical value can for instance specify the trust in a service provider's ability to provide a service with the required quality.

In [135], Griffiths et al. present a trust model based on fuzzy logic for peer-to-peer systems. The trust model uses fuzzy logic to represent and reason with imprecise and uncertain information regarding peers' trustworthiness. A peer can use the model to select the most appropriate service providers among the other peers. More precisely, the model enables a peer to maximize the quality of the services that it requires according to its current preferences.

4.5.2 Trust-based decisions in interconnected systems

Reputation systems [136] are often used to decide whom to trust on the Internet. Such systems collect, distribute, and aggregate feedback about participants' past behavior. In [137], Resnick and Zeckhauser present an empirical analysis of eBay's reputation system. The analysis was based on a large data set from 1999 provided by eBay. The analysis resulted in discoveries such as: feedback was provided more than half the time; the feedback was almost always positive; and sellers' feedback profiles were predictive of future performance. The authors found the low rate of negative feedback highly suspicious. Despite this, the authors come to the conclusion that the system appears to be working. One of the explanations that the authors consider is that the system may still work, even if it is unreliable and unsound, if its participants think it is working. Thus, if sellers believe that a strong reputation is needed in order to sell goods, then they will behave in ways that result in positive feedback.

In [138], Lim et al. investigate the effectiveness of two trust-building strategies to influence actual buying behavior in online shopping environments, particularly for first-time visitors to an Internet store without an established reputation. The two strategies investigated were portal association (e.g., the store is associated with Yahoo,

Amazon, etc.) and satisfied customer endorsements. Two studies were conducted at a large public university in Hong Kong with students as test subjects. Of the two strategies investigated, satisfied customer endorsements by similar (local, non-foreign) peers was found to increase the test subjects' trust in the online store investigated in the two studies. Portal association did not lead to an increase in trust.

In addition to the two studies presented above, a number of other surveys, reviews, and empirical studies on trust and reputation approaches have been conducted (see e.g., [139–142]).

Chapter 5

Summary of contribution

This thesis makes two kinds of contributions. Firstly, it contributes in terms of new artifacts. Secondly, it contributes in terms of industrial case studies. The industrial case studies were mainly carried out to support the development of the artifacts, but since the industrial case studies also provide insight into issues of a more general nature, they may be seen as contributions on their own. The three artifacts are

1. *Artifact 1*: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators
2. *Artifact 2*: Method for capturing and monitoring the impact of service dependencies on the quality of provided services
3. *Artifact 3*: Architectural pattern for constructing enterprise level monitoring tools based on indicators

The two industrial case studies are

1. Empirical study on trust-based decisions in interconnected systems
2. Empirical study on the design of indicators for monitoring risk

Although the three new artifacts may be viewed as a contribution on their own, and also be employed and used in practice independently of each other, they are also closely related and may be integrated. As indicated by Figure 5.1, we refer to the result of this integration (which is represented by the gray circle) as the *ValidKI Framework*, while we refer to the application of Artifacts 1, 2, and 3 within the framework as *Indicator Design*, *Dependency Analysis*, and *Monitoring Pattern*, respectively.

The gray circle does only cover parts of the three artifacts, since the artifacts have applications that go beyond the framework. In the framework, Artifact 1 is applied to business objectives focusing on service quality. The artifact is however not limited to this kind of business objectives only. The artifact can be used to design indicators to monitor the fulfillment of all sorts of business objectives focusing on quality, as long as ICT-supported monitoring of the indicators is possible.

In the framework, Artifact 2 is used to capture the impact of service dependencies on risk to the fulfillment of business objectives with respect to service quality. In this context, all unacceptable risks to the fulfillment of a business objective need to be selected for monitoring. When applied independently of the framework, the focus is

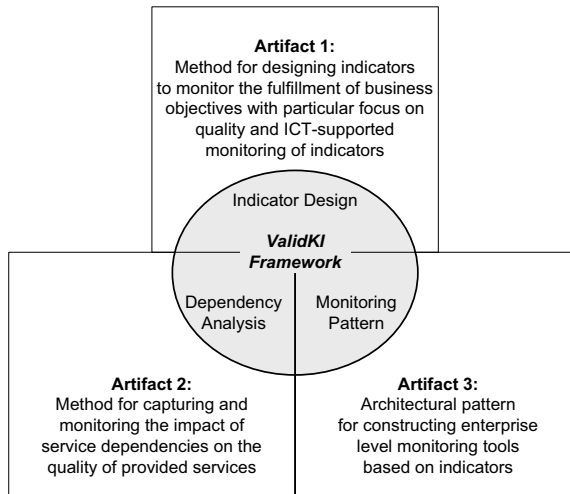


Figure 5.1: Relations between the ValidKI Framework and Artifacts 1–3

still on service quality, but not on business objectives focusing on the achievement of service quality. Then it is up to the client on whose behalf the artifact is applied to select the risks to be monitored.

In the framework, Artifact 3 is used to implement a risk cockpit based on indicators that facilitates the monitoring of business objectives focusing on the achievement of service quality. Artifact 3 is however not limited to the implementation of risk cockpits only. It may be used to implement all sorts of enterprise level monitoring tools based on indicators.

The remainder of this chapter consists of three sections. In Section 5.1 we present the ValidKI Framework. In Section 5.2 we provide an overview of the three contributed artifacts. Finally, Section 5.3 is devoted to the two industrial case studies.

5.1 The ValidKI Framework

In Figure 5.2, the ValidKI Framework is described. The figure shows in particular the integration of Indicator Design and Dependency Analysis corresponding to Artifacts 1 and 2, respectively. Arrows have been used to show in which order the different steps of the two methods are executed. Moreover, bold arrows are used to indicate inputs to and outputs from the artifacts. It should also be noticed that the grayed steps of Indicator Design method are not executed. These steps are replaced by steps of the Dependency Analysis method. In the following we explain the integration of the three artifacts. We use *Step X (ID)* to refer to Step X of Indicator Design, while we use *Step Y (DA)* to refer to Step Y of Dependency Analysis.

The input to the framework is a business objective focusing on the achievement of service quality. In Step 1.1 (ID) the business objective is expressed more precisely in order to understand exactly what it means to fulfill it. Step 1.2 (ID) is used to describe the part of the business that needs to reach the business objective and therefore is to be monitored. With this relevant part of business being interconnected systems that

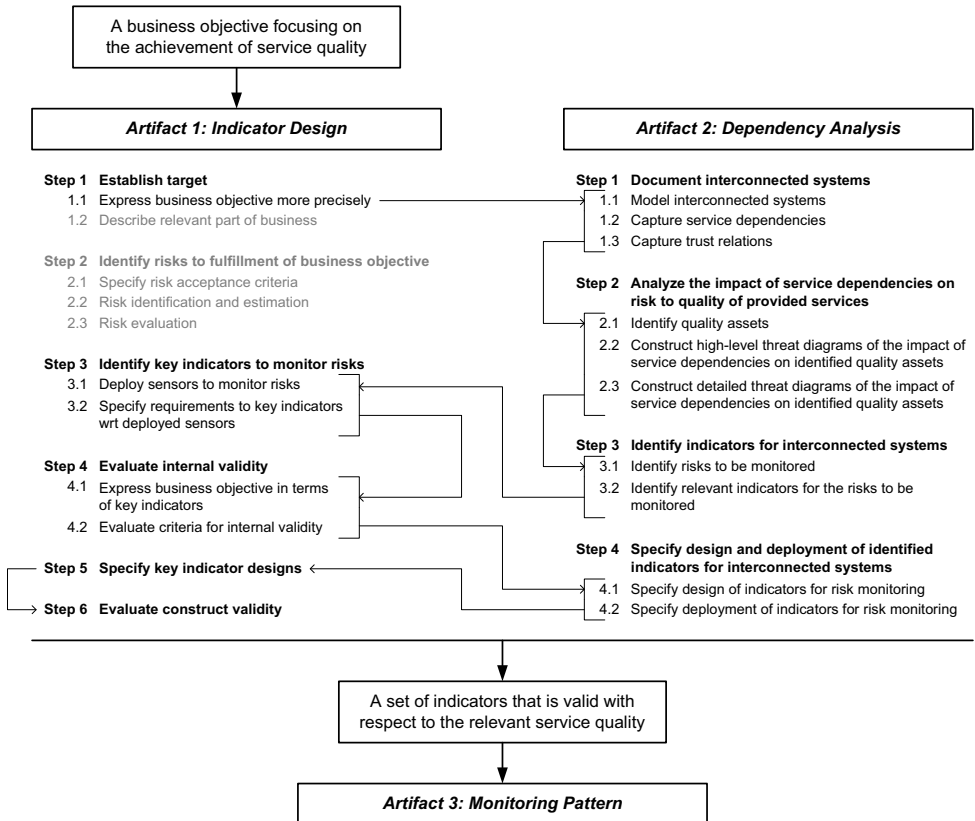


Figure 5.2: The ValidKI Framework

depend on each other through service interactions, we replace Step 1.2 (ID) with Step 1 (DA) in order to document the interconnected systems, their services, service dependencies, quality requirements to services, and trust in services provided by systems of which we have insufficient documentation. Moreover, since we need to capture the impact of service dependencies on risk to the fulfillment of the business objective, Step 2.1 (ID) and Step 2.2 (ID) are replaced by Step 2 (DA).

The result of conducting Step 2 (DA) is a risk model capturing the impact of service dependencies on the fulfillment of the business objective. In Step 3.1 (DA), which replaces Step 2.3 (ID), we identify the risks that should be monitored by the use of indicators. All risks that are unacceptable with respect to the fulfillment of the business objective are identified for monitoring. In Step 3.2 (DA) we identify indicators for monitoring the risks, while in Step 3 (ID) we specify deployments of sensors in the interconnected systems, and we specify requirements to the indicators with respect to the sensor deployments. The sensors are needed for gathering the data necessary for calculating the indicators. After having evaluated the internal validity of the indicators in Step 4 (ID), the design and deployment of indicators are specified in Step 4 (DA). The designs specify in the form of algorithms how indicators should be calculated, while the deployments specify how data needed in the calculations should be extracted and transmitted within the relevant part of business. In Step 5 (ID), the specifications from Step 4 (DA) are refined. The result of Step 5 (ID) is specifications that describe how different sensors, actors, and components/systems need to interact in order to calculate the indicators. The final step is to evaluate the construct validity of the indicators. A risk analysis is conducted as part of Step 6 (ID). To analyze the risks we rely on Step 2 (DA) due to the dependencies between the interconnected systems.

The output from the two integrated artifacts is a set of indicators for monitoring the fulfillment of the business objective received as input. The output is used as input to Monitoring Pattern corresponding to Artifact 3. Based on the input, a risk cockpit that facilitates monitoring of the business objective can be constructed.

5.2 Overview of artifacts

5.2.1 **Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators**

The methodology is used for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators. For a detailed description of the methodology, we refer to Chapter 9.

The focus of the methodology is on designing valid indicators. A set of indicators is valid with respect to a business objective if it measures the degree to which the business or relevant part thereof fulfills the business objective. Six main steps are conducted in order to design valid indicators for a business objective. The first main step is all about understanding what needs to be monitored. More precisely, we need to express the business objective in a precise manner, and we need to provide a description of the part of the business that needs to reach the business objective. The second main step is concerned with conducting a risk analysis to identify risks to the fulfillment of the

business objective. We distinguish between three sub-steps. Risk acceptance criteria are specified in the first sub-step, while risks are identified in the second sub-step. In the third sub-step, the identified risks are evaluated with respect to the specified risk acceptance criteria.

The third main step is concerned with identifying indicators to monitor the unacceptable risks identified in the previous main step. We distinguish between two sub-steps. Sensors to be deployed in the relevant part of business are identified in the first sub-step. In the second sub-step we identify indicators to be calculated based on data gathered by the sensors and we specify requirements to the indicators with respect to the deployed sensors. The internal validity of the set of indicators is evaluated in the fourth main step. We distinguish between two sub-steps. In the first sub-step we reformulate the precise business objective by expressing it in terms of the identified indicators. In the second sub-step we evaluate the internal validity of the set. The set is internally valid if the precise business objective expressed in terms of the indicators correctly measures the degree to which the business objective is fulfilled. For each indicator we evaluate whether it is internally valid based on a set of criteria.

In the fifth main step we specify designs for indicators, i.e., how they should be calculated based on data gathered by the sensors, while in the sixth main step, we evaluate whether the set of indicators has construct validity with respect to the business objective. The set has construct validity if the gathering of the sensor measurements of each indicator is suitable with respect to its requirements. Construct validity is evaluated based on a set of criteria. To evaluate the different criteria, we re-do the risk analysis from the second main step with the precise business objective replaced by the reformulated precise business objective. The latter objective is the precise business objective expressed in terms of indicators. For each indicator we identify risks towards the correctness of the reformulated precise business objective that are the result of threats to criteria for construct validity that the indicator needs to fulfill. If the risk analysis does not result in any new unacceptable risks, then we have established construct validity for each indicator. If the set of indicators is both internally valid and has construct validity with respect to the business objective, then we have established that the set is valid.

5.2.2 Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services

The methodology is used for capturing the impact of service dependencies on risk to the quality of provided services in interconnected systems, and for setting up monitoring of selected risks by the use of indicators for the purpose of providing a dynamic risk picture for the provided services. The methodology is described in detail in Chapter 10.

The client, on whose behalf the methodology is applied, controls some of the interconnected systems. These systems depend on other systems, which are controlled by other parties. In the first main step of the methodology, we document the interconnected systems in the form of a target model. The model documents systems, services, quality requirements to services, service dependencies, and trust relations. The trust relations are used when analyzing service dependencies involving systems of which we

have insufficient documentation. Each relation assigns a trust level to a quality requirement. The trust level states the degree to which the client trusts the service to be delivered according to the quality requirement in question. These trust levels are used in the second main step.

In the second main step, we conduct a risk analysis to capture the impact of service dependencies on risk to quality of provided services. For a provided service, its quality is represented by a number of quality attributes. For each provided service to be analyzed, we identify in the first sub-step one or more quality assets, where each asset represents a quality attribute of the service. By identifying these assets we restrict the identification of risks caused by service dependencies to only those risks that may harm the quality of provided services. The next sub-step is to construct high-level risk models of the impact of service dependencies on the identified quality assets. The models are constructed schematically from the target model by following a schematic procedure. These models establish a high-level understanding of how the failure of services to be delivered according to their quality requirements may lead to the failure of dependent services to be delivered according to their quality requirements. In the third sub-step, these high-level models are further detailed in order to establish a risk picture that can be monitored. As part of this detailing, we use the trust levels identified in the first main step to estimate likelihoods of quality requirements not being achieved for services involving systems of which we have insufficient documentation.

The third main step concerns the identification of risks to be monitored, as well as identification of indicators for monitoring their risk values. In the fourth main step we specify the design and deployment of the identified indicators. The designs specify how the indicators should be calculated, while the deployments specify how the indicators should be embedded in the interconnected systems. More precisely, the deployments specify how data needed in the calculations should be extracted and transmitted within the interconnected systems.

5.2.3 Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators

The architectural pattern serves as a basis for implementing monitoring tools, or more specifically enterprise level monitoring tools based on indicators. These are tools that: collect low-level indicators from the ICT infrastructure or similar; aggregate the low-level indicators into high-level indicators, useful at the enterprise level; and present the high-level indicators in a way that is understandable to the intended users. In the following we explain how the pattern is applied for constructing an enterprise level monitoring tool based on indicators. For a detailed description of the architectural pattern, we refer to Chapter 11.

The architectural pattern divides an enterprise level monitoring tool into three components; *MonitorModel* which collects low-level indicators from some data source and aggregate these indicators into high-level indicators; *MonitorConsole* which presents results from the monitoring in the form of high-level indicators and results from the evaluation of these indicators in a specific way to a specific group of users; and *MonitorConfig* which is used to set up the enterprise level monitoring tool and to configure it during run-time.

The *MonitorConfig* makes use of a data configuration file to configure the *MonitorModel*. This file specifies the low-level indicators to be collected and functions for

aggregating these indicators into high-level indicators. The MonitorConsole is configured by the use of a presentation model. This model specifies how results from the monitoring should be presented to a specific group of users. The presentation model is parameterized with respect to high-level indicators found in the MonitorModel's data configuration file. The MonitorConsole can therefore easily be updated when high-level indicators, referred to in its presentation model, change. In the case of risk monitoring, the presentation model will typically be a risk model. In this risk model, some of the likelihood and consequence values, used to calculate risk values, have been replaced by high-level indicators. Likelihood and consequence values will be updated as a result of high-level indicators being updated, which again results in risk values being updated.

5.3 Overview of industrial case studies

5.3.1 Empirical study on trust-based decisions in interconnected systems

The first empirical study was conducted as part of an industrial project focusing on the use of a UML-based trust analysis method to model and analyze a public eProcurement system. This system makes use of a Validation Authority (VA) service for validating electronic IDs (eIDs) and digital signatures. The trust analysis was conducted on behalf of Det Norske Veritas (DNV) in the autumn of 2008. The goal of the trust analysis was to obtain a better understanding of the potential usefulness of a VA service for supporting trust-based decisions in systems which rely on electronically signed documents. The empirical study and the industrial project are described in detail in Chapter 12. In the following we provide an overview of the industrial project and the trust analysis method used, and we describe the main results from the empirical study.

Public eProcurement is used by public authorities, for instance within the EU, to award public contracts to economic operators. To apply for public contracts, economic operators submit electronically signed tenders, containing legal, financial, and technical information, to a public eProcurement system. A public eProcurement system must be aware of the risk implied by accepting digital signatures. The eProcurement system does not have information on whether the tender is authentic or not, i.e., whether the economic operator is willing or not willing to fulfill the tender. Thus, a possible scenario is that an economic operator can refute the validity of the submitted tender, if awarded the contract. The system can ensure that this risk is acceptable by making an assessment of the signature quality and accepting only those of a certain quality. The higher the quality is, the harder it would be for an economic operator to refute the validity of their tender. The quality of a signature can be decided from the quality of the eID, which is derived from the certificate policy of the certificate authority and the cryptography used. The eProcurement system can use a VA to assess the quality of digital signatures with respect to a quality policy set by the system. Based on the assessments, the VA informs the system whether the signatures are to be trusted or not.

The trust analysis method used for modeling and analyzing the eProcurement system consists of three main steps. In the first step we model the target. The models should only capture the aspects of the system (eProcurement system) and other actors

(the VA and economic operators) that enhance our understanding of the decisions that are taken on the basis of trust and the considerations that lie behind these decisions, as well as the resulting system behavior and outcomes that are relevant. In the second step we conduct the actual analysis. This involves investigating the current system behavior and the way in which trust-based decisions are being made, as well as potential alternative behaviors. The aim is to obtain a good understanding of the risks and opportunities involved. For an eProcurement system the risks are to accept trusted tenders that are non-authentic and to reject not trusted tenders that are authentic. Moreover, the opportunities are to accept trusted tenders that are authentic and to reject not trusted tenders that are non-authentic. In the third step we use the obtained knowledge to form policies to ensure and enforce the desirable behavior. The aim here is to select a quality policy that results in an optimal balance between risks and opportunities.

The empirical study motivates the need for using trust analysis methods when reasoning about the behavior of systems/actors in cases where the behavior of these systems/actors may result in risks and/or opportunities. For the particular method applied, the empirical study gave strong indications that the trust analysis method is feasible in practice. The empirical study also shows that: this kind of trust analysis can be carried within the frame of 100 man-hours (not including writing of a final report); there were no instances where the analysts (researchers) were not able to capture the relevant information in the models; and the models to a large extent were comprehensible for the industrial participant with some experience in UML but no background in the specific extensions used by the method.

5.3.2 Empirical study on the design of indicators for monitoring risk

The second empirical study was integrated in a commercial security risk analysis conducted in 2010. In this analysis, indicators were designed for the purpose of validating likelihood estimates obtained from expert judgments. In the following we provide a brief overview of the steps of the commercial security risk analysis that were of relevance to the empirical study, and we present the main results from the study. We refer to Chapter 13 for further details on the commercial security risk analysis and the empirical study.

The commercial security risk analysis included a six step process that was of relevance to the empirical study. The empirical study builds on data collected during the analysis and on semi-structured interviews with domain experts that participated on behalf of the client in the analysis. In Step 1 of the process, the domain experts provided the analysis team (the researchers) with likelihood estimates for security risks based on expert judgments. Indicators for validating the likelihood estimates were identified in Step 2. The analysis team designed a number of indicators, and these indicators were revised during a meeting with the domain experts in Step 3. During this meeting some indicators were rejected, some were subject to minor modifications, and some new indicators were identified. In Step 4 the analysis team formulated validation criteria for the likelihood estimates in terms of indicators. Each criterion specifies the expected values of the indicators related to the likelihood estimate in question. Here, each criterion makes a prediction about the value of a set of indicators under the assumption that the likelihood estimate in question is correct. Indicator values

were obtained by the domain experts in Step 5. In Step 6 the validation criteria were evaluated and some of the initial likelihood estimates were adjusted.

One result from the empirical study was that two out of 28 likelihood estimates were adjusted, while the main result was the identification of a number of challenges related to design of indicators for monitoring security risks. First, the empirical study shows that it is challenging to design indicators for which it is feasible to obtain values within the available time and resources of a security risk analysis. For a number of the indicators designed, their values were not obtainable within the client's organization. By having some knowledge on the kinds of historical data that are available within the organization and whose responsible for the different kinds of data, it should be easier to both identify indicators and obtain their values. Unfortunately, it may be difficult to obtain this knowledge since data is often spread across the organization and since few, if any, have a complete overview of the data available. Second, it is challenging to relate likelihood estimates to indicators. It is especially difficult to predict how indicator values affect a likelihood estimate when the indicators are only indirectly related to the estimate in question. This will typically be a problem when formulating validation criteria for likelihood estimates of incidents that are not easily observable. Third, the indicator values obtained from an organization may vary when it comes to correctness. In order to get the most out of indicator-based monitoring, the uncertainty of the values should be taken into account. Moreover, one should strive to reduce uncertainty by using several independent indicators to calculate/validate the same estimate.

Chapter 6

Overview of research papers

The main results of the work presented in this thesis are documented in the papers presented in Part II of the thesis. In this chapter we provide an overview of these papers. For each paper we describe its main topics and indicate how much of the work that is credited to the author of this thesis.

6.1 Paper A: ValidKI: A method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators

Authors: Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen.

Publication status: Technical report SINTEF A23413, SINTEF ICT, 2012. The report presented in the thesis is an extended and revised version of the paper published in International Journal on Advances in Intelligent Systems (vol. 5, no. 1-2, 2012) [143]. This paper is again an extended and revised version of the paper published in proceedings of the First International Conference on Business Intelligence and Technology (BUSTECH'2011) [144]. The latter paper received a best paper award at the conference.

My contribution: Olav Skjelkvåle Ligaarden was the main author, responsible for about 90% of the work.

Main topics: The report presents the method ValidKI (Valid Key Indicators), which is a method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators. The main focus of the method is on the design of valid indicators. A set of indicators is valid with respect to a business objective if it measures the degree to which the business or relevant part thereof fulfills the business objective. The method is divided into six main steps. In the report, the method is demonstrated on an example case focusing on the use of electronic patient records in a hospital environment. The main output from the method is specifications that describe the design and deployment of

the indicators. These specifications are to be used to implement ICT-based monitoring of the indicators.

In ValidKI, indicators are designed for monitoring unacceptable risks to the fulfillment of business objectives. A business objective is fulfilled if all of its unacceptable risks become acceptable as a result of the monitoring. Acceptable risks to the fulfillment of business objectives may be thought of to represent uncertainty we can live with. In other words, their potential occurrences are not seen to significantly influence the fulfillment of the business objectives. The validity of the indicators is evaluated based on validation criteria from the domain of software engineering metrics. The validation criteria used are general, thus not specific to software engineering.

6.2 Paper B: Using indicators to monitor risk in interconnected systems: How to capture and measure the impact of service dependencies on the quality of provided services

Authors: Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen.

Publication status: Technical report SINTEF A22301, SINTEF ICT, 2012. The report presented in the thesis is an extended and revised version of the paper published as chapter in the book “IT Security Governance Innovations: Theory and Research” (D. Mellado, L. E. Sánchez, E. Fernández-Medina, and M. Piattini (eds.), IGI Global, 2012) [145].

My contribution: Olav Skjelkvåle Ligaarden was the main author, responsible for about 90% of the work.

Main topics: The report presents a method for capturing the impact of service dependencies on risk to the quality of provided services in interconnected systems, and for setting up monitoring of selected risks by the use of indicators for the purpose of providing a dynamic risk picture for the provided services. The method is divided into four main steps focusing on documenting the interconnected systems and their service dependencies, establishing the impact of service dependencies on risk to quality of provided services, identifying measurable indicators for dynamic monitoring, and specifying their design and deployment, respectively. These design and deployment specifications are to be used to implement risk monitoring based on the identified indicators. The report describes each of the four main steps as well as their sub-steps in terms of a detailed guideline. The method is illustrated in an example-driven fashion based on a case study from the domain of power supply.

6.3 Paper C: An architectural pattern for enterprise level monitoring tools

Authors: Olav Skjelkvåle Ligaarden, Mass Soldal Lund, Atle Refsdal, Fredrik Seehusen, and Ketil Stølen.

Publication status: Published in proceedings of the 2011 IEEE International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA'2011) [146].

My contribution: Olav Skjelkvåle Ligaarden was the main author, responsible for about 90% of the work.

Main topics: The paper presents an architectural pattern to serve as a basis for building enterprise level monitoring tools based on indicators. These are tools that: collect low-level indicators from the ICT infrastructure or similar; aggregate the low-level indicators into high-level indicators, useful at the enterprise level; and present the high-level indicators in a way that is understandable to the intended users. In the paper we identify the core components of such tools and describe their interactions. The pattern is the result of generalizing the architecture of a risk monitor that exhibits a number of features that are not specific to the monitoring of risks, but general to a broad class of enterprise level monitoring tools. In the paper, we demonstrate the pattern by showing the risk monitor as an instance, and we exemplify the use of the risk monitor in a health care scenario.

Errata: Replace “design pattern” with “architectural pattern” in Section V (Conclusion) in Paper C.

6.4 Paper D: Experiences from using a UML-based method for trust analysis in an industrial project on electronic procurement

Authors: Tormod Vaksvik Håvaldsrud, Olav Skjelkvåle Ligaarden, Per Myrseth, Atle Refsdal, Ketil Stølen, and Jon Ølnes.

Publication status: Published in Journal of Electronic Commerce Research (vol. 10, no. 3-4, 2010) [147].

My contribution: Olav Skjelkvåle Ligaarden was one of two main authors, responsible for about 45% of the work.

Main topics: The paper reports on experiences from using a UML-based method for trust analysis in an industrial project. The overall aim of the trust analysis method is to provide a sound basis for making trust policy decisions. The method makes use of UML sequence diagrams extended with constructs for probabilistic choice and subjective

belief, as well as the capture of policy rules. The trust analysis method is evaluated with respect to a set of criteria. The industrial project focused on the modeling and analysis of a public electronic procurement (eProcurement) system making use of a validation authority service for validating electronic certificates and signatures. The evaluation of the method gave strong indications that the trust analysis method is feasible in practice.

6.5 Paper E: Experiences from using indicators to validate expert judgments in security risk analysis

Authors: Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen.

Publication status: Technical report SINTEF A21560, SINTEF ICT, 2012. The report presented in the thesis is an extended version of the paper published in the proceedings of the Third International Workshop on Security Measurements and Metrics (MetriSec'2011) [148].

My contribution: Olav Skjelkvåle Ligaarden was the main author, responsible for about 90% of the work.

Main topics: The report presents experiences from a commercial security risk analysis where indicators were used to validate likelihood estimates obtained from expert judgments. The experiences build on data collected during the analysis and on semi-structured interviews with the client experts who participated in the analysis. The empirical study identified several challenges related to the design of indicators for monitoring security risks.

Chapter 7

Discussion

In this chapter, we evaluate and discuss the contributions of this thesis. In Chapter 2 we defined success criteria for the framework and the three artifacts. In Section 7.1 we evaluate to what extent we have fulfilled the different success criteria, while in Section 7.2 we discuss how the three artifacts of this thesis relate to and extend the state of the art presented in Chapter 4.

7.1 Fulfillment of the success criteria

In the following three sub-sections we evaluate the success criteria of Artifacts 1–3. The success criterion of the framework as a whole is evaluated in the fourth sub-section, since this criterion depends on the evaluation of the other success criteria.

7.1.1 Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators

Success criterion 2 *The application of the method results in indicators that measure correctly to what extent the business objectives received as input are fulfilled.*

As previously explained, the industrial case study described in Paper E (presented in Chapter 13) focused on the use of indicators to validate likelihood estimates obtained from expert judgments. The need for indicators that measure correctly to what extent the business objectives received as input are fulfilled is motivated by experiences from this study. In particular, the study identified two cases where likelihood estimates could not be validated because the indicator values were too uncertain.

The method delivers a set of indicators that is valid with respect to a business objective received as input, where valid is defined as follows: *a set of indicators is valid with respect to a business objective if it is valid in the following two ways:*

1. **internal validity** – *the precise business objective expressed in terms of the indicators correctly measures the degree to which the business objective is fulfilled;*
and

2. **construct validity** – *the gathering of the sensor measurements of each indicator is suitable with respect to its requirements specification.*

We claim that our notion of validity is a good approximation of correctness referred to in the success criterion. Firstly, the distinction between internal and construct validity is well-established in the literature. According to Meneely et al. [63], a metric has internal validity if *“the metric measures the attribute it purports to measure.”* Similar definitions are given in [67,149]. Moreover, [63] presents a systematic literature review of validation criteria for software metrics. The review ultimately ended up focusing on 20 papers. Most of the authors of these papers discuss some form of internal validation.

According to Meneely et al., a metric has construct validity if *“the gathering of a metric’s measurements is suitable for the definition of the targeted attribute.”* This definition is based on [150] which refers to construct validity as when *“the operational definition yields data related to an abstract concept.”* In both cases, construct refers to the implementation of a metric/indicator. As shown by Meneely et al., a number of the authors of the reviewed papers discuss validation criteria that can be classified as construct validity types of criteria, i.e., that they concern the validation of the implementation of metrics/indicators.

Our definitions of internal and construct validity follow the definitions given by Meneely et al. As can be seen in Section II-B in Paper A (presented in Chapter 9), we use a risk-based approach to evaluate the fulfillment of business objectives. For each business objective, we identify risks towards its fulfillment. Indicators are used to measure risks that represent uncertainty we cannot live with, i.e., that the potential occurrences of the risks significantly influence the fulfillment of the business objective. More precisely, the indicators are used to measure risk attributes (likelihood or consequence). Each indicator has internal validity if it measures such an attribute correctly.

In the case of construct validity, each indicator is associated with a requirements specification which specifies what it means to measure the risk attribute. The indicator has construct validity if it can be implemented correctly, i.e., if the gathering of the sensor measurements for which the calculation of the indicator relies is suitable with respect to the requirements specification.

By reviewing the 20 papers, Meneely et al. extracted and categorized 47 unique validation criteria for software metrics. A number of these criteria are general, thus not specific to software engineering. For some of the validation criteria, Meneely et al. noticed that the criteria are not atomically satisfiable criteria, but broad categories that can contain other criteria. The main categories are: internal, external, and construct validity. According to Meneely et al., a metric has external validity if *“it is related in some way (e.g., by prediction, association, or causality) with an external quality factor.”* An external quality factor is an external attribute that is measured by a metric. In software engineering, the relationships between internal and external attributes of software products are often examined [151]. For instance, a metric for the internal attribute code size has external validity if it is related to the metric for the external attribute maintainability in some way.

The business objectives that we address focus on quality. The attributes that we measure by the use of indicators are of course related in some way to the quality attributes represented by the business objective. However, external validity is not rele-

vant in our case, since we do not measure to what extent a business objective is fulfilled by relating indicators that measure different attributes to indicators that measure quality attributes. Instead, we measure to what extent a business objective is fulfilled by measuring risk attributes of risks whose potential occurrences may significantly influence the fulfillment of the business objective. This means that none of the validation criteria that Meneely et al. have classified as external validity types of criteria are relevant.

To evaluate internal and construct validity, we have taken a number of the criteria categorized in the internal and construct validity categories into consideration. In the following we justify the selection and rejection of different criteria. In the case of internal validity, the criterion “attribute validity” is taken into consideration. According to Meneely et al., a metric has attribute validity if *“if the measurements correctly exhibit the attribute that the metric is intending to measure.”* This criterion is relevant since it helps us to evaluate whether an indicator correctly exhibits the risk attribute (likelihood or consequence) of the risk that it is measuring.

In [63], the criterion “representation condition” is classified as a category. According to Meneely et al., a metric satisfies the representation condition if *“the attribute is a numerical characterization that preserves properties of both the attribute and the number system it maps to.”* The representation condition is a property from measurement theory. Under the representation condition, any property of the number system must appropriately map to a property of the attribute being measured and vice versa. From this category we have selected the following criteria:

- “appropriate continuity” (definition according to [63]: *“a metric has appropriate continuity if the metric is defined (or undefined) for all values according to the attribute being measured”*);
- “dimensional consistency” (definition according to [63]: *“a metric has dimensional consistency if the formulation of multiple metrics into a composite metric is performed by a scientifically well-understood mathematical function”*); and
- “unit validity” (definition according to [63]: *“a metric has unit validity if the units used are an appropriate means of measuring the attribute”*).

The criterion “appropriate continuity” is relevant since it helps us to check whether the indicator has any unexpected discontinuities. Such discontinuities may for instance arise from fraction calculation with a zero denominator. On the other hand, the criterion “dimensional consistency” is relevant for evaluating whether information is lost during the construction of a composite indicator from basic indicators. Loss of information may for instance be experienced if different scales are used for the basic and composite indicators. And finally, the criterion “unit validity” is relevant for evaluating whether the indicator’s unit is appropriate for measuring the risk attribute.

In the “representation condition” category we also find the criterion “scale validity.” According to Meneely et al., a metric has scale validity if *“it is defined on an explicit, appropriate scale such that all meaningful transformations of the metric are admissible.”* The scales discussed are typically nominal, ordinal, interval, ratio, and absolute. Each scale type comes with a set of admissible transformations. During the evaluation of “dimensional consistency” we can check whether appropriate scales are being used. Thus, it is not necessary to evaluate the criterion “scale validity” separately. The category also contains a number of criteria that are not relevant since they

are specific to software engineering metrics. This is the case for “appropriate granularity,” “interaction sensitivity,” “monotonicity,” “permutation validity,” and “renaming insensitivity.” We also do not find following criteria in the same category relevant:

- “increasing growth validity” (definition according to [63]: *“a metric has increasing growth validity if the metric increases when concatenating two entities together”*) and
- “non-uniformity” (definition according to [63]: *“a metric has “non-uniformity if it can produce different values for at least two different entities”*).

A metric does not have non-uniformity if it for instance rates all software programs as equal. Then the metric is not really a measure. With risks being our entities, neither of the criteria is relevant. In the method in Paper A, risks can be combined if the pull in the same direction. Since this is done before we identify indicators, the criterion “increasing growth validity” is not relevant. In our context, an indicator is used to measure a risk attribute of a risk at different points in time. We have only two requirements to the values produced by an indicator. The first requirement is that the indicator should measure the risk attribute correctly, while the second is that the indicator should not produce values that always result in the risk being unacceptable or acceptable. These requirements are checked during the evaluation of the “attribute validity” criterion. This means that the “non-uniformity” criterion is not relevant.

Besides the software engineering specific validation criteria already mentioned, we also find the category “content validity,” the criterion “product and process relevance” contained in this category, and the two criteria “actionability” and “transformation invariance” most relevant in a software engineering setting. In addition, we do not find the following two criteria relevant:

- “economic productivity” (definition according to [63]: *“a metric has economic productivity if using the metric quantifies a relationship between cost and benefit”*) and
- “non-exploitability” (definition according to [63]: *“a metric exhibits non-exploitability if developers cannot manipulate a metric to obtain desired results”*).

A metric/indicator does not have “economic productivity” if it does not result in saving money in the long run. We do not consider this criterion since it more relevant to consider the cost/benefit of an indicator when taking a decision on whether to implement it or not. In the case of the “non-exploitability” criterion, Meneely et al. introduces the term “exploitability” to describe the phenomenon where people can manipulate a metric’s measurements without changing the attribute being measured. Contrary to Meneely et al., we consider this criterion during the evaluation of construct validity rather than during the evaluation of internal validity. We do not evaluate the criterion directly. Instead, we consider manipulation of an indicator as one of many threats to construct validity types of criteria.

In addition to the validation criteria already selected, we have also selected the following two criteria from the internal validity category:

- “internal consistency” (definition according to [63]: *“a metric has internal consistency if “all of the elementary measurements of a metric are assessing the same construct and are inter-related”*”) and

- “factor independence” (definition according to [63]: “*a metric has factor independence if the individual measurements used in the metric formulation are independent of each other*”).

The first criterion applies especially to indicators that are composed of basic indicators. If the basic indicators are not conceptually related, then the composite indicators will not have internal consistency and for that reason be hard to interpret. The second criterion also applies especially to indicators that are composed of basic indicators. If the basic indicators use measures that are not independent of each other, then the composite indicator’s ability to measure a risk attribute may be affected. In the category the “factor independence” criterion belongs to, we also find the “causal relationship validity” criterion. According to Meneely et al., a metric has causal relationship validity if “*it has a causal relationship to an external quality factor.*” In our opinion, it would have been more suitable to categorize the criterion as an external validity type of criteria than an internal validity type of criteria, since it concerns the relationship between a metric/indicator and an external quality factor. We have therefore not selected this criterion. It should also be noticed that we have not selected the criterion “metric reliability.” According to Meneely et al., a metric has metric reliability if “*the measurements are accurate and repeatable.*” Even though this criterion is relevant, we have not selected it since it is covered by the “internal consistency” and “attribute validity” criteria, and by construct validity types of criteria.

In the case of construct validity, we consider the following criteria:

- “instrument validity” (definition according to [63]: “*a metric has instrument validity if the underlying measurement instrument is valid and properly calibrated*”);
- “stability” (definition according to [63]: “*a metric has stability if it produces the same values on repeated collections of data under similar circumstances*”); and
- “definition validity” (definition according to [63]: “*a metric has definition validity if the metric definition is clear and unambiguous such that its collection can be implemented in a unique, deterministic way*”).

The “instrument validity” criterion is relevant for evaluating whether the sensors provide correct data to the calculations of the indicators, while the “stability” criterion is relevant for evaluating whether calculations of indicators that involves human decisions result in correct indicators. Moreover, the “definition validity” criterion is relevant since it concerns the implementation of indicators. To implement indicators correctly, their designs must be given in a clear and unambiguous way.

On the other hand, the following criteria from the construct validity category are not considered:

- “protocol validity” (definition according to [63]: “*a metric has protocol validity if it is measured by a widely accepted measurement protocol*”);
- “usability” (definition according to [63]: “*a metric has usability if it can be cost-effectively implemented in a quality assurance program*”); and
- “notation validity” (definition according to [63]: “*a metric has notation validity if the metric is reasoned about mathematically with precise, consistent notation*”).

We do not consider the “protocol validity” criterion since we will in many cases calculate indicators based on measures for which there do not exist widely accepted measurement protocols. The “usability” criterion is also not considered since it is similar to the criterion “economic productivity” discussed earlier. It is more relevant to consider the “usability” criterion when taking a decision on whether to implement the indicator or not. In the case of the “notation validity” criterion, it should be noticed that this criterion is a member of the category “definition validity.” We do find “notation validity” to be a too strict criterion. In many cases we will design indicators that cannot be reasoned about “mathematically with precise, consistent notation.” The “definition validity” criterion is more general, since it can be used to evaluate the design of all kinds of indicators. This completes the evaluation of the success criterion.

Success criterion 3 *The application of the method results in specifications of indicators that are well-suited for ICT-based monitoring.*

It seems fair to argue that the specifications are well-suited for ICT-based monitoring if they contain information that makes it easy for developers to correctly implement the indicators. In our method, as described in Paper A (presented in Chapter 9), each indicator is specified in terms of:

- a deployment specification for the sensors used by the indicator;
- a requirements specification; and
- a design specification.

The deployment specification pinpoints the locations of the sensors. The requirements specification is basically a pre-post specification formally defining the indicator function, while the design specification, in the form of a UML [94] sequence diagram, describes its implementation within the target. The decision of using UML sequence diagrams was not only based on their ability to document the implementation of indicators. It was also based on results from the evaluation of their comprehensibility in the industrial case study described in Paper D (presented in Chapter 12). The results from this evaluation indicate that UML sequence diagrams to a large extent are comprehensible for industrial participants with some knowledge of UML. By being comprehensible, the sequence diagrams serve as an aid in communication between the analysts and the industrial participants.

The design specifications may be used to configure the sensors to extract the data needed in the calculations of indicators. Moreover, since the design specifications describe the main entities (sensors, components, humans) that need to take part in the calculation of indicators and their interactions in the form of data exchanges, they provide a good starting point for implementing monitoring tools, ICT-based work processes, and the infrastructure needed for monitoring the indicators. In addition, with the design specifications given in the form of UML sequence diagrams, developers can detail these diagrams further and achieve specifications that are close to actual implementations. All of this is very much in line with best-practice software engineering.

Success criterion 4 *The method is applicable in an industrial context within acceptable effort.*

Paper A (presented in Chapter 9) demonstrates the method on a large, realistic example case focusing on the use of electronic patient records in a hospital environment. The example case covers in detail all steps of the method. Based on this we may at least claim that the method is applicable in an industrial context given the following two assumptions:

- a) the client is able to provide the required data; and
- b) the analyst is equal to the method designer (and main author of Paper A).

In the following we argue that assumption a) is justified. In Section II-B in Paper A, we have described the different models/descriptions that are developed when the method is applied. An overview of the different models/descriptions is given in Fig. 2 in Paper A. To develop the different models/descriptions, the analyst relies on relevant data to be provided by the client and its co-operating parties. The business objectives for which indicators should be designed are central to all the models/descriptions to be developed. Since these business objectives are central to the success of the client's organization, they are easily obtainable by the client.

The specification "Specification of relevant part of business" documents the actors/systems that are to comply with the business objectives. It also documents how these actors/systems interact by exchanging data. To develop this specification we rely on the client to provide data on relevant actors/systems. If the fulfillment of the business objectives also relies on actors/systems of parties that the client co-operates with, then data on these actors/systems must be provided as well. The specification is to be used to specify the deployment of sensors for monitoring data exchanges between actors/systems. The data needed for developing the specification is available and easily obtainable both within the client's and the co-operating partners' organizations. To develop the specification we only rely on high-level descriptions of the architectures of the client's and the co-operating parties' ICT infrastructures and data on the actors that rely on these infrastructures.

To develop the other eight models/descriptions described in Section II-B in Paper A, we rely on information provided by domain experts that act on behalf of the client. We require domain experts that have knowledge about the business objectives, the relevant part of business and its threat picture, and the ICT infrastructures of the client and the co-operating parties. Knowledge about the latter is needed in order to design indicators for ICT-based monitoring. The client and its co-operating partners are able to provide us with domain experts with the above mentioned knowledge, since the knowledge that we require is essential for the operation of both the client's and the co-operating partners' organizations.

In the following we argue that assumption b) may be weakened into the analyst being an experienced risk analyst. In fact, we claim that most analysts with some experience in risk analysis and a basic understanding of indicators/metrics would be able to do equally well as the method designer. An important part of every risk analysis is to understand the target to be analyzed. An experienced risk analyst is therefore able to conduct Step 1 of the method, since this step is all about understanding the target to be analyzed. Moreover, an experienced risk analyst is able to conduct Steps 2 and 6

of the method, since these steps concern the application of risk analysis. He/she will also have some knowledge about, and quite possibly experience with, risk monitoring. Thus, he/she will have a basic understanding of metrics/indicators. The experienced risk analyst is therefore capable of conducting Step 3 when supported by the domain experts. By interacting with the domain experts, the experienced risk analyst can identify and document sensors to be deployed in the ICT infrastructures of the client and its co-operating partners, as well as specifying requirements to indicators with respect to the identified sensors. The experienced risk analyst is also able to conduct Step 4 when supported by the domain experts. By interacting with the domain experts, the experienced risk analyst can evaluate to what extent the different criteria to internal validity are fulfilled for the different indicators. He/she is also capable of specifying the designs of indicators (Step 5) when supported by the domain experts, since an experienced risk analyst has experience with UML [94] sequence diagrams or similar documentation approaches.

The arguments above show that an analyst only needs a basic understanding of indicators/metrics, since he/she is supported by domain experts during the design of the indicators.

Regarding the effort, consider Table 7.1 estimating the expected effort for the example case in Paper A. We assume that the analyst team consists of two persons; one analysis leader and one analysis secretary. The first leads the meetings with the client, while the second is responsible for documenting the results of the analysis. Moreover, we assume that three domain experts participate on behalf of Client H (the client); one from Client H and one from each of the two co-operating partners Blood test analysis and X-ray. The three domain experts have knowledge about the business objectives, the relevant part of business and its threat picture, and the ICT infrastructures of Client H, Blood test analysis, and X-ray.

In Table 7.1 we have estimated the expected effort in hours for each step and sub-step of the method for the Analyst (total for the two analysts) and the Client (total for the three domain experts). We estimate both time spent in meetings (M) and time spent between and preparing for meetings (P). Both of the analysts participate in all the steps and sub-steps of the method. This is also true for all of the domain experts.

Step 1.1 and Step 1.2 are conducted as 2 and 3 hours meetings, respectively. In Step 1.1, the domain expert from Client H presents the business objective and constraints that should be satisfied in order to fulfill the objective. Based on this presentation, the business objective is expressed more precisely. Since only one business objective is considered in the example case, a 2 hours meeting should be sufficient. For the same reason, a preparation time of 3 hours should be sufficient for giving the presentation. No preparation time is needed for the analysts and the other two domain experts.

In the case of Step 1.2, the domain experts give a presentation of the different actors/systems and their interactions that may be of relevance for describing the relevant part of business. A preparation time of 5 hours in total should be sufficient for the domain experts. The domain experts will also provide the two analysts with documentation before the meeting. A preparation time of 5 hours is reasonable for each of the analysts. Based on the presentation, the participants of the meeting discuss the relevance of the different actors/systems and interactions. The analysts use the results from this discussion and the documentation provided by the domain experts to develop a specification of the relevant part of business. As can be seen in Fig. 4 in Paper A, this development results in a high-level description of how different actors/systems interact.

Table 7.1: Expected effort (in hours) for the Analyst (total for the two analysts) and the Client (total for the three domain experts) with respect to the example case in Paper A. M = “time spent in meetings,” while P = “time spent between and preparing for meetings”

Step	Analyst		Client	
	P	M	P	M
Step 1: Establish target	22	10	11	15
Step 1.1: Express business objectives more precisely	0	4	3	6
Step 1.2: Describe relevant part of business	22	6	8	9
Step 2: Identify risks to fulfillment of business objective	20	18	0	27
Step 2.1: Specify risk acceptance criteria	0	2	0	3
Step 2.2: Risk identification and estimation	20	12	0	18
Step 2.3: Risk evaluation	0	4	0	6
Step 3: Identify key indicators to monitor risks	23	12	10	18
Step 3.1: Deploy sensors to monitor risks	11	6	5	9
Step 3.2: Specify requirements to key indicators wrt deployed sensors	12	6	5	9
Step 4: Evaluate internal validity	3	6	0	9
Step 4.1: Express business objective in terms of key indicators	1	0	0	0
Step 4.2: Evaluate criteria for internal validity	2	6	0	9
Step 5: Specify key indicator designs	24	6	13	9
Step 6: Evaluate construct validity	20	16	0	24
Total (preparations and meetings)	112	68	34	102
	180		136	
	316			
Write the analysis report	40		0	
Total (preparations, meetings, and analysis report)	220		136	
	356			

Based on this, we would say that no more than 10 hours are needed for developing the specification. After the specification has been developed it needs to be checked by the domain experts. Each domain expert should not need more than 1 hour to check the correctness of the specification, while the analysts should not need more than 2 hours to implement changes proposed by the domain experts.

In Step 2.1 we specify the risk acceptance criteria. A 1 hour meeting should be sufficient for specifying the criteria. In Step 2.2 we identify risks towards the fulfillment of the business objective and we estimate their risk values. The results from this step are documented in Figs. 5–9 in Paper A. Even though it does not seem as much work to develop the risk models depicted in Figs. 5–9, we expect that it may take as much as two meetings of 3 hours each and 10 hours spent by the analysts outside the meetings to develop the risk models. Moreover, we believe that the analysts would require 5 hours each to prepare for the meetings. No preparation is necessary for the domain experts in this case. In the case of Step 2.3, we both evaluate risks and accumulate risks that pull in the same direction. A 2 hours meeting should be sufficient for conducting this step. The risk evaluation mainly consists of plotting the risks according to their likelihoods and consequences in a risk matrix given by the risk acceptance criteria, while risks are accumulated by accumulating their likelihood and consequence values.

Steps 3.1 and 3.2 will be conducted in parallel. Fig. 10 in Paper A specifies the deployment of sensors in the relevant part of business. Based on this figure, we would say that it is possible to identify these sensors within a time period of 3 hours. Moreover, 3 hours should also be sufficient for conducting Step 3.2. This means that a 6 hours meeting is sufficient for conducting Steps 3.1 and 3.2. In order to conduct this meeting within 6 hours, the analysts should come up with proposals to sensor deployments and requirements to indicators with respect to the deployed sensors before the meetings. We estimate that each of the analysts would need to spend 10 hours to prepare the proposals. The domain experts should use these proposals in their preparation for the meetings. We expect that the three domain expert use a total of 10 hours to prepare for the meeting. In addition to the time spent in the meeting and for preparations, we estimate that the analysts spend 1 hour in total to develop Fig. 10 in Paper A and 2 hours in total to develop the requirements specifications in Tables V–X after the meeting has been concluded.

Not much effort is required to conduct Step 4.1. The business objective can easily be expressed in terms of indicators by using the results of Steps 2.3 and 3.2. An hour spent by one of the analysts should be sufficient for conducting this step. In the case of Step 4.2, we estimate that a 3 hour meeting should be sufficient. Moreover, an additional 2 hours would be needed by the analysts to document the results of the evaluation of internal validity after the meeting has been concluded.

The results of Step 5 are the indicator design specifications given in Figs. 11–24 in Paper A. To develop these specifications, the analysts should first come up with some proposals to the design of indicators that can be discussed in a meeting. We estimate that 10 hours should be sufficient for coming up with these proposals, while the domain experts would need about 10 hours in total to study these proposals before the meeting. A meeting of 3 hours should be sufficient for discussing the proposals. After the meeting has been conducted, the analysts may need as much as 10 hours to update the specifications. The updated specifications need to be checked by the domain experts. We estimate that a total of 3 hours are used to check their correctness. An additional 4 hours may be needed by the analysts to update the specifications that are

not correct.

The results of Step 6 are documented in Figs. 25–30 and Table XII in Paper A. The expected effort for the analysts and the domain experts in Step 6 should be similar to the expected effort for Steps 2.2 and 2.3, since Step 6 and the two sub-steps concerns similar tasks, and since the outputs from Step 6 and the two sub-steps are similar with respect to size.

After all the meetings have been conducted, the analysts document the analysis and its results in a report. The report will contain the models/descriptions developed during the analysis, and text explaining these models/descriptions. We estimate that 40 hours should be sufficient for writing the report. The domain experts will not spend any time in relation to the report writing.

As can be seen in Table 7.1, the estimated effort for the analysts and the three domain experts for the example case in Paper A is 356 hours, when including the time spent on writing the analysis report. To assess whether this estimated effort is acceptable, we compare the estimated effort of the example case in Paper A to the estimated effort of the industrial case study in Paper E (presented in Chapter 13). The industrial case study is described in Appendix A in Paper E. Even though the two methods reported on in the two papers differ and the industrial case study in Paper E requires more effort than the example case in Paper A, the industrial case study has a number of things in common with the example case:

- Both the industrial case study and the example case focus on the fulfillment of business objectives. In the industrial case study, the client presented us with a business objective that they need to comply with. To comply with this business objective, a number of requirements must be fulfilled. The main purpose of the industrial case study was to identify and assess different risks to the fulfillment of these requirements.
- Both the industrial case study and the example case rely heavily on risk analysis by the use of CORAS and the design of indicators.
- Both the industrial case study and the example case involved two analysts and three domain experts.

Because of the similarities between the industrial case study and the example case, the industrial case study can be used to some extent to assess whether the estimated effort of the example case is acceptable. As can be seen in Table II in Paper E, the estimated effort for the industrial case study was 485 hours. Experiences from other commercial projects of this kind indicate that this is a reasonable effort. For example, CORAS [11] has been developed for analyses handling 150–300 man-hours on behalf of the analyst team alone. The estimate from the industrial case study does not include the time spent by the domain experts between meetings, because we do not have these numbers. It does also not include the time spent on writing the analysis report. Since the effort in the industrial case study was acceptable from the client’s viewpoint and since the example case is smaller than the industrial case study, we find it reasonable that the client in the example case also would find the effort acceptable.

Based on all of the above discussion, we conclude that the method is applicable in an industrial setting within acceptable effort.

7.1.2 Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services

Success criterion 5 *The application of the method results in specifications of indicators that correctly capture and measure the impact of service dependencies on the quality of provided services.*

It seems fair to argue that the ability of the method in Paper B (presented in Chapter 10) to result in specifications of indicators that correctly capture and measure the impact of service dependencies on the quality of provided services may be decomposed into its ability to deliver:

- a) a target model that correctly captures the service dependencies of relevance for the quality of the provided services;
- b) a risk model that correctly captures the impact of service dependencies on risk to the quality of provided services with respect to the target model; and
- c) specifications of indicators that correctly measure to what extent the risks identified for monitoring are acceptable.

Regarding a), which is the output of Step 1, it is of course possible to get the target model wrong. However, we claim that our modeling language has the essential features needed for capturing the service dependencies of relevance for the quality of the provided services. Firstly, our approach to capture relations at the system level is completely general since we use graphs to represent systems (vertices) and services (edges). Moreover, quality requirements to services are captured by annotating edges with required service levels. This is also much in line with other approaches as for example [84–87]. In [84], Holmgren models electric power delivery networks by the use of graphs. The models are used to calculate values of topological characteristics of networks and to compare their error and attack tolerance. In [85], the CIMS framework for infrastructure interdependency modeling and analysis is presented. In this framework, infrastructure entities are modeled by the use of vertices, while edges are used to represent the flow of a physical quantity, information, or influence (e.g., geographical, policy/procedural, etc.). The latter shows that the framework does not only consider dependencies that are the result of physical linkages between entities. In [86], Svendsen presents a multigraph model for critical infrastructures. The model uses vertices to represent infrastructure components that act as consumers and/or providers of different commodities, while edges represent exchanges of commodities. In addition, requirements to the exchanged commodities are specified by the use of response functions and predicates that represent the maximum capacity of an edge and the lower threshold of flow through the edge. In [87], an approach to modeling interdependent infrastructures in the context of vulnerability analysis is presented. Each infrastructure is represented by both a network model, in the form of a graph, and a functional model. The network model shows how the infrastructure’s physical components interact, while the functional model captures physical and operational characteristics of the infrastructure.

Secondly, we also facilitate the capturing of dependencies between services by using logical gates. It could of course be argued that it should be possible to put weights on

the dependencies in order to capture the strengths of the dependencies, and this is a potential generalization of our approach. We have not gone into this issue in this thesis however, since we know from experience (e.g., [147, 148]) how difficult it is to get this kind of estimates.

Thirdly, we support the modeling of trust relations in the case of third-party dependencies. These relations are captured in terms of probabilities, which is very much in line with standard trust definitions [20, 21]. The third-party dependencies involve third-party systems for which we have insufficient information. As previously explained in Section 3.4, the need for reasoning about trust is motivated by the industrial case study described in Paper D (presented in Chapter 12). In this paper, trust is used to decide whether tenders submitted by economic operators for which we have insufficient information should be trusted to be authentic or not.

One issue we have not gone into is the issue of uncertainty. We acknowledge that getting exact estimates regarding probabilities in particular and quantitative measures in general is problematic. However, we claim that our methodologies may be generalized to work with intervals following the recommendations in [152].

Regarding b), which is the output of Step 2, high-level risk models are schematically constructed from the target model by following the schematic procedure specified in Section 3.2.2 in Paper B. By following this procedure, the high-level impact of service dependencies on risk to quality of provided services is documented. The high-level risk models are constructed by the use of the CORAS risk modeling language [11]. This language is part of the CORAS approach, which is based on the ISO 31000 standard [88] which is preceded by the AS/NZS 4360 standard [90]. To achieve a detailed understanding of the impact of service dependencies on risk to quality of provided services and to establish a risk picture that can be monitored, the high-level risk models are detailed by using the CORAS risk modeling language. The language has characteristics that support the construction of correct risk models. It has been developed to facilitate communication and interaction during structured brain-storming sessions involving people of heterogeneous backgrounds [153, 154]. Moreover, the language makes use of graphical symbols that are closely related to the underlying risk concepts, and that are intended to be easily comprehensible.

Regarding c), which is the output of Steps 3 and 4, then our notion of correctness corresponds to internal and construct validity. We have already discussed this in detail in relation to **Success criterion 2**.

Success criterion 6 *The application of the method results in specifications for the deployment and design of indicators that are sufficient for setting up risk monitoring based on indicators.*

The method in Paper B (presented in Chapter 10) identifies indicators for measuring likelihood and consequence values that are used in the monitoring of risk values. It seems fair to argue that the application of the method results in specifications for the deployment and design of indicators that are sufficient for setting up risk monitoring based on indicators if the specifications describe:

- a) how indicators for measuring likelihood/consequence values should be calculated;
- b) how entities (e.g., sensors, humans, etc.) for extracting the data needed in the calculation of indicators should be deployed within the interconnected systems, and

how and where the extracted data should be transmitted; and

- c) how risk values should be monitored based on the indicators and other relevant factors.

Regarding a), the indicators' design specifications describe how the different indicators for measuring likelihood/consequence values should be calculated. Each design specification is given in the form of an algorithm, where the algorithm specifies the data needed for calculating an indicator and how the indicator should be calculated based on the data. For the example case in Paper B, examples of such algorithms are given in Tables 3, 4, 7–9, 12–14, 17, and 18 in Paper B. By implementing the algorithms, the indicators needed for measuring likelihood/consequence values can be calculated.

Regarding b), the indicators' deployment specifications describe how the data needed in the calculation of indicators should be extracted and transmitted within the interconnected systems. More precisely, the specifications describe the data to be extracted, the deployment of entities for extracting the data, when the data should be extracted, and where and how the extracted data should be transmitted for further processing. For the example case in Paper B, deployment specifications are given in Tables 5, 10, 11, 15, 16, 19, and 20 in Paper B. The deployment specifications serve as a good starting point for deployment of sensors, humans, etc., and for setting up the monitoring infrastructure and the ICT-based work processes needed for extracting and transmitting the data needed in the calculation of indicators.

Regarding c), we can describe how the likelihood/consequence values measured by the indicators should be used in the monitoring of risk values. The risk value of a risk is derived from its likelihood and consequence. Based on likelihoods measured by indicators, we can calculate other likelihoods, including the likelihood of the risk. On the other hand, consequences measured by indicators can be used directly in the calculation of risk values. The rules specified in Appendix D.1 in Paper B are used to specify how risk values should be calculated based on indicators and other relevant factors. Such specifications are used by a risk monitor to calculate risk values. Appendices D.2–D.5 document the application of these rules on the example case in Paper B.

Based on the above arguments for a), b), and c), we conclude that the deployment and design specifications are sufficient for setting up risk monitoring based on indicators.

Success criterion 7 *The method is applicable in an industrial context within acceptable effort.*

Paper B (presented in Chapter 10) demonstrates the method on a large, realistic example case within the domain of power supply. Domain knowledge on systems with dependencies was used to develop the example case. This knowledge was obtained in the industrial case study that is described in Paper E (presented in Chapter 13). The example case covers in detail all steps of the method. Based on this we may at least claim that the method is applicable in an industrial context given the following two assumptions:

- a) the client is able to provide the required data; and
- b) the analyst is equal to the method designer (and main author of Paper B).

In the following we argue that assumption a) is justified. In Section 3 in Paper B, we have described the documentation provided by the client during the development of the models/specifications. To develop the target model in Step 1.1, the analyst relies on documentation from the client on the interconnected systems, their service interactions, and the requirements to the different services in the form of required service levels. As can be seen in Figure 10 in Paper B, the different systems and their services are documented at a high level of abstraction in the target model. Requirements to the services may be found in service level agreements or in other relevant documentation. In the target model, it is only necessary to document systems of other parties that the client's systems interact directly with, but other systems in the environment of the client's systems may be documented as well if such information is available. The documentation needed for developing the target model is available and easily obtainable within the client's organization, since only high-level documentation is needed.

To annotate the target model with dependency constructs in Step 1.2, the analyst relies on documentation from the client on how services provided by the client's systems depend on other services. Detailed documentation is not needed, since dependencies are only documented at a high level of abstraction, as can for instance be seen in Figure 11 in Paper B. It is not necessary to document dependencies for systems of parties that operate in the environment of the client's systems, but these dependencies may be documented if the relevant information is available. The documentation needed for annotating the target model with dependency constructs is available and easily obtainable within the client's organization, since only high-level documentation on service dependencies is needed.

To annotate the target model with trust relations in Step 1.3, the analyst relies on trust estimates from the client. These estimates will typically be the result of expert judgments given by domain experts that act on behalf of the client. Each trust estimate is assigned to a required service level of a service provided by an external system to a system of the client. The trust estimate states the degree to which the client trusts the required service level of the service to be delivered. In the client's organization there will be employees that have knowledge about the services provided by external parties. Thus, the knowledge required for specifying the trust estimates is available.

To perform Steps 2-4, we rely on information provided by domain experts that act on behalf of the client. We require domain experts that have knowledge about the relevant systems and services, the dependencies between the different services, the threat picture of the systems and services of the client, and the ICT infrastructure underlying the different systems. Knowledge about the latter is needed in order to design indicators for ICT-based monitoring. The client is able to provide us with domain experts with the above mentioned knowledge, since the knowledge that we require is essential for the operation of the client's systems and services.

In the following we argue that assumption b) may be weakened into the analyst being an experienced risk analyst. In fact, we claim that most analysts with some experience in risk analysis and a basic understanding of service dependencies and indicators/metrics would be able to do equally well as the method designer. An important part of every risk analysis is to understand the target to be analyzed. An experienced risk analyst is therefore able to conduct Step 1 of the method, since this step is all about understanding the target to be analyzed. In addition, an experienced risk analyst will also have a basic understanding of service dependencies, since service dependencies are found in numerous risk analysis targets. The experienced risk analyst is therefore able

to document the service dependencies of the target when supported by the domain experts. Moreover, an experienced risk analyst is able to conduct Step 2 of the method, since this step concerns the application of risk analysis. He/she will also have some knowledge about, and quite possibly experience with, risk monitoring. Thus, he/she will have a basic understanding of metrics/indicators. The experienced risk analyst is therefore capable of conducting Step 3 when supported by the domain experts. By interacting with the domain experts, the experienced risk analyst can identify the risks to be monitored and indicators for monitoring these risks. He/she is also capable of specifying the designs and deployments of indicators (Step 4) when supported by the domain experts, since an experienced risk analyst have experience with documentation approaches suitable for conducting such tasks.

The arguments above show that an analyst only needs a basic understanding of service dependencies and indicators/metrics, since he/she is supported by domain experts during the identification and documentation of service dependencies and during the specification of designs and deployments for indicators.

Regarding the effort, consider Table 7.2 estimating the expected effort for the example case in Paper B. We assume that the analyst team consists of two persons; one analysis leader and one analysis secretary. The first leads the meetings with the client, while the second is responsible for documenting the results of the analysis. Moreover, we assume that three domain experts participate on behalf of Client EPP (the client). The three domain experts have knowledge about the relevant systems and services, the dependencies between the different services, the threat picture of the systems and services of the client, and the ICT infrastructure underlying the different systems.

In Table 7.2 we have estimated the expected effort in hours for each step and sub-step of the method for the Analyst (total for the two analysts) and the Client (total for the three domain experts). We estimate both time spent in meetings (M) and time spent between and preparing for meetings (P). Both of the analysts participate in all the steps and sub-steps of the method. This is also true for all of the domain experts.

Step 1.1 is conducted as two meetings of 3 and 2 hours. In the first meeting, one of the domain experts presents the different systems and services of the electrical power production infrastructure (EPP), the public telecom infrastructure (PTI), and the electrical power grid (EPG) that may be of relevance to the analysis. A preparation time of 5 hours should be sufficient for giving this presentation. The domain expert will also provide the two analysts with relevant documentation before the meeting. A preparation time of 5 hours is also reasonable for each of the analysts. The two other domain experts would not need to prepare for the meeting. Based on the presentation, we identify the different systems and services that should be considered in the analysis. We also decide to capture the impact of service dependencies on risk to the quality of each service that Client EPP provides to systems of the PTI and the EPG. Based on this meeting, the analysts develop an initial target model. The two analysts will use 4 hours in total to develop the initial target model. This initial target model is distributed to the three domain experts. In the second meeting, the domain experts identify errors and shortcomings in the initial target model. We also specify the required service levels for the different services documented in the initial target model based on information provided by the domain experts. The result of this meeting is the target model in Figure 10 in Paper B. Based on the size of the complete target model in Figure 10, we expect that each of the domain experts will need 2 hours each to prepare for the meeting. A preparation time of 2 hours is also reasonable for each of the analysts.

Table 7.2: Expected effort (in hours) for the Analyst (total for the two analysts) and the Client (total for the three domain experts) with respect to the example case in Paper B. M = “time spent in meetings,” while P = “time spent between and preparing for meetings”

Step	Analyst		Client	
	P	M	P	M
Step 1: Document interconnected systems	26	18	17	27
Step 1.1: Model interconnected systems	20	10	11	15
Step 1.2: Capture service dependencies	3	4	3	6
Step 1.3: Capture trust relations	3	4	3	6
Step 2: Analyze the impact of service dependencies on risk to quality of provided services	58	32	0	48
Step 2.1: Identify quality assets	2	0	0	0
Step 2.2: Construct high-level threat diagrams of the impact of service dependencies on identified quality assets	10	0	0	0
Step 2.3: Construct detailed threat diagrams of the impact of service dependencies on identified quality assets	46	32	0	48
Step 3: Identify indicators for interconnected systems	30	16	16	24
Step 3.1: Identify risks to be monitored	0	4	6	6
Step 3.2: Identify relevant indicators for the risks to be monitored	30	12	10	18
Step 4: Specify design and deployment of identified indicators for interconnected systems	25	12	10	18
Step 4.1: Specify design of indicators for risk monitoring	12.5	6	5	9
Step 4.2: Specify deployment of indicators for risk monitoring	12.5	6	5	9
Total (preparations and meetings)	139	78	43	117
	217		160	
	377			
Write the analysis report	40		0	
Total (preparations, meetings, and analysis report)	257		160	
	417			

After the meeting has been conducted, we estimate that the analysts will use 2 hours to update the target model.

We conduct both Step 1.2 and Step 1.3 during a 4 hours meeting. For each step, we spend 2 hours. The target model in Figure 10 is distributed to the domain experts before the meeting. The domain experts are asked to identify service dependencies and to come up with trust estimates. Based on the size of the target model in Figure 10, a preparation time of 2 hours is reasonable for each of the domain experts. A preparation time of 2 hours is also reasonable for each of the analysts. For both the analysts and the domain experts, we divide the preparation time between the two steps. During the meeting, we discuss the findings of the domain experts and we annotate the target model with dependency constructs and trust relations. After the meeting has been conducted, we estimate that the analysts use about 2 hours in total to update the target model based on the results from the meeting. The 2 hours are divided between the two steps.

The two analysts conduct Steps 2.1 and 2.2 without the involvement of the domain experts. In Step 2.1, a quality asset is identified for each of the required service levels of the five provided services for which service dependencies' impact on risk to quality should be captured. We estimate that 2 hours in total are sufficient for identifying and documenting these assets. The results of conducting Step 2.2 are the high-level threat diagrams in Figures 13 and 23–26 in Paper B. Each diagram provides a high-level overview of the impact of service dependencies on risk to quality of a provided service. Moreover, each diagram has been schematically constructed from the target model in Figure 12 by following the procedure described in Section 3.2.2 in Paper B. With the exception of the diagram in Figure 26, it should be straight-forward to schematically construct all of these diagrams from the target model. We estimate that 10 hours, in average 2 hours for each diagram, should be sufficient.

A number of meetings are required for conducting Step 2.3. In the first meeting, the analysts present the results of Steps 2.1 and 2.2 to the domain experts. In this meeting, we also create the likelihood scale in Table 1 in Paper B and the consequence scales in Tables 2 and 6, as well as the risk evaluation criteria in Equations 1 and 2 and in Equations 5–7. We estimate that the first meeting can be conducted within 3 hours. A total preparation time of 2 hours is estimated for the two analysts. The domain experts do not need to prepare for this meeting.

In the other meetings, we detail the high-level threat diagrams in Figures 13 and 23–26 in Paper B. The high-level threat diagrams in Figures 13, 24, and 25 are detailed during the same meeting, since all of these diagrams focus on services provided by the same system. Thus, some of the diagrams resulting from the detailing of these high-level threat diagrams have a lot in common. It should also be noticed that most of the diagrams resulting from the detailing of the high-level threat diagram in Figure 24 also represent a detailing of the high-level threat diagram in Figure 25. To detail the high-level threat diagrams in Figures 13, 24, and 25, we estimate that a six hours meeting should be sufficient. Before the meeting, the analysts create the first version of all the diagrams that should result from the detailing of the high-level threat diagrams. Most of these diagrams will not contain much information. By creating them before the meeting, the time that would otherwise be used for modeling during the meeting can be used for more important tasks. We estimate that the analysts will use 4 hours in total to create the first version of the diagrams in Figures 14–20, 36–42, 45, and 46 in Paper B. After the meeting has been conducted, we estimate that the analysts will

use about 16 hours in total to complete the diagrams that were developed during the meeting.

A meeting of 3 hours should be sufficient for detailing the high-level threat diagram in Figure 23. We estimate that the analysts will use about 2 hours in total to create the first version of the diagrams in Figures 27–33 before the meeting. After the meeting has been conducted, we estimate that the analysts will use about 8 hours in total to complete the diagrams that were developed during the meeting.

For the high-level threat diagram in Figure 26, we estimate that a meeting of 4 hours should be sufficient for detailing it. We estimate that the analysts will use about 2 hours in total to create the first version of the diagrams in Figures 47–52, 57, and 58. The diagrams in Figures 53–56 will be created from scratch during the meeting, since these diagrams are the result of detailing the diagram in Figure 52. After the meeting has been conducted, we estimate that the analysts will use about 12 hours in total to complete the diagrams that were developed during the meeting.

We estimate that Step 3.1 can be conducted during a 2 hours meeting. The domain experts should use some time to prepare before this meeting, in order to make correct decisions regarding the risks that should be monitored. A preparation time of 6 hours in total should be sufficient. Two meetings of 3 hours each are necessary for conducting Step 3.2. The first meeting is used for brain-storming ideas on how to monitor the identified risks by the use of indicators. Based on the ideas from the meeting, the analysts create proposals for indicators. These proposals are distributed to the domain experts. In the second meeting, the proposals are discussed and we identify the indicators that should be used for monitoring the risks. To prepare for the two meetings, we estimate that the domain experts will need 10 hours for preparations in total, while the two analysts will need about 15 hours in total to prepare for the meetings and to create the proposals. The diagrams in Figures 21, 34, 35, 43, 44, and 59 in Paper B will be created during and between these meetings. These diagrams are created based on diagrams developed during Step 2.3. As part of this step, the analysts also need to specify how the risk values of the risks should be monitored based on the identified indicators. For the example case in Paper B, this is documented in Appendices D.2–D.5. We estimate that the analysts need to use 15 hours in total to specify how risk values should be monitored.

Steps 4.1 and 4.2 will be conducted in parallel. Two meetings of 3 hours each are necessary for conducting the two steps. The first meeting is used for brain-storming ideas on how the indicators should be calculated and how they should be deployed within the ICT-infrastructure of Client EPP. Based on the ideas from the meeting, the analysts develop design and deployment specifications for the different indicators. These specifications are distributed to the domain experts. In the second meeting, the specifications are discussed. Based on the discussion during this meeting, the analysts will correct errors and shortcomings in the specifications after the meeting has been concluded. To prepare for the two meetings, we estimate that the domain experts will need 10 hours for preparations in total, while the two analysts will need about 25 hours in total to prepare for the meetings and to create and update the specifications. The specifications in Tables 3–5 and 7–20 in Paper B will be created during and between these meetings. In total, the estimated time spent by the analysts on Steps 4.1 and 4.2 is 37 hours, while the estimated time spent by the domain experts is 28 hours. In Table 7.2 we have divided the estimated time spent by the analysts and the domain experts equally between Steps 4.1 and 4.2.

After all the meetings have been conducted, the analysts document the analysis and its results in a report. The report will contain the different models/specifications developed during the analysis, and text explaining these models/specifications. We estimate that 40 hours should be sufficient for writing the report. The domain experts will not spend any time in relation to the report writing.

As can be seen in Table 7.2, the estimated effort for the analysts and the three domain experts for the example case in Paper B is 417 hours. To assess whether this estimated effort is acceptable, we compare it with the estimated effort of the industrial case study in Paper E in the same way as we did in relation to **Success criterion 4**. The industrial case study and the example case have the following things in common:

- Both the industrial case study and the example case address systems that depend on other systems.
- Both the industrial case study and the example case rely heavily on risk analysis by the use of CORAS and the design of indicators.
- Both the industrial case study and the example case involved two analysts and three domain experts.

By using the same arguments as the ones given in relation to **Success criterion 4**, we come to the conclusion that the estimated effort of the example case in Paper B is acceptable. Based on all of the above discussion, we conclude that the method is applicable in an industrial setting within acceptable effort.

7.1.3 **Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators**

Success criterion 8 *The architectural pattern serves as a basis for building monitoring tools based on indicators within a wide range of domains and enterprises.*

It seems fair to argue that the architectural pattern in Paper C (presented in Chapter 11) serves as a basis for building monitoring tools based on indicators within a wide range of domains and enterprises if:

- a) it is documented in such a way that it is easy to learn, compare, and use;
- b) the capabilities that we consider to be core features of enterprise level monitoring tools are general; and
- c) the architectural pattern captures all of the core features.

Regarding a) it may of course be discussed what classifies as a good description of a pattern, but in general the description must capture the essence of solving the recurring problem in such a way that the pattern is easy to learn, compare, and use. In [23, 24], pattern templates have been used to achieve such descriptions. The architectural pattern is presented in Paper C by the use of the template described in [24]. This template is very similar to the pattern template by Gamma et al. [23], which is one of the best-known pattern templates.

According to the template in [24], a pattern is described by the use of 14 different categories. Table 7.3 provide a summary of these categories. We have not used all of

Table 7.3: Summary of the pattern template in [24]

Category	Short description
Name	The name and a short summary of the pattern.
Also Known As	Other names for the pattern, if applicable.
Example	A real-world example that demonstrates the existence of the problem and the need for the pattern. The example is used throughout the description to illustrate solution and implementation aspects where this is necessary or useful.
Context	The situations in which the pattern may apply.
Problem	The problem that the pattern addresses.
Solution	The fundamental solution principle underlying the pattern.
Structure	A detailed specification of the structural aspects of the pattern.
Dynamics	Typical scenarios describing the run-time behavior of the pattern.
Implementation	Guidelines for implementing the pattern in form of example code. Notice that the guidelines only serve as a suggestion.
Example Resolved	Discussion of any important aspects for resolving the real-world example that have not been covered in the Solution, Structure, Dynamics, and Implementation categories.
Variants	A brief description of variants or specializations of the pattern.
Known Uses	Examples of the pattern in use, taken from existing systems.
Consequences	The benefits that the pattern provide and any potential liabilities.
See Also	References to patterns that solve similar problems and to patterns that help us to refine the pattern we are describing.

these categories, since not all of them are relevant for describing our pattern. In the following we describe how we applied the template in Paper C, and we argue for the exclusion of certain categories.

We cover most of the template categories in Section II (Architectural Pattern) of Paper C. For some categories, however, we found it more suitable to cover them in other sections. For instance, the two categories “Context” and “Problem” are covered as part of Section I (Introduction), while the category “Example” is covered as part of Section III (Demonstration of Architectural Pattern). In the case of the “Example” category, it was not necessary to illustrate different aspects of the pattern during its presentation in Section II. This also means that we found the category “Example Resolved” redundant. Moreover, we also found the categories “Also Known As” and “Variants” redundant, since the pattern is not known by any other names and since there are no other variants or specializations of the pattern. It should also be noticed that we do not explicitly document the “Known Uses” category in the paper. On the other hand, we refer several times to the risk monitor that the architectural pattern is a generalization of. For instance, in Section III we show that the risk monitor is an instance of the pattern.

All of the other categories are covered in Section II. In the case of the categories “Name” and “See Also,” we used the names “Name and short summary” and “Related patterns” instead. We found these names to be more descriptive than the category names given in [24]. In the case of the “Structure” category, Buschmann et al. [24] recommends the use of Class-Responsibility-Collaborator (CRC) cards [155] and OMT class diagrams [156] to describe the structural aspects of the pattern. On the other hand, in case of the “Dynamics” category, the authors of [24] recommend the use of Object Message Sequence Charts¹ to describe the run-time behavior of the pattern. To document relations between the components of the pattern we have used a UML [94] class diagram instead of an OMT class diagram, while CRC cards have been used to document the responsibilities of each component of the pattern. Moreover, we have used UML sequence diagrams instead of Object Message Sequence Charts to document typical scenarios that describes the run-time behavior of the pattern, while in the case of the “Implementation” category, we have used Java for the example code. These modeling/programming languages, and especially UML and Java, are well-known to software developers, and therefore suitable for describing different aspects of the pattern.

Regarding b) as previously explained, the architectural pattern is a generalization of the architecture of the CORAS risk monitor [36] that was developed in the MASTER [37] research project. The risk monitor has a number of features that makes it well suited for dynamic risk monitoring in the context of MASTER [36]. In particular, likelihoods and consequences in the CORAS threat diagrams (the risk picture) can be expressed by indicators that are dynamically updated during run-time. In the MASTER framework, the CORAS risk monitor has been successfully integrated in the part of the framework that provides functionality for monitoring the execution of business and control processes, as well as indicator values and risk levels. The risk monitor exhibits a number of features that are not specific to the monitoring of risks, but general to a broad class of enterprise level monitoring tools. The features are as follows:

¹The authors of [24] own adaption of Message Sequence Charts [157].

1. Collect low-level data from the ICT infrastructure or similar.
2. Aggregate the collected low-level data.
3. Evaluate the aggregated data.
4. Present the aggregated data and the evaluation results to different types of enterprise users.
5. Present the most recent aggregated data and evaluation results.
6. Configure the tool with respect to:
 - a. The low-level data that should be collected.
 - b. How the low-level data should be aggregated into information that is relevant and useful.
 - c. How aggregated data should be evaluated.
 - d. The kind of aggregated data and evaluation results that should be presented and how this should be made comprehensible to different types of enterprise users.

In the following we argue that these features are general. We consider monitoring tools that operate based on an ICT infrastructure or other types of ICT-based systems. When designing such a tool we need to take into consideration the specific characteristics of the infrastructure/system on which the tool is based. The data generated at the ICT infrastructure level is typically of a low-level nature, e.g., events such as service calls or responses. At the enterprise level, an individual event has often little significance when considered in isolation. In order to make sense of the events, they need to be collected and aggregated into high-level information that is both useful and relevant at the enterprise level.

In an enterprise setting, monitoring is typically used for evaluation. An enterprise may want to evaluate whether it complies with requirements from laws and regulations, whether risks are at an acceptable level, or whether it fulfills business objectives. Thus, an enterprise level monitoring tool needs to evaluate the aggregated data. Moreover, the various enterprise users have different information needs and different preferences for how information should be presented. For instance, the chief security officer may want a high-level assessment of the information security risk to which the company is exposed, while a security engineer may want to know how often a port in a firewall is open.

It is almost needless to say, but when users of an enterprise level monitoring tool are presented with updates of aggregated data and evaluation results, they should always be presented with the most recent aggregated data and evaluation results. Moreover, an enterprise level monitoring tool will often experience changes with respect to the collection of low-level data, the aggregation of low-level data, the evaluation of aggregated data, and the presentation of aggregated data and evaluation results during its life-time. Thus, it must be possible to configure the tool.

Regarding c), the architectural pattern divides an enterprise level monitoring tool into three components: MonitorModel, which contains the core monitoring functionality and data; MonitorConsole, which presents aggregated data and evaluation results

in a specific way to a group of users; and MonitorConfig, which is used to set up the enterprise level monitoring tool and to configure it during run-time. If the enterprise level monitoring tool needs to present aggregated data and evaluation results to more than one group of users, then the tool will have more than one MonitorConsole.

The MonitorModel collects relevant low-level data in the form of basic indicators from a component referred to as DataSource in Paper C. This component is part of the pattern's environment. The actual collection of low-level data from the ICT infrastructure or similar, for instance by the use of sensors, is outside the scope of the pattern. The MonitorModel aggregates the basic indicators into composite indicators. Thus, MonitorModel captures features 1 and 2. Moreover, the MonitorConsoles retrieve the most recent updated composite indicators from MonitorModel, evaluate them if needed, and update the displays used by their users based on the composite indicators and evaluation results. Thus, the MonitorConsoles capture features 3, 4, and 5. The MonitorModel and the MonitorConsoles are configured by the MonitorConfig before run-time and during run-time. The MonitorModel is configured with respect to 6a and 6b, while the MonitorConsoles are configured with respect to 6c and 6d. Thus, MonitorConfig captures features 6 a–d.

Based on the above arguments for a), b), and c), we conclude that the architectural pattern can be used as a starting point for building specialized monitoring tools within various kinds of domains and enterprises.

Success criterion 9 *The architectural pattern facilitates modularity and reuse.*

ISO/IEC/IEEE 24765 [17] defines modularity as “*the degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components,*” while it defines reuse as “*building a software system at least partly from existing pieces to perform a new application.*” Moreover, it defines reusable as “*pertaining to a software module or other work product that can be used in more than one computer program or software system.*”

It seems fair to argue that the architectural pattern in Paper C (presented in Chapter 11) facilitates modularity and reuse if:

- changes to a MonitorModel, MonitorConsole, or MonitorConfig has minimal impact on the other components in the enterprise level monitoring tool; and
- the MonitorModel, MonitorConsoles, and MonitorConfig developed for one enterprise level monitoring tool can at least be partly reused in another enterprise level monitoring tool.

Regarding a), our architectural pattern is closely related to the Model-View-Controller (MVC) pattern [24, 120], since this pattern was used as inspiration when we designed our pattern. As previously explained in Section 4.4.3, MVC divides an interactive application into three main components: model, view, and controller. The model encapsulates core data and functionality, while views and controllers together comprise the user interface of the application. The model is independent of the user interface. The same is also true for our pattern. As described in Section II in Paper C, the pattern separates user interfaces (MonitorConsoles) from the component handling core monitoring data and functionality (MonitorModel). Thus, changes to the implementations

of MonitorConsoles will not require changes to the implementation of the MonitorModel. Also, the introduction of new MonitorConsoles during run-time has no effect on the implementation of the MonitorModel.

The MonitorConfig is used to configure the MonitorModel and the MonitorConsoles with a data configuration file and presentation models, respectively. There is a close coupling of MonitorModel to MonitorConfig. The MonitorModel depends on the specific language in which its data configuration file is expressed. Change of language may most likely require changes to the implementation of MonitorModel. Similar, a MonitorConsole may depend on the specific language in which its presentation model is expressed. Changes to this language may require changes to the implementation of the MonitorConsole. However, changes to the language are considered to be quite rare in both cases.

Both the MonitorConsoles and the MonitorConfig make direct calls to the MonitorModel. In addition, the MonitorConfig makes direct calls to the MonitorConsoles. This means that changes to the interface of MonitorModel will break the code of both the MonitorConsoles and the MonitorConfig, while changes to the interfaces of MonitorConsoles will break the code of the MonitorConfig. Such changes to the interfaces will however be quite rare. It should be noticed that MonitorModel does not make direct calls to MonitorConsoles or MonitorConfig. The interaction between MonitorConfig and MonitorModel is one-way only, while it interacts with MonitorConsoles through Observer objects. The latter is due to the use of a change-propagation mechanism, which is implemented by the use of the Publisher-Subscriber design pattern [24].

A MonitorConsole depends on a number of the MonitorModel's composite indicators. A re-configuration of the MonitorModel may result in the removal of composite indicators that one or more MonitorConsoles depend on. If this happens, then the MonitorConfig needs to re-configure the affected MonitorConsoles. If a MonitorConsole is re-configured with a presentation model that is parameterized with new composite indicators, then the MonitorConsole needs to notify the MonitorModel that it wants to receive updates for the new ones.

Regarding b), if the same language is used to create all the data configuration files, then it is possible to implement a generic MonitorModel, which becomes specialized when configured. Two MonitorModels that are based on the same generic MonitorModel will only be different with respect to: the data sources that they retrieve basic indicators from; the basic indicators that they retrieve; the composite indicators that result from the aggregation of their basic indicators; the MonitorConsoles that depend on their composite indicators; and the MonitorConfigs that configure them. In the case of different languages, the majority of the code base of one MonitorModel can be used in the implementation of another.

A MonitorConfig has functionality for creating and updating data configuration files and presentation models, and for configuring MonitorModels and MonitorConsoles. The data configuration files are created and updated by the use of some text-based editor, while each presentation model is created and updated either by the use of a text-based editor or a graphical modeling tool. Both a text-based editor and a graphical modeling tool can often be extended to handle new languages. Thus, it may be possible to use a MonitorConfig in another monitoring tool where support for other languages is required. If it is not possible to extend the MonitorConfig, then at least some of its code base may be reused.

The reuse of a MonitorConsole depends on how tightly integrated its presentation

model is with the rest of the component. If the component only updates the composite indicators that the presentation model is parameterized with, then it may be the case that the presentation model is easily replaceable by another. On the other hand, if the component performs other types of updates of the model or is heavily involved in the evaluation of the composite indicators, then it may be more difficult to reuse the component. If it is not possible to reuse the MonitorConsole, then at least some of its code base may be reused.

Based on the above arguments for a) and b), we conclude that the architectural pattern facilitates modularity and reuse.

7.1.4 Framework for analyzing and monitoring the impact of dependencies on quality

Success criterion 1 *The framework fulfills its intended purpose.*

As explained in Section 2.2.1, the purpose of the framework is to: (1) analyze the impact of service dependencies on quality of services; and (2) support the set-up of monitoring of the impact of service dependencies on quality of services.

In Section 5.1 we have described the integration of the three artifacts. The framework takes as input a business objective that focus on the achievement of service quality. With the relevant part of business being interconnected systems that depend on each other through service interactions, some of the steps of the method of Artifact 1 are replaced by steps of the method of Artifact 2 in order to document the interconnected systems and to capture the impact of service dependencies on the relevant service quality.

Artifact 2 can be used to analyze and capture the impact of service dependencies on risk to the fulfillment of the business objective that focus on the achievement of service quality. The artifact is therefore an answer to (1). By using Artifact 2, we can also identify indicators for monitoring all risks that are unacceptable with respect to the business objective.

Artifact 1 is used to specify deployments of sensors within the interconnected systems, where the sensors are needed for gathering the data necessary for calculating the indicators. It is also used to specify requirements to the indicators with respect to the sensor deployments. By using Artifacts 1 and 2, we can specify how the indicators should be calculated and how the calculations should be embedded within the interconnected systems. Artifact 1 is also used to evaluate the validity of the designed indicators with respect to the relevant service quality.

The output from the integration of Artifacts 1 and 2 is a set of indicators that is valid with respect to the relevant service quality. These indicators can be used to monitor risks to the fulfillment of the business objective. By using Artifact 3, we can implement a risk monitor based on the designed indicators. With respect to (2), Artifact 2 is partly an answer, while Artifacts 1 and 3 are the main answer.

The integration as described in Section 5.1 is feasible. Moreover, the evaluation of the fulfillment of the success criteria for Artifacts 1–3 in Sections 7.1.1–7.1.3 shows that the three artifacts fulfill their intended purposes. Based on the above discussion and the results from the evaluation of Artifacts 1–3, we conclude that the framework also fulfills its intended purpose.

7.2 How the artifacts relate to and extend the state of the art

In this section we discuss how the three artifacts relate to and extend the state of the art presented in Chapter 4. We structure the discussion into three sub-sections; one for each of the artifacts. The reason why such a discussion was not conducted already in Chapter 4 is that the different artifacts were first presented in detail in Chapter 5. For further comparison of our artifacts to the state of the art, the reader is referred to the related work sections of the papers presented in Part II of this thesis.

7.2.1 Artifact 1: Method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators

Section 4.1 presents different approaches related to the use, design and deployment, and validation of indicators. GQM [54,55] and GQ(I)M [56,57] are two approaches that are that are closely related to our method. What makes GQM and GQ(I)M different from our method is that they do not put the same emphasis on the validation of metrics/indicators. For both GQM and GQ(I)M, no specific method, beyond reviews, is specified for validating the metrics/indicators. Our method and the two approaches have in common that all three specify the design and deployment of metrics/indicators. In [55], Solingen and Berghout provide instructions for how to document metrics when using the GQM approach, while in [57], Goethert and Siviý provide a template that can be use to document the construction and use of indicators when using GQ(I)M.

The frameworks presented in [58,59] by Popova and Sharpanskykh are also related to our method. In [58,59], the relationships between goals and performance indicators are made explicit. Moreover, the frameworks use a formal approach to model organizational goals based on performance indicators. The frameworks also use mechanisms for establishing goal satisfaction and for checking consistency of and correspondence between the goal and the performance indicator structures. Popova and Sharpanskykh do not evaluate the validity of the performance indicators. What can be said, however, is that the modeling results in a hierarchical goal structure that can be used during the evaluation of internal validity. Based on [58,59], we have not identified any specifications/models that can be used during the evaluation of construct validity.

Our research on how to validate indicators has been inspired by research on the validation of software engineering metrics. This research has resulted in many criteria for evaluating metrics, where many of them are not specific to software engineering. Based on the systematic literature review of Meneely et al. [63] of papers focusing on validation of software engineering metrics, we selected a number of criteria for evaluating the validity of the indicators that our method designs.

7.2.2 Artifact 2: Method for capturing and monitoring the impact of service dependencies on the quality of provided services

Artifact 2 is a specialization of the approach presented in [105] by Refsdal and Stølen. The approach in [105] is general in the sense that it only restricts the risk identification to the identified assets and nothing else. In our approach, the risk identification focuses entirely on risk to quality of provided services that have been caused by service dependencies. The approach in [105] can of course be used to identify indicators for the purpose of measuring the impact of service dependencies on risk to quality of provided services, because of its generality. Compared to our approach, however, it is inferior. The approach in [105] does not offer any support for dealing with service dependencies. In addition, it focuses to a much lesser extent on the calculations of indicators, and it cannot be used to specify how the indicator calculations should be deployed in the systems to be monitored.

Section 4.2.2 presents approaches for the modeling and analysis of service dependencies. Our graph-based approach to the modeling of service dependencies is similar to other approaches within critical infrastructure protection (see e.g., [84–87]). Our modeling approach also supports the modeling of trust relations in the case of third-party service dependencies. These relations are captured in terms of probabilities. In our method, trust is used to estimate likelihoods of third-party services failing to be delivered according to their requirements. The trust approaches presented in [131–133, 135] can be used to address a similar problem. All these approaches can be used to assess service providers' ability to provide services with the expected quality. The main difference between our approach to trust and the trust approaches presented in [131–133, 135] is that the latter approaches focus on dynamic aspects of trust. These dynamic trust approaches also differ from our approach in that they are primarily used for selecting appropriate interaction partners, for instance in a peer-to-peer network.

Section 4.3.2 presents approaches for analysis of risk in the context of dependencies. In [92, 93], Giese et al. present a method for compositional hazard analysis of components by the use fault trees [95]. For each component, incoming, outgoing, and internal failures are described, as well as the dependencies between the different failures. The dependency information is used to capture the propagation of failures by combining failure information of the different components. This method differs from our method in many respects. First, fault trees cannot be used to address mutual dependencies. Second, the method of Giese et al. is limited to failures caused by software and/or hardware. Thus, human failures, either accidental or deliberate, cannot be addressed. Both mutual dependencies and human failures can be addressed by the use of our method. In [96] and [99], other approaches to component-based hazard analysis are presented. These approaches do also apply fault trees. Thus, they lack the ability to address mutual dependencies.

7.2.3 Artifact 3: Architectural pattern for constructing enterprise level monitoring tools based on indicators

Section 4.4.3 presents state of the art on patterns and the implementation of monitoring tools. The Model-View-Controller (MVC) pattern [24, 120] was used as inspiration when we designed our pattern. Both patterns separate the core data and functional-

ity from the user interfaces. In addition, both patterns use the same mechanism for reflecting changes in the core data in the user interfaces. More precisely, the Publisher-Subscriber design pattern [24] is used to implement this change-propagation mechanism. One of the main differences between our pattern and MVC is that all the user interfaces in MVC display the same core data, while in our pattern the MonitorConsoles can display different core monitoring data, which means that they need to be updated differently. Another difference is that we have replaced the controller component in MVC with the MonitorConfig component. In MVC, each controller is used for handling user input for a view. The MonitorConfig component on the other hand is used for configuring the MonitorModel and the MonitorConsoles before and during run-time by the use of configuration files.

Approaches like [121] and [123] differ from our architectural pattern in that they address specific monitoring problems within specific domains. Our architectural pattern, on the other hand, may be used to build monitoring tools within a wide range of domains and enterprises. The pattern presented in [121] do also differ from our pattern in that it focus on the sensors collecting the information needed in the monitoring. The construction of the sensor infrastructure is outside the scope of our pattern.

The tool framework called Mozart [122] uses a model driven approach to create monitoring applications that uses key performance indicators (KPIs). Mozart differs from our pattern in that it focuses both on the aggregation of already existing indicators and on the transformation of these indicators into a monitoring application. This means that Mozart both design and deploy indicators. Our pattern, on the other hand, is only concerned with the deployment of indicators through the implementation of a monitoring tool.

Chapter 8

Conclusion

This chapter concludes Part I of the thesis by summarizing the achievements and by outlining directions for future work.

8.1 What has been achieved

The main contributions of this thesis are three new artifacts

1. a method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators;
2. a method for capturing and monitoring the impact of service dependencies on the quality of provided services; and
3. an architectural pattern for constructing enterprise level monitoring tools based on indicators.

These artifacts may be integrated into a framework for analyzing and monitoring the impact of service dependencies on quality of services. We have argued that the framework is

1. well-suited to analyze the impact of service dependencies on quality of services;
2. well-suited to support the set-up of monitoring of the impact of service dependencies on quality of services; and
3. applicable in an industrial context within acceptable effort.

Artifact 2 can be used to analyze the impact of service dependencies on quality of services. The artifact is therefore an answer to part 1 of the overall objective (presented above). We are not aware of any other approaches that have the same capabilities as Artifact 2.

In the framework, Artifacts 1 and 2 are used to design indicators for monitoring the impact of service dependencies on quality of services. Artifact 1 is also used evaluate the validity of the designed indicators with respect to the relevant service quality. Moreover, Artifact 3 can be used to implement a monitoring tool that deploys the indicators designed by Artifacts 1 and 2. Artifact 2 is partly an answer to part 2 of the overall objective, while Artifacts 1 and 3 are the main answer. We are not aware

of any other approaches that have the same capabilities as Artifact 1. The same may also be said about Artifact 3.

The evaluation of the success criteria in Section 7.1 shows that Artifacts 1 and 2 are applicable in an industrial context within acceptable effort. Based on this, Artifacts 1 and 2 are the answer to part 3 of the overall objective.

This thesis also contributes in terms of

1. an empirical study on trust-based decisions in interconnected systems; and
2. an empirical study on the design of indicators for monitoring risk.

These industrial case studies were mainly carried out to support the development of the artifacts, but since they also provide insight into issues of a more general nature, they may be seen as contributions on their own.

8.2 Directions for future work

There are a number of directions for future work. An obvious direction for future work is to apply the framework and the artifacts in an industrial setting to gather more empirical evidence on their applicability and to assess their performance in a practical setting.

As already mentioned during the evaluation of Success criterion 5 in Section 7.1.2, the method in Paper B (presented in Chapter 10) does not take into account the strengths of service dependencies. A potential generalization of our method is therefore to put weights on the service dependencies in order to capture their strengths. By taking this into account we will get a more correct risk picture of the impact of service dependencies on the quality of provided services. Moreover, one way to improve the correctness of estimates, e.g., likelihood estimates, is to measure the same thing in different ways. This is especially relevant when it comes to monitoring by the use indicators. We can for instance use several independent indicators to monitor the same likelihood value. The possibilities and challenges with respect to using several independent indicators are something that we want to investigate further.

Another direction for future work is to provide tool support for Step 1 and Step 2.2 of the method in Paper B. Step 1 concerns the creation of the target model, i.e., the modeling of interconnected systems, services, service dependencies, etc., while Step 2.2 concerns the schematic construction of high-level risk models of the impact of service dependencies on risk to the quality of provided services from the target model. A tool should be able to construct these high-level risk models automatically from the target model by following the schematic procedure specified in Section 3.2.2 in Paper B.

An interesting direction for future work is to combine the framework with approaches from the critical infrastructure domain that have simulation capabilities. By using an approach such as [86] we can for instance run simulations on a graph-based model of interconnected systems and investigate how the functionality of systems and services changes when nodes are removed. In this way, the most critical systems of the graph can be identified. The framework can then be applied for these systems to analyze and monitor the impact of service dependencies on the quality of the critical services they provide.

Another interesting direction for future work is the use of a dynamic trust approach for reasoning about third-party service dependencies. In the method in Paper B, trust

is used to estimate likelihoods of third-party services failing to be delivered according to their requirements. Dynamic trust approaches (e.g., [131–133, 135]) can be used to capture the change in trust over time. By using a dynamic trust approach we can update the initial trust estimates based on past behavior of third-party services and other factors. The likelihoods estimated based on trust estimates can be expressed by the use of indicators. The likelihoods can then be monitored, and changes in trust can be reflected in the risk picture.

Bibliography

- [1] I. Solheim and K. Stølen, “Technology Research Explained,” Tech. Rep. SINTEF A313, SINTEF, 2007.
- [2] A. Refsdal, *Specifying Computer Systems with Probabilistic Sequence Diagrams*. PhD thesis, Faculty of Mathematics and Natural Sciences, University of Oslo, 2008.
- [3] C. Stoll, “An Epidemiology of Viruses & Network Worms,” in *Proceedings of 12th National Computer Security Conference*, pp. 369–377, 1989.
- [4] R. A. Kemmerer, “Cybersecurity,” in *Proceedings of the 25th International Conference on Software Engineering*, pp. 705–715, IEEE Computer Society, 2003.
- [5] H. Sundmaecker, P. Guillemin, P. Friess, and S. Woelfflé, eds., *Vision and Challenges for Realising the Internet of Things*. European Commission, 2010.
- [6] R. Roman, P. Najera, and J. Lopez, “Securing the Internet of Things,” *IEEE Computer*, vol. 44, no. 9, pp. 51–58, 2011.
- [7] United States Nuclear Regulatory Commission, “NRC Information Notice 2003-14: Potential Vulnerability of Plant Computer Network to Worm Infection,” August 29, 2003.
- [8] U.S.-Canada Power System Outage Task Force, “Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations,” 2004.
- [9] C. J. Alberts and J. Davey, “OCTAVE Criteria Version 2.0,” Tech. Rep. CMU/SEI-2001-TR-016, Carnegie Mellon University, 2004.
- [10] B. Barber and J. Davey, “The use of the CCTA Risk Analysis and Management Methodology CRAMM in Health Information Systems,” in *Proceedings of 7th International Congress on Medical Informatics (MEDINFO’92)*, pp. 1589–1593, 1992.
- [11] M. S. Lund, B. Solhaug, and K. Stølen, *Model-Driven Risk Analysis: The CORAS Approach*. Springer, 1st ed., 2010.
- [12] P. Pederson, D. Dudenhoeffer, S. Hartley, and M. Permann, “Critical Infrastructure Interdependency Modeling: A Survey of U.S. and International Research,” Tech. Rep. INL/EXT-06-11464, Idaho National Laboratory, 2006.

- [13] A. A. Ghorbani and E. Bagheri, “The State of the Art in Critical Infrastructure Protection: A Framework for Convergence,” *International Journal of Critical Infrastructures*, vol. 4, no. 3, pp. 215–244, 2008.
- [14] J. M. Yusta, G. J. Correa, and R. Lacal-Aránategui, “Methodologies and Applications for Critical Infrastructure Protection: State-of-the-Art,” *Energy Policy*, vol. 39, no. 10, pp. 6100–6119, 2011.
- [15] American Heritage Dictionary Editors, *The American Heritage Dictionary of the English Language*. Houghton Mifflin, 2006.
- [16] Institute of Electrical and Electronics Engineers, “IEEE Std 829 – IEEE Standard for Software and System Test Documentation,” 2008.
- [17] International Organization for Standardization, International Electrotechnical Organization, and Institute of Electrical and Electronics Engineers, “ISO/IEC/IEEE 24765 Systems and Software Engineering – Vocabulary,” 2010.
- [18] A. Hammond, A. Adriaanse, E. Rodenburg, D. Bryant, and R. Woodward, *Environmental Indicators: A Systematic Approach to Measuring and Reporting on Environmental Policy Performance in the Context of Sustainable Development*. World Resources Institute, 1995.
- [19] B. Ragland, “Measure, Metrics or Indicator: What’s the Difference?,” *Crosstalk: The Journal of Defense Software Engineering*, vol. 8, no. 3, 1995.
- [20] D. Gambetta, “Can We Trust Trust?,” in *Trust: Making and Breaking Cooperative Relations* (D. Gambetta, ed.), pp. 213–237, Basil Blackwell, 1988.
- [21] A. Jøsang, C. Keser, and T. Dimitrakos, “Can We Manage Trust?,” in *Proceedings of the Third International Conference on Trust Management (iTrust’05)*, pp. 93–107, Springer, 2005.
- [22] T. Lysemose, T. Mahler, B. Solhaug, J. Bing, D. Elgesem, and K. Stølen, “ENFORCE Conceptual Framework,” Tech. Rep. SINTEF A1209, SINTEF, 2007.
- [23] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1st ed., 1994.
- [24] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture, Volume 1: A System of Patterns*. Wiley, 1st ed., 1996.
- [25] J. Hartmanis, “Some Observations About the Nature of Computer Science,” in *Proceedings of 13th Conference on the Foundations of Software Technology and Theoretical Computer Science (FSTTCS’93)* (R. K. Shyamasundar, ed.), vol. 761 of *Lecture Notes in Computer Science*, pp. 1–12, Springer, 1993.
- [26] G. Brændeland, *Component-based Risk Analysis*. PhD thesis, Faculty of Mathematics and Natural Sciences, University of Oslo, 2011.
- [27] P. J. Denning, “Is Computer Science Science?,” *Communications of the ACM*, vol. 48, no. 4, pp. 27–31, 2005.

-
- [28] F. B. Brooks, *The Mythical Man-Month: Essays on Software Engineering*. Addison-Wesley, 20th anniversary ed., 1995.
- [29] F. B. Brooks, "The Computer Scientist as Toolsmith II," *Communications of the ACM*, vol. 39, no. 3, pp. 61–68, 1996.
- [30] H. Abelson and G. J. Sussman, *Structure and Interpretation of Computer Programs*. MIT Press, 2nd ed., 1996.
- [31] J. Knowles, *Theory of Science: A Short Introduction*. Tapir Akademisk Forlag, 2006.
- [32] W. F. Tichy, "Should Computer Scientists Experiment More?," *IEEE Computer*, vol. 31, no. 5, pp. 32–40, 1998.
- [33] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.
- [34] H. A. Simon, *The Sciences of the Artificial*. MIT Press, 3rd ed., 1996.
- [35] J. E. McGrath, *Groups: Interaction and Performance*. Prentice Hall, 1984.
- [36] MASTER, "D1.1.5: Risk Analysis Modelling (3)," 2011. D1.1.5 Deliverable.
- [37] "MASTER: Managing Assurance, Security and Trust for sERvices." <http://www.master-fp7.eu/>. Accessed: 2012-07-17.
- [38] A. Neely, J. Mills, K. Platts, H. Richards, M. Gregory, M. Bourne, and M. Kennerley, "Performance Measurement System Design: Developing and Testing a Process-based Approach," *International Journal of Operation & Production Management*, vol. 20, no. 10, pp. 1119–1145, 2000.
- [39] T. Lester, "Measure for Measure." <http://www.ft.com/cms/s/2/31e6b750-16e9-11d9-a89a-00000e2511c8.html#axzz1ImHJOLmg>, 5. October 2004. Accessed: 2012-07-17.
- [40] R. S. Kaplan and D. P. Norton, "The Balanced Scorecard – Measures That Drive Performance," *Harvard Business Review*, vol. 70, no. 1, pp. 71–79, 1992.
- [41] A. Neely and M. Bourne, "Why Measurement Initiatives Fail," *Measuring Business Excellence*, vol. 4, no. 4, pp. 3–6, 2000.
- [42] IT Governance Institute, "COBIT 4.1," 2007.
- [43] W. Jansen, *Directions in Security Metrics Research*. DIANE Publishing, 2010.
- [44] IT Governance Institute, "The Val IT Framework 2.0," 2008.
- [45] M. Iqbal and M. Nieves, *ITIL V3 Service Strategy*. The Stationary Office, 2007.
- [46] V. Lloyd and C. Rudd, *ITIL V3 Service Design*. The Stationary Office, 2007.
- [47] S. Lacy and I. Macfarlane, *ITIL V3 Service Transition*. The Stationary Office, 2007.

- [48] D. Cannon and D. Wheeldon, *ITIL V3 Service Operation*. The Stationary Office, 2007.
- [49] G. Spalding and G. Case, *ITIL V3 Continual Service Improvement*. The Stationary Office, 2007.
- [50] International Organization for Standardization and International Electrotechnical Organization, “ISO/IEC 27004 Information Technology – Security Techniques – Information Security Management – Measurement,” 2009.
- [51] International Organization for Standardization and International Electrotechnical Organization, “ISO/IEC 27001 Information Technology – Security Techniques – Information Security Management Systems - Requirements,” 2005.
- [52] International Organization for Standardization and International Electrotechnical Organization, “ISO/IEC 27002 Information Technology – Security Techniques – Code of Practice for Information Security Management,” 2005.
- [53] E. Chew, M. Swanson, K. Stine, N. Bartol, A. Brown, and W. Robinson, “Performance Measurement Guide for Information Security,” Tech. Rep. NIST Special Publication 800-55 Revision 1, National Institute of Standards and Technology, 2008.
- [54] V. R. Basili and D. M. Weiss, “A Methodology for Collecting Valid Software Engineering Data,” *IEEE Transactions on Software Engineering*, vol. SE-10, no. 6, pp. 728–738, 1984.
- [55] R. V. Solingen and E. Berghout, *The Goal/Question/Metric method: A Practical Guide for Quality Improvement of Software Development*. McGraw-Hill International, 1999.
- [56] R. E. Park, W. B. Goethert, and W. A. Florae, *Goal-driven Software Measurement: A Guidebook (Handbook CMU/SEI-96-HB-002)*. Carnegie Mellon University, 1996.
- [57] W. B. Goethert and J. Siviyy, “Applications of the Indicator Template for Measurement and Analysis,” Tech. Rep. CMU/SEI-2004-TN-024, Carnegie Mellon University, 2004.
- [58] V. Popova and A. Sharpanykh, “Modeling Organizational Performance Indicators,” *Information Systems*, vol. 35, no. 4, pp. 505–527, 2010.
- [59] V. Popova and A. Sharpanykh, “Formal Modeling of Organizational Goals Based on Performance Indicators,” *Data & Knowledge Engineering*, vol. 70, no. 4, pp. 335–364, 2011.
- [60] B. Kitchenham, S. L. Pfleeger, and N. Fenton, “Towards a Framework for Software Measurement Validation,” *IEEE Transactions on Software Engineering*, vol. 21, no. 12, pp. 929–944, 1995.
- [61] N. Schneidewind, “Methodology for Validating Software Metrics,” *IEEE Transactions on Software Engineering*, vol. 18, no. 5, pp. 410–422, 1992.

-
- [62] N. Fenton and B. Kitchenham, “Validating Software Measures,” *Journal of Software Testing, Verification and Reliability*, vol. 1, no. 2, pp. 27–42, 1990.
- [63] A. Meneely, B. Smith, and L. Williams, “Software Metrics Validation Criteria: A Systematic Literature Review,” Tech. Rep. TR-2010-2, North Carolina State University, 2010.
- [64] D. H. Krantz, R. D. Luce, P. Suppes, and A. Tversky, *Foundations of Measurement, Vol. I: Additive and Polynomial Representations*. Academic Press, 1971.
- [65] P. Suppes, D. H. Krantz, R. D. Luce, and A. Tversky, *Foundations of Measurement, Vol. II: Geometrical, Threshold, and Probabilistic Representations*. Academic Press, 1989.
- [66] R. D. Luce, D. H. Krantz, P. Suppes, and A. Tversky, *Foundations of Measurement, Vol. III: Representation, Axiomatization, and Invariance*. Academic Press, 1990.
- [67] A. L. Baker, J. M. Bieman, N. E. Fenton, D. A. Gustafson, A. Melton, and R. W. Whitty, “A Philosophy for Software Measurement,” *Journal of Systems and Software*, vol. 12, no. 3, pp. 277–281, 1990.
- [68] L. Briand, K. El-Emam, and S. Morasca, “On the Application of Measurement Theory in Software Engineering,” *Empirical Software Engineering*, vol. 1, no. 1, pp. 61–88, 1996.
- [69] C. Kaner and W. P. Bond, “Software Engineering Metrics: What Do They Measure and How Do We Know,” in *Proceedings of 10th International Software Metrics Symposium (METRICS’04)*, IEEE Computer Society, 2004.
- [70] A. Morali and R. Wieringa, “Towards Validating Risk Indicators Based on Measurement Theory,” in *Proceedings of First International Workshop on Risk and Trust in Extended Enterprises*, pp. 443–447, IEEE Computer Society, 2010.
- [71] J. F. Allen and A. M. Frisch, “What’s in a Semantic Network?,” in *Proceedings of the 20th Annual Meeting on Association for Computational Linguistics (ACL’82)*, pp. 19–27, Association for Computational Linguistics, 1982.
- [72] L. Cox, H. S. Delugach, and D. Skipper, “Dependency Analysis Using Conceptual Graphs,” in *Proceedings of the 9th International Conference on Conceptual Structures*, Springer, 2001.
- [73] J. F. Sowa, “Conceptual Graphs for a Data Base Interface,” *IBM Journal of Research and Development*, vol. 20, no. 4, pp. 336–357, 1976.
- [74] International Electrotechnical Organization, “IEC 61078 Analysis Techniques for Dependability – Reliability Block Diagram,” 1991.
- [75] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*. Wiley, 2nd ed., 2004.

- [76] C. Ensel and A. Keller, “An Approach for Managing Service Dependencies with XML and the Resource Description Framework,” *Journal of Network and Systems Management*, vol. 10, no. 2, pp. 147–170, 2002.
- [77] M. Randic, B. Blaskovic, and P. Knezevic, “Modeling Service Dependencies in Ad Hoc Collaborative Systems,” in *Proceedings of EUROCON 2005*, pp. 1842–1845, IEEE Computer Society, 2005.
- [78] A. Keller and G. Kar, “Dynamic Dependencies in Application Service Management,” in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA’00)* (H. R. Arabnia, ed.), CSREA Press, 2000.
- [79] H. Ding and L. Sha, “Dependency Algebra: A Tool for Designing Robust Real-Time Systems,” in *Proceedings of the 26th IEEE Real-Time Systems Symposium (RTSS’05)*, pp. 210–220, IEEE Computer Society, 2005.
- [80] H. Debar, N. Kheir, N. Cuppens-Boulahia, and F. Cuppens, “Service Dependencies in Information Systems Security,” in *Proceedings of the 5th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security (MMM-ACNS’10)*, pp. 1–20, Springer, 2010.
- [81] F. Baiardi, S. Stuin, C. Telmon, and M. Pioli, “Assessing the Risk of an Information Infrastructure Through Security Dependencies,” in *Critical Information Infrastructures Security, First International Workshop (CRITIS’06)* (J. Lopez, ed.), vol. 4347 of *Lecture Notes in Computer Science*, pp. 42–54, Springer, 2006.
- [82] B. Gruschke, “Integrated Event Management: Event Correlation Using Dependency Graphs,” in *Proceedings of Ninth Annual IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM’98)*, IEEE, 1998.
- [83] A. Rugina, K. Kanoun, and M. Kaâniche, “A System Dependability Modeling Framework Using AADL and GSPNs,” in *Architecting Dependable Systems IV* (R. de Lemos, C. Gacek, and A. Romanovsky, eds.), pp. 14–38, Springer, 2007.
- [84] Å. J. Holmgren, “Using Graph Models to Analyze the Vulnerability of Electric Power Networks,” *Risk Analysis*, vol. 26, no. 4, pp. 955–969, 2006.
- [85] D. D. Dudenhofer, M. R. Permann, and M. Manic, “CIMS: A Framework for Infrastructure Interdependency Modeling and Analysis,” in *Proceedings of the 38th Conference on Winter Simulation* (L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.), pp. 478–485, Winter Simulation Conference, 2006.
- [86] N. K. Svendsen, *Interdependencies in Critical Infrastructures – A Qualitative Approach to Model Physical, Logical, and Geographical Interdependencies*. PhD thesis, Faculty of Mathematics and Natural Sciences, University of Oslo, 2008.
- [87] J. Johansson and H. Hassel, “An Approach for Modelling Interdependent Infrastructures in the Context of Vulnerability Analysis,” *Reliability Engineering & System Safety*, vol. 95, no. 12, pp. 1335–1344, 2010.

-
- [88] International Organization for Standardization, “ISO 31000 Risk Management – Principles and Guidelines,” 2009.
- [89] F. Swiderski and W. Snyder, *Threat Modeling*. Microsoft Press, 2004.
- [90] Standards Australia/Standards New Zealand, “AS/NZS 4360 Risk Management,” 2004.
- [91] International Electrotechnical Organization, “IEC 61165 Application of Markov Techniques,” 1995.
- [92] H. Giese, M. Tichy, and D. Schilling, “Compositional Hazard Analysis of UML Component and Deployment Models,” in *Proceedings of 23rd International Conference on Computer Safety, Reliability and Security (SAFECOMP’04)* (M. Heisel, P. Liggesmeyer, and S. Wittmann, eds.), vol. 3219 of *Lecture Notes in Computer Science*, pp. 166–179, Springer, 2004.
- [93] H. Giese and M. Tichy, “Component-based Hazard Analysis: Optimal Designs, Product Lines, and Online-reconfiguration,” in *Proceedings of 25th International Conference on Computer Safety, Reliability and Security (SAFECOMP’06)* (J. Górski, ed.), vol. 4166 of *Lecture Notes in Computer Science*, pp. 156–169, Springer, 2006.
- [94] Object Management Group, “Unified Modeling Language Specification, Version 2.0,” 2004.
- [95] International Electrotechnical Organization, “IEC 61025 Fault Tree Analysis (FTA),” 1990.
- [96] Y. Papadopoulos, J. McDermid, R. Sasse, and G. Heiner, “Analysis and Synthesis of the Behaviour of Complex Programmable Electronic Systems in Conditions of Failure,” *Reliability Engineering & System Safety*, vol. 71, no. 3, pp. 229–247, 2001.
- [97] Society of Automotive Engineers Inc, “Aerospace Recommended Practice (ARP) 4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment,” 1996.
- [98] A. Bouti and A. D. Kadi, “A State-of-the-Art Review of FMEA/FMECA,” *International Journal of Reliability, Quality and Safety Engineering*, vol. 1, no. 4, pp. 515–543, 1994.
- [99] B. Kaiser, P. Liggesmeyer, and O. Mäkel, “A New Component Concept for Fault Trees,” in *Proceedings of 8th Australian Workshop on Safety Critical Systems and Software (SCS’03)*, pp. 37–46, Australian Computer Society, 2003.
- [100] H. Otway and D. von Winterfeldt, “Expert Judgment in Risk Analysis and Management: Process, Context, and Pitfalls,” *Risk Analysis*, vol. 12, no. 1, pp. 83–93, 1992.
- [101] N. C. Dalkey, “Delphi,” Tech. Rep. P-3704, Rand Corporation, 1967.

- [102] A. L. Delbecq, A. H. V. de Ven, and D. H. Gustafson, *Group Techniques for Program Planning: A Guide to Nominal Group and Delphi Processes*. Green Briar Press, 1986.
- [103] S. Kaplan, “Expert Information vs Expert Opinions: Another Approach to the Problem of Eliciting/Combining/Using Expert Knowledge in PRA,” *Reliability Engineering & System Safety*, vol. 35, no. 1, pp. 61–72, 1992.
- [104] M. Stone, “The Opinion Pool,” *Annals of Mathematical Statistics*, vol. 32, no. 4, pp. 1339–1342, 1961.
- [105] A. Refsdal and K. Stølen, “Employing Key Indicators to Provide a Dynamic Risk Picture with a Notion of Confidence,” in *Proceedings of Third IFIP WG 11.11 International Conference (IFIPTM’09)*, pp. 215–233, Springer, 2009.
- [106] W. H. Baker, L. P. Rees, and P. S. Tippet, “Necessary Measures: Metric-driven Information Security Risk Assessment and Decision Making,” *Communications of the ACM*, vol. 50, no. 10, pp. 101–106, 2007.
- [107] J. Breier and L. Hudec, “Risk Analysis Supported by Information Security Metrics,” in *Proceeding of the 12th International Conference on Computer Systems and Technologies (CompSysTech’11)*, pp. 393–398, ACM, 2011.
- [108] R. Jurca, B. Faltings, and W. Binder, “Reliable QoS Monitoring Based on Client Feedback,” in *Proceedings of the 16th International Conference on World Wide Web (WWW’07)*, pp. 1003–1012, ACM, 2007.
- [109] G. Wang, C. Wang, A. Chen, H. Wang, C. Fung, S. Uczekaj, Y.-L. Chen, W. Guthmiller, and J. Lee, “Service Level Management Using QoS Monitoring, Diagnostics, and Adaptation for Networked Enterprise Systems,” in *Proceedings of the Ninth IEEE International EDOC Enterprise Computing Conference*, pp. 239–248, IEEE Computer Society, 2005.
- [110] Y. Fu, “Data Mining,” *IEEE Potentials*, vol. 16, no. 4, pp. 18–20, 1997.
- [111] W. M. P. van der Aalst, B. F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A. J. M. M. Weijters, “Workflow Mining: A Survey of Issues and Approaches,” *Data Knowledge Engineering*, vol. 47, no. 2, pp. 237–267, 2003.
- [112] M. N. Frolick and T. Ariyachandra, “Business Performance Management: One Truth,” *Information Systems Management*, vol. 23, no. 1, pp. 41–48, 2006.
- [113] A. Trad, D. Kalpic, and K. Fertalj, “Proactive Monitoring of the Information System Risk and Quality,” in *Proceedings of the 24th International Conference on Information Technology Interfaces (ITI’02)*, pp. 279–284, IEEE Computer Society, 2002.
- [114] B. Crispo, G. Gheorghe, V. D. Giacomo, and D. Presentza, “MASTER as a Security Management Tool for Policy Compliance,” in *Towards a Service-Based Internet* (E. D. Nitto and R. Yahyapour, eds.), vol. 6481 of *Lecture Notes in Computer Science*, pp. 213–214, Springer, 2010.

-
- [115] K. Dempsey, N. S. Chawla, A. Johnson, R. Jonhston, A. C. Jones, A. Oregaugh, M. Scholl, and K. Stine, "Information Security Continuous Monitoring (ICSM) for Federal Information Systems and Organizations," Tech. Rep. NIST Special Publication 800-137, National Institute of Standards and Technology, 2011.
- [116] J. Aubert, T. Schaberreiter, C. Incoul, D. Khadraoui, and B. Gâteau, "Risk-Based Methodology for Real-Time Security Monitoring of Interdependent Services in Critical Infrastructures," in *Proceedings of International Conference on Availability, Reliability, and Security (ARES'10)*, pp. 262–267, IEEE Computer Society, 2010.
- [117] F. Caldeira, T. Schaberreiter, E. Monteiro, J. Aubert, P. Simões, and D. Khadraoui, "Trust Based Interdependency Weighting for On-line Risk Monitoring in Interdependent Critical Infrastructures," in *Proceedings of the 6th International Conference on Risk and Security of Internet and Systems (CRiSIS'11)*, IEEE Computer Society, 2011.
- [118] K. Haslum and A. Årnes, "Multisensor Real-time Risk Assessment Using Continuous-time Hidden Markov Models," in *Proceedings of International Conference on Computational Intelligence and Security*, pp. 1536–1540, IEEE Computer Society, 2006.
- [119] X. Tan, Y. Zhang, X. Cui, and H. Xi, "Using Hidden Markov Models to Evaluate the Real-time Risks of Network," in *Proceedings of IEEE International Symposium on Knowledge Acquisition and Modeling Workshop*, pp. 490–493, IEEE Computer Society, 2008.
- [120] G. E. Krasner and S. T. Pope, "A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80," *Journal of Object-Oriented Programming*, vol. 1, no. 3, pp. 26–49, 1988.
- [121] A. Lau and R. Seviora, "Design Patterns for Software Health Monitoring," in *Proceedings of the 10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS'05)*, pp. 467–476, IEEE Computer Society, 2005.
- [122] M. Abe, J. Jeng, and Y. Li, "A Tool Framework for KPI Application Development," in *Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE'07)*, pp. 22–29, IEEE Computer Society, 2007.
- [123] X. Wang, X. Liu, H. Xie, Z. Du, and L. Jin, "Design and Implementation of a Performance Monitoring Tool for Clustered Streaming Media Server Systems," in *Proceedings of International Symposium on Information Science and Engineering (ISISE'10)*, pp. 314–318, IEEE Computer Society, 2010.
- [124] "SolarWinds ipMonitor." <http://www.solarwinds.com/products/ipmonitor/>. Accessed: 2012-07-17.
- [125] "IBM Tivoli Monitoring Software." <http://www-01.ibm.com/software/tivoli/products/monitor/>. Accessed: 2012-07-17.

- [126] F. Caldeira, E. Monteiro, and P. Simões, “Trust and Reputation Management for Critical Infrastructure Protection,” *International Journal of Electronic Security and Digital Forensics*, vol. 3, no. 3, pp. 187–203, 2010.
- [127] A. Jøsang, “A Logic for Uncertain Probabilities,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 3, pp. 279–311, 2001.
- [128] A. Jøsang, “Probabilistic Logic Under Uncertainty,” in *Proceedings of the Thirteenth Australasian Symposium on Theory of Computing (CATS’07)*, pp. 101–110, Australian Computer Society, 2007.
- [129] A. Jøsang, D. Bradley, and S. J. Knapskog., “Belief-based Risk Analysis,” in *Proceedings of the Second Workshop on Australasian Information Security, Data Mining and Web Intelligence, and Software Internationalisation (ACSW Frontiers’04)*, pp. 63–68, Australian Computer Society, 2004.
- [130] E. Charniak, “Bayesian Networks Without Fears: Making Bayesian Networks More Accessible to the Probabilistically Unsophisticated,” *AI Magazine*, vol. 12, no. 4, pp. 50–63, 1991.
- [131] G. Wu, J. Wei, X. Qiao, and L. Li, “A Bayesian Network Based QoS Assessment Model for Web Services,” in *Proceedings of IEEE International Conference on Services Computing (SCC’07)*, pp. 498–505, IEEE Computer Society, 2007.
- [132] D. Melaye and Y. Demazeau, “Bayesian Dynamic Trust Model,” in *Multi-Agent Systems and Applications IV* (M. Pechoucek, P. Petta, and L. Varga, eds.), vol. 3690 of *Lecture Notes in Computer Science*, pp. 480–489, Springer, 2005.
- [133] Y. Wang and J. Vassileva, “Bayesian Network Trust Model in Peer-to-Peer Networks,” in *Agents and Peer-to-Peer Computing* (G. Moro, C. Sartori, and M. P. Singh, eds.), vol. 2872 of *Lecture Notes in Computer Science*, pp. 249–279, Springer, 2005.
- [134] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall, 1st ed., 1995.
- [135] N. Griffiths, K.-M. Chao, and M. Younas, “Fuzzy Trust for Peer-to-Peer Systems,” in *Proceedings of 26th IEEE International Conference on Distributed Computing Systems (ICDCS’06)*, pp. 73–78, IEEE Computer Society, 2006.
- [136] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwavara, “Reputation Systems,” *Communications of the ACM*, vol. 43, no. 12, pp. 45–48, 2000.
- [137] P. Resnick and R. Zeckhauser, “Trust Among Strangers in Internet Transactions: Empirical Analysis of eBay’s Reputation System,” in *The Economics of the Internet and E-Commerce, Volume 11 of Advances in Applied Microeconomics* (M. R. Baye, ed.), pp. 127–157, Elsevier Science, 2002.
- [138] K. Lim, C. Sia, M. Lee, and I. Benbasat, “Do I Trust You Online, and If So, Will I Buy? An Empirical Study of Two Trust-Building Strategies,” *Journal of Management Information Systems*, vol. 23, no. 2, pp. 233–266, 2006.

-
- [139] D. Artz and Y. Gil, “A Survey of Trust in Computer Science and the Semantic Web,” *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 2, pp. 58–71, 2007.
- [140] J. Sabater and C. Sierra, “Review on Computational Trust and Reputation Models,” *Artificial Intelligence Review*, vol. 24, no. 1, pp. 33–60, 2005.
- [141] S. Grabner-Kräuter and E. A. Kaluscha., “Empirical Research in On-line Trust: A Review and Critical Assessment,” *International Journal of Human-Computer Studies - Special Issue: Trust and Technology*, vol. 58, no. 6, pp. 783–812, 2003.
- [142] A. Jøsang, R. Ismail, and C. Boyd, “A Survey of Trust and Reputation Systems for Online Service Provision,” *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.
- [143] O. S. Ligaarden, A. Refsdal, and K. Stølen, “Designing Indicators to Monitor the Fulfillment of Business Objectives with Particular Focus on Quality and ICT-supported Monitoring of Indicators,” *International Journal on Advances in Intelligent Systems*, vol. 5, no. 1–2, pp. 175–193, 2012.
- [144] O. S. Ligaarden, A. Refsdal, and K. Stølen, “ValidKI: A Method for Designing Key Indicators to Monitor the Fulfillment of Business Objectives,” in *Proceedings of the First International Conference on Business Intelligence and Technology (BUSTECH’11)*, pp. 57–65, IARIA, 2011.
- [145] O. S. Ligaarden, A. Refsdal, and K. Stølen, “Using Indicators to Monitor Security Risk in Systems of Systems: How to Capture and Measure the Impact of Service Dependencies on the Security of Provided Services,” in *IT Security Governance Innovations: Theory and Research* (D. Mellado, L. E. Sánchez, E. Fernández-Medina, and M. Piattini, eds.), pp. 256–292, IGI Global, 2012.
- [146] O. S. Ligaarden, M. S. Lund, A. Refsdal, F. Seehusen, and K. Stølen, “An Architectural Pattern for Enterprise Level Monitoring Tools,” in *Proceedings of the 2011 IEEE International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA’11)*, IEEE Computer Society, 2011.
- [147] T. V. Håvaldsrud, O. S. Ligaarden, P. Myrseth, A. Refsdal, K. Stølen, and J. Ølnes, “Experiences from Using a UML-based Method for Trust Analysis in an Industrial Project on Electronic Procurement,” *Journal of Electronic Commerce Research*, vol. 10, no. 3–4, pp. 441–467, 2010.
- [148] O. S. Ligaarden, A. Refsdal, and K. Stølen, “Experiences from Using Indicators to Validate Expert Judgments in Security Risk Analysis,” in *Proceedings of the Third International Workshop on Security Measurements and Metrics (MetriSec’11)*, IEEE Computer Society, 2011.
- [149] N. E. Fenton, “Software Metrics: Theory, Tools and Validation,” *Software Engineering Journal*, vol. 5, no. 1, pp. 65–78, 1990.
- [150] B. Curtis, “Measurement and Experimentation in Software Engineering,” *Proceedings of the IEEE*, vol. 68, no. 9, pp. 1144–1157, 1980.

- [151] N. E. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*. International Thomson Computer Press, 2nd ed., 1996.
- [152] A. Omerovic, *PREDIQT: A Method for Model-based Prediction of Impacts of Architectural Design Changes on System Quality*. PhD thesis, Faculty of Mathematics and Natural Sciences, University of Oslo, 2012.
- [153] I. Hogganvik and K. Stølen, “Risk Analysis Terminology for IT-systems: Does it Match Intuition?,” in *Proceedings of 4th International Symposium on Empirical Software Engineering (ISESE’05)*, pp. 13–23, IEEE Computer Society, 2005.
- [154] I. Hogganvik and K. Stølen, “A Graphical Approach to Risk Identification, Motivated by Empirical Investigations,” in *Proceedings of 9th International Conference on Model Driven Engineering Languages and Systems (MoDELS’06)*, vol. 4199 of *Lecture Notes in Computer Science*, pp. 574–588, Springer, 2006.
- [155] K. Beck and W. Cunningham, “A Laboratory for Teaching Object-Oriented Thinking,” in *Proceedings of Object-oriented programming systems, languages and applications (OOPSLA ’89)*, pp. 1–6, ACM, 1989.
- [156] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, *Object-Oriented Modeling and Design*. Prentice Hall, 1991.
- [157] International Telecommunication Union, “ITU-T Recommendation Z.120 – Message Sequence Chart (MSC),” 2004.

Part II
Research Papers

Chapter 9

Paper A: ValidKI: A method for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators

Report

ValidKI: A Method for Designing Indicators to Monitor the Fulfillment of Business Objectives with Particular Focus on Quality and ICT-supported Monitoring of Indicators

Author(s)

Olav Skjelkvåle Ligearden, Atle Refsdal, and Ketil Stølen

SINTEF IKT
SINTEF ICT

Address:
Postboks 124 Blindern
NO-0314 Oslo
NORWAY

Telephone: +47 73593000
Telefax: +47 22067350

postmottak.ikt@sintef.no
www.sintef.no
Enterprise /VAT No:
NO 948 007 029 MVA

Report

ValidKI: A Method for Designing Indicators to Monitor the Fulfillment of Business Objectives with Particular Focus on Quality and ICT-supported Monitoring of Indicators

KEYWORDS:

Indicator,
Key indicator,
Business objective,
Quality,
ICT-supported monitoring,
Electronic patient record

VERSION

Final version

DATE

2012-10-01

AUTHOR(S)

Olav Skjelkvåle Ligaarden, Atle Refsdal, and Ketil Stølen

CLIENT(S)

Research Council of Norway

CLIENT'S REF.

180052/S10

PROJECT NO.

90B245

NUMBER OF PAGES/APPENDICES:

48/0

ABSTRACT

In this report we present our method ValidKI for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators. A set of indicators is valid with respect to a business objective if it measures the degree to which the business or relevant part thereof fulfills the business objective. ValidKI consists of six main steps. We demonstrate the method on an example case focusing on the use of electronic patient records in a hospital environment.

PREPARED BY

Olav Skjelkvåle Ligaarden


SIGNATURE



CHECKED BY

Fredrik Seehusen

SIGNATURE



APPROVED BY

Bjørn Skjellaug, Research Director

SIGNATURE



REPORT NO.

SINTEF A23413

ISBN

978-82-14-05579-5

CLASSIFICATION

Unrestricted

CLASSIFICATION THIS PAGE

Unrestricted

CONTENTS

I	Introduction	4
II	Basic terminology and definitions	5
II-A	The artifacts addressed by ValidKI	5
II-B	The models/descriptions developed by ValidKI	6
II-C	Validity	6
III	Overview of ValidKI	7
III-A	Establish target	7
III-B	Identify risks to fulfillment of business objective	7
III-C	Identify key indicators to monitor risks	8
III-D	Evaluate internal validity	8
III-E	Specify key indicator designs	9
III-F	Evaluate construct validity	9
IV	Establish target	9
IV-A	Express business objectives more precisely (Step 1.1 of ValidKI)	10
IV-B	Describe relevant part of business (Step 1.2 of ValidKI)	10
V	Identify risks to fulfillment of business objective	13
V-A	Specify risk acceptance criteria (Step 2.1 of ValidKI)	13
V-B	Risk identification and estimation (Step 2.2 of ValidKI)	13
V-C	Risk evaluation (Step 2.3 of ValidKI)	16
VI	Identify key indicators to monitor risks	19
VI-A	Deploy sensors to monitor risks (Step 3.1 of ValidKI)	19
VI-B	Specify requirements to key indicators wrt deployed sensors (Step 3.2 of ValidKI)	21
VII	Evaluate internal validity	23
VII-A	Express business objective in terms of key indicators (Step 4.1 of ValidKI)	23
VII-B	Evaluate criteria for internal validity (Step 4.2 of ValidKI)	24
VIII	Specify key indicator designs	25
VIII-A	Key indicator designs for $K_{PR-SP-EPR-INFO}$ and its basic key indicators	25
VIII-B	Key indicator designs for $K_{PR-HSP-EPR-INFO}$ and its basic key indicators	26
VIII-C	Key indicator designs for $K_{NOT-APP-UNAUTH-ACC}$ and its basic key indicators	28
VIII-D	Key indicator designs for $K_{SP-EPR-INFO}$ and its basic key indicators	28
VIII-E	Key indicator designs for $K_{HSP-EPR-INFO}$ and its basic key indicators	31
VIII-F	Key indicator designs for $K_{ILL-ACC-SC}$ and its basic key indicators	31
IX	Evaluate construct validity	38
X	Related work	44
XI	Conclusion	45
	References	45

ValidKI: A Method for Designing Indicators to Monitor the Fulfillment of Business Objectives with Particular Focus on Quality and ICT-supported Monitoring of Indicators

Olav Skjelkvåle Ligaarden^{*†}, Atle Refsdal^{*}, and Ketil Stølen^{*†}

^{*} Department for Networked Systems and Services, SINTEF ICT
PO Box 124 Blindern, N-0314 Oslo, Norway

E-mail: {olav.ligaarden, atle.refsdal, ketil.stolen}@sintef.no

[†] Department of Informatics, University of Oslo
PO Box 1080 Blindern, N-0316 Oslo, Norway

Abstract

In this report we present our method ValidKI for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators. A set of indicators is valid with respect to a business objective if it measures the degree to which the business or relevant part thereof fulfills the business objective. ValidKI consists of six main steps. We demonstrate the method on an example case focusing on the use of electronic patient records in a hospital environment.

Keywords

Indicator, key indicator, business objective, quality, ICT-supported monitoring, electronic patient record

I. INTRODUCTION

Today's companies benefit greatly from ICT-supported business processes, as well as business intelligence and business process intelligence applications monitoring and analyzing different aspects of a business and its processes. The output from these applications may be indicators which summarize large amounts of data into single numbers. Indicators can be used to evaluate how successful a company is with respect to specific business objectives. For this to be possible it is important that the indicators are valid. A set of indicators is valid with respect to a business objective if it measures the degree to which the business or relevant part thereof fulfills the business objective. Valid indicators facilitate decision making, while invalid indicators may lead to bad business decisions, which again may greatly harm the company.

In today's business environment, companies cooperate across company borders. Such co-operations often result in sharing or outsourcing of ICT-supported business processes. One example is the interconnected electronic patient record (EPR) infrastructure. The common goal for this infrastructure is the exchange of EPRs facilitating the treatment of the same patient at more than one hospital. In such an infrastructure, it is important to monitor the use of EPRs in order to detect and avoid misuse. This may be achieved through the use of indicators. It may be challenging to identify and compute good indicators that are valid with respect to business objectives that focus on quality in general and security in particular. Furthermore, in an infrastructure or system stretching across many companies we often have different degrees of visibility into how the cooperating parties perform their part of the business relationship, making the calculation of indicators particularly hard.

In [1] we presented the method *ValidKI* (Valid Key Indicators) for designing indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators. ValidKI facilitates the design of a set of indicators that is valid with respect to a business objective. In this report we present an improved version of the method.

We demonstrate ValidKI by applying it on an example case targeting the use of EPRs. We have developed ValidKI with the aim of fulfilling the following characteristics:

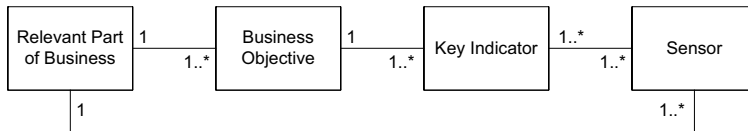


Fig. 1. The artifacts addressed by ValidKI

- **Business focus:** The method should facilitate the design and assessment of indicators for the purpose of measuring the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of indicators.
- **Efficiency:** The method should be time and resource efficient.
- **Generality:** The method should be able to support the design and assessment of indicators based on data from systems that are controlled and operated by different companies or organizations.
- **Heterogeneity:** The method should not place restrictions on how indicators are designed.

The rest of the report is structured as follows: in Section II we introduce our basic terminology and definitions. In Section III we give an overview of ValidKI and its six main steps. In Sections IV – IX we demonstrate our six-step method on an example case addressing the use of EPRs in a hospital environment. In Section X we present related work, while in Section XI we conclude by characterizing our contribution and discussing the suitability of our method.

II. BASIC TERMINOLOGY AND DEFINITIONS

Hammond et al. defines indicator as “*something that provides a clue to a matter of larger significance or makes perceptible a trend or phenomenon that is not immediately detectable*” [2]. For example, a drop in barometric pressure may signal a coming storm, while an unexpected rise in the traffic load of a web server may signal a denial of service attack in progress. Thus, the significance of an indicator extends beyond what is actually measured to a larger phenomenon of interest.

Indicators are closely related to metrics. ISO/IEC/IEEE 24765 [3] defines metric as “*a quantitative measure of the degree to which a system, component, or process possesses a given attribute,*” while it defines attribute as “*the specific characteristic of the entity being measured.*” For the web server mentioned above, an example of an attribute may be availability. An availability metric may again act as an indicator for denial of service attacks, if we compare the metric with a baseline or expected result [4]. As we can see, metrics are not that different from indicators. For that reason, indicators and metrics are often used interchangeably in the literature.

Many companies profit considerably from the use of indicators [5] resulting from business process intelligence applications that monitor and analyze different aspects of a business and its processes. Indicators can be used to measure to what degree a company fulfills its business objectives and we then speak of key indicators. Some business objectives may focus on business performance, while others may focus on risk or compliance with laws and regulations. We will in the remainder of the report refer to indicators as key indicators, since we focus on indicators in the context of business objectives.

A. The artifacts addressed by ValidKI

The UML [6] class diagram in Fig. 1 relates the main artifacts addressed by ValidKI. The associations between the different concepts have cardinalities that specify how many instances of one concept that may be associated to an instance of the other concept.

As characterized by the diagram, one or more key indicators are used to measure to what extent a business objective is fulfilled with respect to a relevant part of the business. Each key indicator is calculated based on data provided by one or more sensors. The sensors gather data from the relevant part of the business. A sensor may gather data for more than one key indicator.

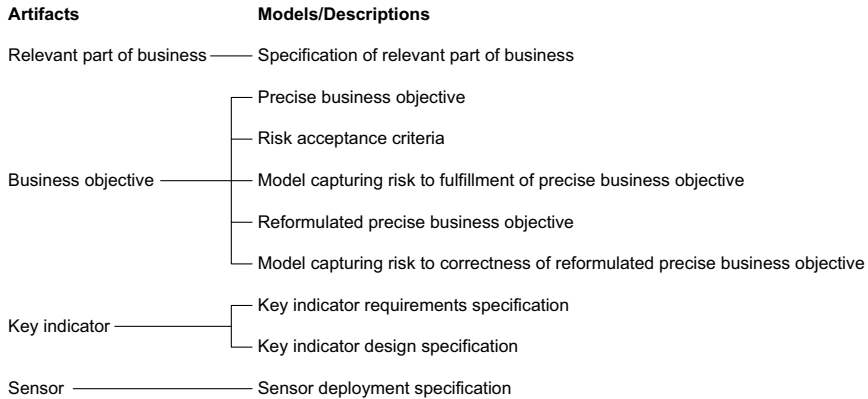


Fig. 2. The models/descriptions developed by ValidKI

B. The models/descriptions developed by ValidKI

As illustrated by Fig. 2, performing the steps of ValidKI results in nine different models/descriptions each of which describes one of the artifacts of Fig. 1 from a certain perspective.

A specification, at a suitable level of abstraction, documents the relevant part of the business in question.

Business objectives are typically expressed at an enterprise level and in such a way that they can easily be understood by for example shareholders, board members, partners, etc. It is therefore often not completely clear what it means to fulfill them. This motivates the need to capture each business objective more precisely.

The fulfillment of a precise business objective may be affected by a number of risks. We therefore conduct a risk analysis to capture risk to the fulfillment of the precise business objective. To evaluate which risks that are acceptable and not acceptable with respect to the fulfillment of the precise business objective, we use risk acceptance criteria. It is the risks that are not acceptable that we need to monitor. The acceptable risks may be thought of to represent uncertainty we can live with. In other words, their potential occurrences are not seen to significantly influence the fulfillment of the business objective.

The degree of fulfillment of a precise business objective is measured by a set of key indicators. To measure its degree of fulfillment there is a need to express each precise business objective in terms of key indicators. We refer to this reformulation as the reformulated precise business objective. Moreover, the correctness of key indicators will be affected if they are not implemented correctly. This may again lead to new unacceptable risks that affect the fulfillment of the precise business objective. Since the reformulated precise business objective is the precise business objective expressed in terms of key indicators, we need to analyze risks to the correctness of the reformulated precise business objective.

The computation of key indicators relies on different kinds of data. To collect the data, sensors need to be deployed in the relevant part of business. Thus, there is a need to specify the deployment of different sensors.

For each key indicator we distinguish between two specifications: the key indicator requirements specification and the key indicator design specification. The first captures requirements to a key indicator with respect to the sensor deployment specifications, while the second defines how the key indicator should be calculated.

C. Validity

ISO/IEC 9126 defines validation as “*confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled*” [7]. Since an indicator is basically a metric that can be compared to a baseline/expected result, the field of metric validation is highly relevant. There is however no agreement upon what constitutes a valid metric [8]. In [8], Meneely et al. present a systematic literature review of papers focusing on validation of software engineering metrics. The literature review began with 2288 papers, which were later reduced to 20 papers. From these 20 papers, the authors extracted and categorized 47 unique validation

Input:	A business objective
Step 1: Establish target	
Step 1.1:	Express business objectives more precisely
Step 1.2:	Describe relevant part of business
Step 2: Identify risks to fulfillment of business objective	
Step 2.1:	Specify risk acceptance criteria
Step 2.2:	Risk identification and estimation
Step 2.3:	Risk evaluation
Step 3: Identify key indicators to monitor risks	
Step 3.1:	Deploy sensors to monitor risks
Step 3.2:	Specify requirements to key indicators wrt deployed sensors
Step 4: Evaluate internal validity	
Step 4.1:	Express business objective in terms of key indicators
Step 4.2:	Evaluate criteria for internal validity
Step 5: Specify key indicator designs	
Step 6: Evaluate construct validity	
Output:	A set of key indicators and a report arguing its validity with respect to the business objective received as input

Fig. 3. Overview of ValidKI

criteria. The authors argue that metric researchers and developers should select criteria based on the intended usage of the metric. Even though the focus in [8] is on validation of software engineering metrics, a number of the validation criteria presented are general, thus not specific to software engineering. In particular, following [8] we define a set of key indicators to be valid with respect to a business objective if it is valid in the following two ways:

- 1) **internal validity** – the precise business objective expressed in terms of the key indicators correctly measures the degree to which the business objective is fulfilled; and
- 2) **construct validity** – the gathering of the sensor measurements of each key indicator is suitable with respect to its requirements specification.

III. OVERVIEW OF VALIDKI

Fig. 3 provides an overview of the ValidKI method. It takes as input a business objective and delivers a set of key indicators and a report arguing its validity with respect to the business objective received as input. When using ValidKI in practice we will typically develop key indicators for a set of business objectives, and not just one which we restrict our attention to here. It should be noticed that when developing key indicators for a set of business objectives, we need to take into account that key indicators (i.e., software or infrastructure) developed for one business objective may affect the validity of key indicators developed for another.

In the following we offer additional explanations for each of the six main steps of the ValidKI method.

A. Establish target

The first main step of ValidKI is all about understanding the target, i.e., understanding exactly what the business objective means and acquiring the necessary understanding of the relevant part of business for which the business objective has been formulated. We distinguish between two sub-steps. In the first sub-step we characterize the business objective more precisely by formulating constraints that need to be fulfilled. In the second sub-step we specify the relevant part of the business.

B. Identify risks to fulfillment of business objective

The second main step of ValidKI is concerned with conducting a risk analysis to identify risks to the fulfillment of the business objective. We distinguish between three sub-steps. In the first sub-step the risk acceptance criteria are

specified. The criteria classify a risk as either acceptable or unacceptable based on its likelihood and consequence. In the second sub-step we identify how threats may initiate risks. We also identify vulnerabilities and threat scenarios leading up to the risks, and we estimate likelihood and consequence. During the risk analysis we may identify risks that pull in the same direction. Such risks should be combined into one risk. The individual risks may be acceptable when considered in isolation, while the combined risk may be unacceptable. In the third sub-step we evaluate the identified risks with respect to the specified risk acceptance criteria.

C. Identify key indicators to monitor risks

The third main step of ValidKI is concerned with identifying key indicators to monitor the unacceptable risks identified in the previous step. We distinguish between two sub-steps. In the first sub-step we specify how sensors should be deployed in the relevant part of business. The key indicators that we identify are to be calculated based on data gathered by the sensors. In the second sub-step we specify our requirements to the key indicators with respect to the deployed sensors. The two sub-steps are typically conducted in parallel.

D. Evaluate internal validity

The fourth main step of ValidKI is concerned with evaluating whether the set of key indicators is internally valid with respect to the business objective. We distinguish between two sub-steps. In the first sub-step we reformulate the precise business objective by expressing it in terms of the identified key indicators. This step serves as an introductory step in the evaluation of internal validity. In the second sub-step we evaluate whether the set of key indicators is internally valid by showing that the reformulated precise business objective from Step 4.1 correctly measures the fulfillment of the precise business objective from Step 1.1.

Internal validity may be decomposed into a broad category of criteria [8]. In the following we list the criteria that we take into consideration. For each criterion, we first provide the definition as given in [8], before we list the papers on which the definition is based.

- **Attribute validity:** “A metric has attribute validity if the measurements correctly exhibit the attribute that the metric is intending to measure” [9][10]. In our case, the key indicator needs to correctly exhibit the risk attribute (likelihood or consequence) of the risk that it is measuring. In addition, the key indicator is of little value if it can only produce values that always result in the risk being acceptable or unacceptable.
- **Factor independence:** “A metric has factor independence if the individual measurements used in the metric formulation are independent of each other” [11]. This criterion applies especially to composite key indicators that are composed of basic key indicators. A composite key indicator has factor independence if the basic key indicators are independent of each other, i.e., if they do not rely on the same measurements.
- **Internal consistency:** “A metric has internal consistency if “all of the elementary measurements of a metric are assessing the same construct and are inter-related”” [12]. This criterion also applies especially to composite key indicators that are composed of basic key indicators. If the basic key indicators measure things that are not conceptually related, then the composite key indicator will not have internal consistency. For instance, let us say that we have a composite key indicator that is composed of two basic key indicators. The first basic key indicator measures the code complexity of a software product, while the second measures the cost of shipping the software product to the customers. In this case, the composite key indicator does not have internal consistency, since the two basic key indicators are not conceptually related.
- **Appropriate continuity:** “A metric has appropriate continuity if the metric is defined (or undefined) for all values according to the attribute being measured” [10]. An example of a discontinuity is fraction calculations when the denominator is zero. To avoid discontinuity, the key indicator should be defined for that case.
- **Dimensional consistency:** “A metric has dimensional consistency if the formulation of multiple metrics into a composite metric is performed by a scientifically well-understood mathematical function” [10][13]. Under dimensional consistency, no information should be lost during the construction of composite key indicators. Loss of information may be experienced if different scales are used for the basic and composite key indicators.
- **Unit validity:** “A metric has unit validity if the units used are an appropriate means of measuring the attribute” [10][14]. For instance, the unit fault rate may be used to measure the attribute program correctness [10].

If the set is not internally valid, then we iterate by re-doing Step 3.

E. Specify key indicator designs

In the fifth main step of ValidKI we specify the designs of the identified key indicators. Each design specifies how the key indicator should be calculated. The design also shows how sensors, actors, and different components interact.

F. Evaluate construct validity

In the sixth main step of ValidKI we evaluate whether the set of key indicators has construct validity with respect to the business objective. As with internal validity, construct validity may be decomposed into a broad category of criteria [8]. In the following we list the criteria that we take into consideration. For each criterion, we first provide the definition as given in [8], before we list the papers on which the definition is based.

- **Stability:** “A metric has stability if it produces the same values “on repeated collections of data under similar circumstances”” [12][15][16]. A key indicator whose calculation involves decisions made by humans, may for example result in different values and thus lack of stability.
- **Instrument validity:** “A metric has instrument validity if the underlying measurement instrument is valid and properly calibrated” [10]. In our case, this criterion concerns the sensors that perform the measurements that the key indicator calculations rely on.
- **Definition validity:** “A metric has definition validity if the metric definition is clear and unambiguous such that its collection can be implemented in a unique, deterministic way” [11][15][16][17][18]. This criterion concerns the implementation of the key indicators. To implement a key indicator correctly, the key indicator’s design specification needs to be clear and unambiguous.

To evaluate the different criteria, we re-do the risk analysis from Step 2.2 with the precise business objective replaced by the reformulated precise business objective, which is the precise business objective expressed in terms of key indicators. For each key indicator we identify risks towards the correctness of the reformulated precise business objective that are the result of threats to criteria for construct validity that the key indicator needs to fulfill. If the risk analysis does not result in any new unacceptable risks, then we have established construct validity for each key indicator. If the set does not have construct validity, then we iterate. We will most likely be re-doing Step 5, but it may also be the case that we need to come up with new key indicators and new sensors. In that case, we re-do Step 3. If the set of key indicators is both internally valid and has construct validity with respect to the business objective, then we have established that the set is valid.

IV. ESTABLISH TARGET

In the following we assume that we have been hired to help the public hospital Client H design key indicators to monitor their compliance with Article 8 in the European Convention on Human Rights [19]. The article states the following:

Article 8 – Right to respect for private and family life

- 1) Everyone has the right to respect for his private and family life, his home and his correspondence.
- 2) There shall be no interference by a public authority with the exercise of this right except such as is in accordance with the law and is necessary in a democratic society in the interests of national security, public safety or the economic well-being of the country, for the prevention of disorder or crime, for the protection of health or morals, or for the protection of the rights and freedoms of others.

Client H needs to comply with Article 8 since it is a public authority. The consequence for Client H of not complying with Article 8 may be economic loss and damaged reputation. One example [20] of violation of Article 8 is from Finland. A Finnish woman was first treated for HIV at a hospital, before she later started working there as a nurse. While working there she suspected that her co-workers had unlawfully gained access to her medical data. She brought the case to the European Court of Human Rights in Strasbourg which unanimously held that the district health authority responsible for the hospital had violated Article 8 by not protecting the medical data of the woman properly. The district health authority was held liable to pay damages to the woman. Client H has therefore established the following business objective:

Business objective BO-A8: Client H complies with Article 8 in the European Convention on Human Rights.

Client H wants to make use of key indicators to monitor the degree of fulfillment of BO-A8, and now they have hired us to use ValidKI to design them. In the rest of this section we conduct Step 1 of ValidKI on behalf of Client H with respect to BO-A8.

A. Express business objectives more precisely (Step 1.1 of ValidKI)

Article 8 states under which circumstances a public authority can interfere with someone’s right to privacy. One of these circumstances is “*for the protection of health,*” which is what Client H wants us to focus on. In the context of Client H this means to provide medical assistance to patients. The ones who provide this assistance are the health-care professionals of Client H.

The medical history of a patient is regarded as both sensitive and private. At Client H, the medical history of a patient is stored in an electronic patient record (EPR). An EPR is “*an electronically managed and stored collection or collocation of recorded/registered information on a patient in connection with medical assistance*” [21]. The main purpose of an EPR is to communicate information between health-care professionals that provide medical care to a patient. To protect the privacy of its patients, Client H restricts the use of EPRs. In order to comply with Article 8, Client H allows a health-care professional to interfere with the privacy of a patient only when providing medical assistance to this patient. Hence, the dealing with EPRs within the realms of Client H is essential.

For Client H it is important that every access to information in an EPR is in accordance with Article 8. A health-care professional should only access a patient’s EPR if he/she provides medical assistance to that patient, and he/she should only access information that is necessary for providing the medical assistance. The information accessed can not be used for any other purpose than providing medical assistance to patients. Accesses to information in EPRs not needed for providing medical assistance would not be in accordance with Article 8. Also, employees that are not health-care professionals and work within the jurisdiction of Client H are not allowed to access EPRs. Based on the constraints provided by Client H, we decide to express BO-A8 more precisely as follows:

Precise business objective PBO-A8: $C_1 \wedge C_2 \wedge C_3$

- **Constraint C_1 :** Health-care professionals acting on behalf of Client H access:
 - a patient’s EPR only when providing medical assistance to that patient
 - only the information in a patient’s EPR that is necessary for providing medical assistance to that patient
- **Constraint C_2 :** Health-care professionals acting on behalf of Client H do not use the information obtained from a patient’s EPR for any other purpose than providing medical assistance to that patient.
- **Constraint C_3 :** Employees that are not health-care professionals and that work within the jurisdiction of Client H do not access EPRs.

As indicated by PBO-A8’s definition, all three constraints must be fulfilled in order for PBO-A8 to be fulfilled.

B. Describe relevant part of business (Step 1.2 of ValidKI)

To design key indicators to monitor BO-A8 we need to understand the part of business that is to comply with BO-A8 and therefore is to be monitored. “Public hospital *Client H*” has outsourced some of its medical services to two private hospitals. These two are referred to as “Private hospital *X-ray*” and “Private hospital *Blood test analysis*” in Fig. 4. The first hospital does all the X-ray work for Client H, while the second hospital does all the blood test analyses. Client H is not only responsible for its own handling of EPRs, but also the outsourcing partners’ handling of EPRs, when they act on behalf of Client H.

In Fig. 4, the rectangles inside and outside the gray containers represent systems/actors, while the arrows in the figure represent the exchange of data between different systems/actors. In the figure, we only show some of the rectangles and arrows that should be part of the gray containers of “Public hospital *Client H*” and “Private hospital *Blood test analysis*.” All the rectangles and arrows with names in italic that are part of the gray container of “Private hospital *X-ray*” should also be part of the gray containers of “Public hospital *Client H*” and “Private hospital *Blood test analysis*.”

As can be seen in Fig. 4, Client H outsources medical tasks to the two private hospitals, and gets in return the results from performing these tasks. All three health-care institutions employs some kind of EPR system for

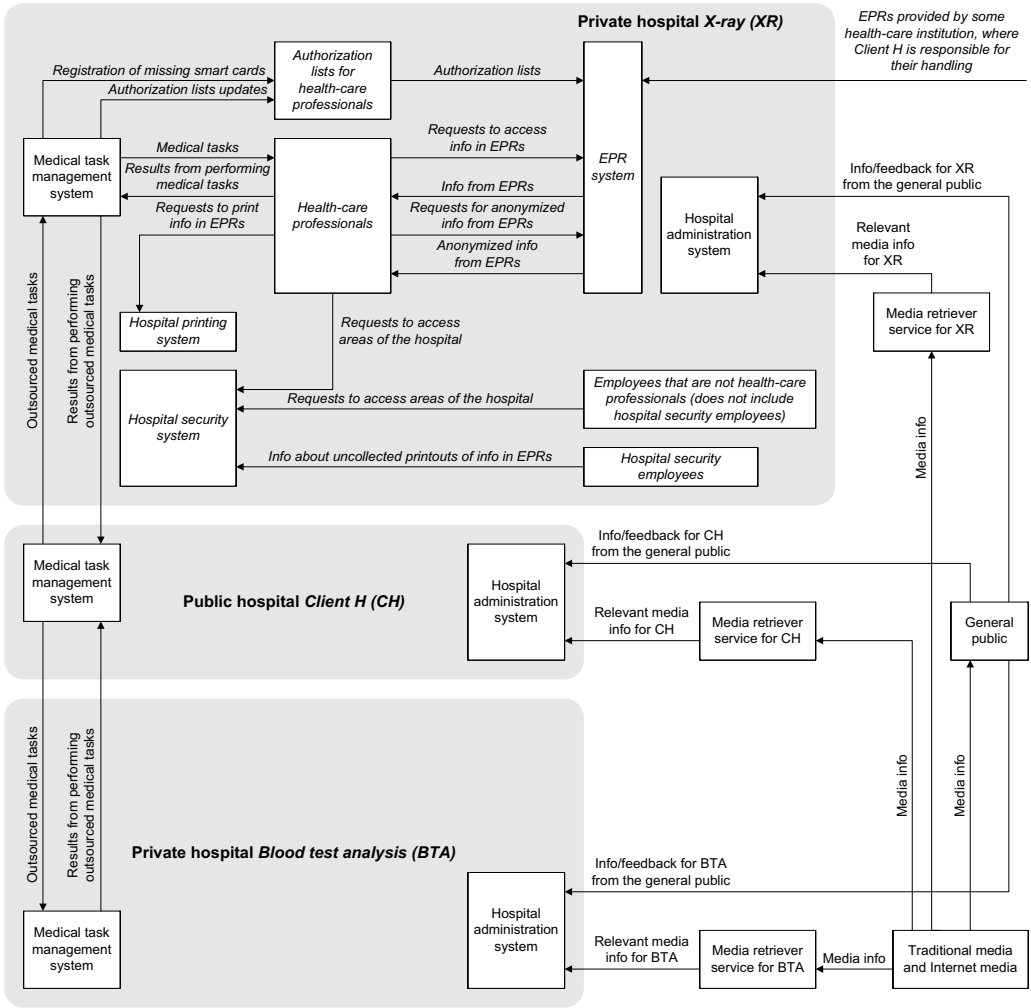


Fig. 4. Specification of relevant part of business

handling the EPRs. An EPR system is “an electronic system with the necessary functionality to record, retrieve, present, communicate, edit, correct, and delete information in electronic patient records” [21]. These systems use EPRs provided by different health-care institutions. As shown in Fig. 4, these systems are only of interest when they handle EPRs where Client H is responsible for their handling.

At the three health-care institutions, most of the medical tasks that a health-care professional conducts during a working day are known in advance. It is known which patients the professional will treat and what kind of information the professional will need access to in order to treat the different patients. Client H and the two outsourcing partners maintain for each health-care professional an authorization list documenting which patients the professional is treating and what kind of information the professional needs for this purpose. These lists are used by the EPR systems and they are updated on a daily basis by the medical task management systems. Many of these updates are automatic. For instance, when Client H is assigned a new patient, then this patient is added to the lists of the health-care professionals who will be treating this patient.

Each EPR is owned by a patient, which is natural since the information stored in the EPR is about the patient in question. As already mentioned, the content of a patient's EPR is both considered sensitive and private. Moreover, some of the EPRs may contain information that is considered highly sensitive and private. Such information may for instance describe medical treatment received by a patient in relation to:

- the patient being the victim of a crime (e.g., rape, violence, etc.);
- sexual transferable diseases or abortion; and
- mortal or infectious mortal diseases.

Information classified as highly sensitive and private is handled with even more care than information that is just classified as sensitive and private. To raise awareness of the criticality of such information and to enable monitoring of its use, the EPR systems at the three health-care institutions tag highly sensitive and private information in EPRs based on predefined rules.

Accesses to information in EPRs can be classified as *authorized* or *unauthorized* based on the authorization lists of health-care professionals. An access is classified as authorized if the professional needs the information to do a planned task. Otherwise, the access is classified as unauthorized. If an access is classified as unauthorized then it is possible to check in retrospect whether the access was necessary. In an emergency situation, for instance when a patient is having a heart attack, a health-care professional often needs access to information in an EPR that he/she was not supposed to access. By checking in retrospect whether unauthorized accesses were necessary it is possible to classify the unauthorized accesses into two groups; one for accesses that were necessary, and one for those that were not. The first group is called *approved* unauthorized accesses, while the second group is called *not approved* unauthorized accesses. All accesses that are classified as not approved unauthorized accesses are considered as *illegal* accesses.

At Client H and the two outsourcing partners, health-care professionals use smart cards for accessing information in EPRs. If a card is lost or stolen, the owner must report it as missing, since missing cards may be used by other health-care professionals or others to access EPRs illegally. When the card has been registered as missing it can no longer be used. When reporting it as missing, the last time the card owner used it before noticing that it was missing is recorded. All accesses to EPRs that have occurred between this time and the time it was registered as missing are considered as illegal accesses.

At the three hospitals, the doors into the different areas are fitted with smart card locks. In order to open a door, an employee needs to insert his/hers smart card into the lock. A security system is used by each hospital to allow or deny an employee access to a specific area based on the employee's access credentials. Moreover, health-care professionals often need to print information in EPRs. Each hospital relies on a printing system to achieve this. This system issues the different print jobs to printers located in rooms with doors fitted with smart card locks. Since each printer is used by a number of employees, the three hospitals run the risk of printed information being disclosed to other employees if the employee responsible for the print job forgets to collect his/hers printout. To minimize this risk, each hospital has security employees that collect uncollected printouts of information from EPRs at the different printers on a regular basis. Each printer at the three hospitals annotates each printout with the date and time it was printed, as well as an ID for the employee that issued the print job. A security employee removes a printout of sensitive and private information from an EPR if it has been laying on the printer for 30 minutes or more, while he/she removes a printout of highly sensitive and private information if it has been laying on the printer for 15 minutes or more. For each removed printout, the health-care professional that issued the print job is notified about the removal and asked to collect the printout at the security office at the hospital in question.

A health-care professional relies from time to time on information obtained from patients' EPRs for other purposes than providing medical assistance to the patients in question. The information may be needed for the purpose of providing medical assistance to another patient, or it may be needed in research projects. To support these tasks, the three hospitals have made it possible for health-care professionals to obtain anonymized information from EPRs, i.e., information that cannot be linked to specific patients. It should be noticed that health-care professionals need to obtain specific permissions to obtain and use anonymized information from EPRs.

At each of the three hospitals, a media retriever service is used to collect relevant information from the traditional media (newspapers, TV, radio, etc.) and the Internet media (Internet newspapers, etc.). The three hospitals also encourage the general public to provide feedback on how satisfied they are with the hospitals' services. The general public also serves another purpose for the three hospitals. The media retriever services can only to a limited extent retrieve information from social media (Facebook, Twitter, etc.) and Internet forums. The three hospitals therefore

TABLE I
CONSEQUENCE SCALE FOR THE ASSET “FULFILLMENT OF PBO-A8” (TOP) AND LIKELIHOOD SCALE (BOTTOM)

Consequence	Description
Catastrophic	Law enforcement agencies penalize Client H after having been notified about the incident
Major	Health authorities penalize Client H after having been notified about the incident
Moderate	Health authorities are notified about the incident
Minor	Head of hospital is notified about the incident
Insignificant	Head of department is notified about the incident

Likelihood	Description
Certain	Five times or more per year $[50, \infty)$: 10 years
Likely	Two to five times per year $[20, 49)$: 10 years
Possible	Once a year $[6, 19)$: 10 years
Unlikely	Less than once per year $[2, 5)$: 10 years
Rare	Less than once per ten years $[0, 1)$: 10 years

TABLE II
RISK EVALUATION MATRIX FOR THE ASSET “FULFILLMENT OF PBO-A8”

Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare					
Unlikely					
Possible					
Likely					
Certain					

encourage the general public to notify them about information found in social media or on Internet forums that may be of relevance. A person of the general public is awarded if the information is very relevant. The information provided by the media retriever services and the general public is first and foremost used by the hospitals to assess how they are perceived by the public. Sometimes, however, the collected information may indicate or reveal that information from EPRs have been leaked to the public.

V. IDENTIFY RISKS TO FULFILLMENT OF BUSINESS OBJECTIVE

A. Specify risk acceptance criteria (Step 2.1 of ValidKI)

Before we specify the risk acceptance criteria, we need to establish scales for measuring likelihood and consequence. Table I presents these scales. We view “Fulfillment of PBO-A8” as the asset to be protected. In Table II the risk acceptance criteria for the asset “Fulfillment of PBO-A8” are expressed in terms of a risk evaluation matrix. Risks whose values belong to the white area of the matrix are acceptable, while risks whose values belong to the gray area are unacceptable.

B. Risk identification and estimation (Step 2.2 of ValidKI)

Based on the information provided by the representatives of Client H, we identify and estimate risk. For this purpose we use the CORAS methodology [22]. However, other approaches to risk analysis may be used instead. Using CORAS we identify how threats may initiate risks that harm the asset “Fulfillment of PBO-A8” if they occur.

The CORAS threat diagram in Fig. 5 provides a high-level overview of how the fulfillment of the precise business objective PBO-A8 may be harmed. The threat diagram contains four referring threat scenarios that refer to the referenced threat scenarios in Figs. 6 – 9. We refer to i_x and o_y of the referring threat scenarios as in-gate and out-gate, respectively. Relations to an element inside a referenced threat scenario must go through an in-gate, while relations to an element outside the referenced threat scenario must go through an out-gate. The likelihood value of an in-gate i_x documents the contribution of an element outside the referenced threat scenario via gate i_x to the

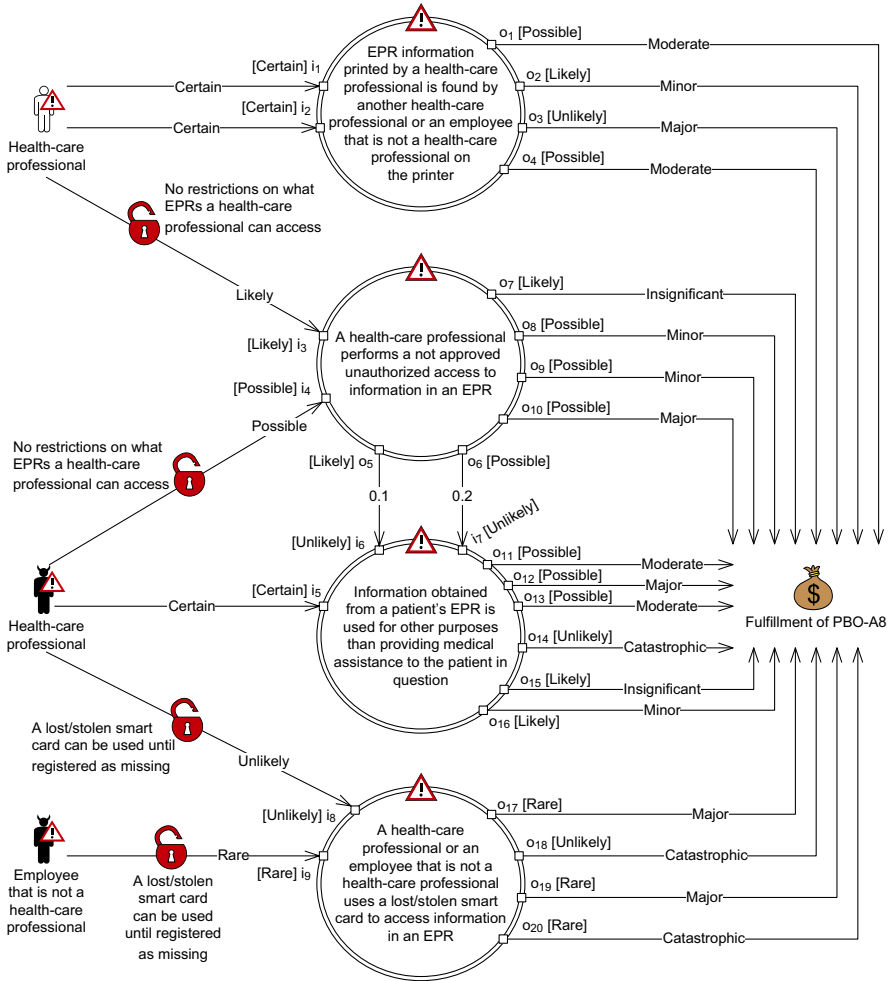


Fig. 5. CORAS threat diagram providing a high-level overview of the results from the risk identification and estimation

likelihood of an element inside the referenced threat scenario, while the likelihood of the out-gate o_y documents the contribution of the likelihood of an element inside the referenced threat scenario via gate o_y to the likelihood of an element outside the referenced threat scenario.

The CORAS threat diagram in Fig. 5 contains three human threats; one accidental (the white one) and two deliberate (the black ones). The accidental human threat “Health-care professional” may initiate the threat scenario “Unauthorized access to information in a patient’s EPR” in the referenced threat scenario “A health-care professional performs a not approved unauthorized access to information in an EPR” in Fig. 7 via the in-gate i_3 with likelihood “Likely” by exploiting the vulnerability “No restrictions on what EPRs a health-care professional can access.” We can also see that the deliberate human threat “Health-care professional” may initiate this threat scenario via the in-gate i_4 with likelihood “Possible” by exploiting the same vulnerability, and that the threat scenario occurs with likelihood “Certain.” If the threat scenario in Fig. 7 occurs then it leads to the threat scenario “Unauthorized access to sensitive and private information” in the same figure with conditional likelihood “0.7.” This threat scenario leads to the risk “R5: Not approved unauthorized access to sensitive and private information in an EPR, where the owner

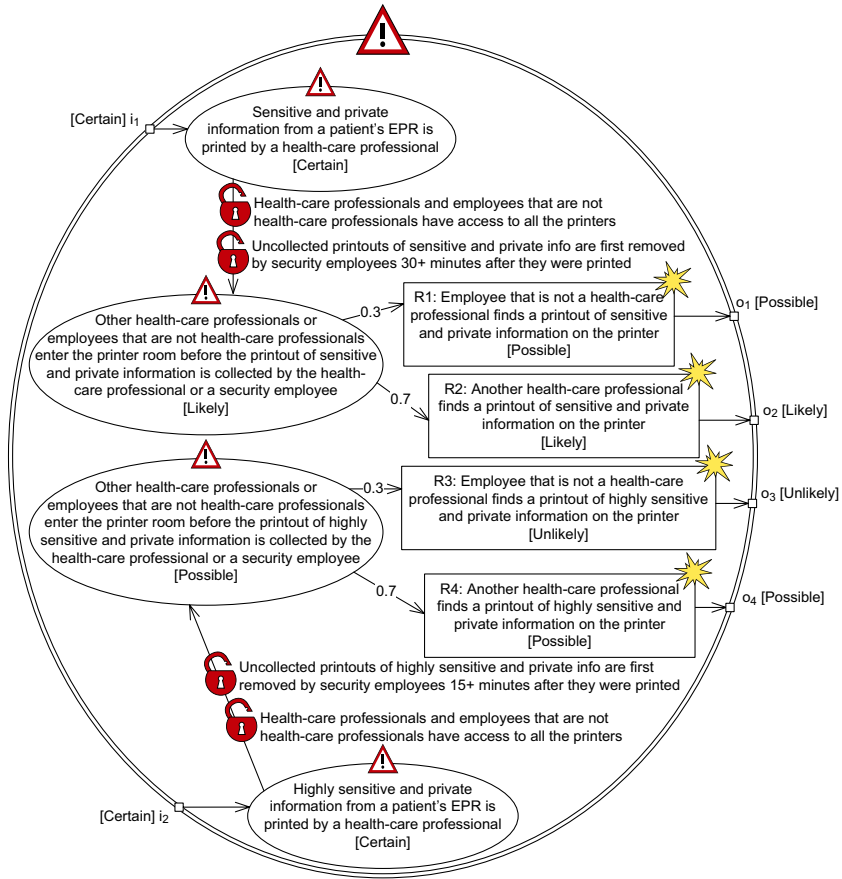
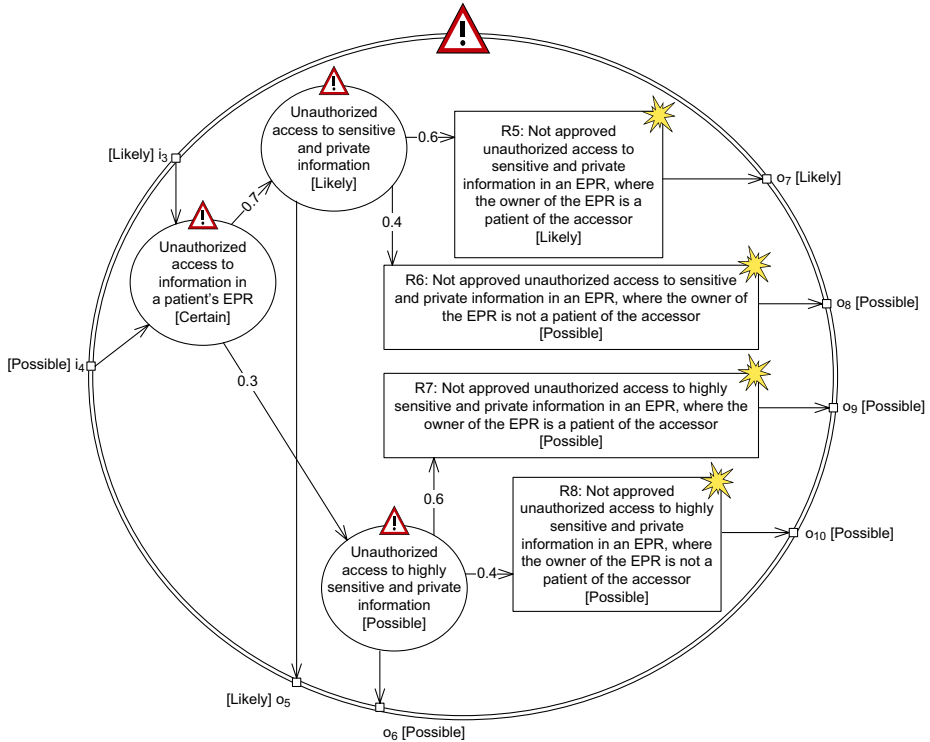


Fig. 6. The referenced threat scenario “EPR information printed by a health-care professional is found by another health-care professional or an employee that is not a health-care professional on the printer,” referred to in Fig. 5

of the EPR is a patient of the accessor” with conditional likelihood “0.6” if it occurs. The risk occurs with likelihood “Likely.” As can be seen in Figs. 5 and 7, the risk impacts the asset “Fulfillment of PBO-A8” via the out-gate o_7 with consequence “Insignificant” if it occurs.

The referenced threat scenarios in Figs. 6 – 9 document risks that affect the fulfillment of the constraints referred to in the precise business objective PBO-A8. The risks R_2 , R_4 , $R_5 - R_8$, R_{15} , and R_{16} affect the fulfillment of constraint C_1 , while the risks $R_9 - R_{14}$ affect the fulfillment of constraint C_2 . Moreover, the risks R_1 , R_3 , R_{17} , and R_{18} affect the fulfillment of constraint C_3 . Notice that in the referenced threat scenario in Fig. 7, we distinguish between not approved unauthorized accesses to information in EPRs where the owner of the EPR is a patient and not a patient of the accessor. Client H finds it most serious if the owner of the EPR is not a patient of the accessor. We also distinguish between not approved unauthorized accesses to sensitive and private information and not approved unauthorized accesses to highly sensitive and private information. Naturally, Client H finds not approved unauthorized accesses to the latter type of information the most serious.



A health-care professional performs a not approved unauthorized access to information in an EPR

Fig. 7. The referenced threat scenario “A health-care professional performs a not approved unauthorized access to information in an EPR,” referred to in Fig. 5

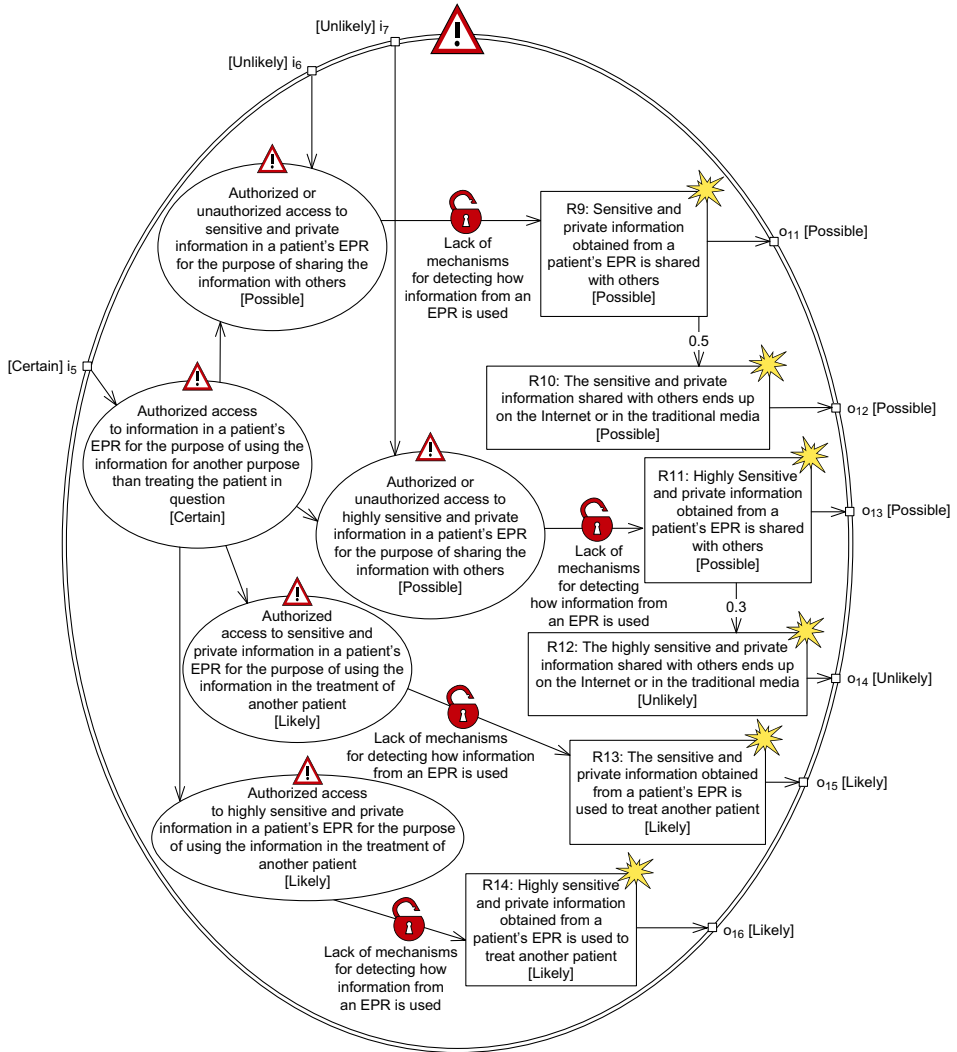
TABLE III
THE RISK EVALUATION MATRIX FROM TABLE II WITH THE ACCEPTABLE AND UNACCEPTABLE RISKS INSERTED

Consequence \ Likelihood	Insignificant	Minor	Moderate	Major	Catastrophic
Rare				R_{15}, R_{17}	R_{18}
Unlikely				R_3	R_{12}, R_{16}
Possible		R_6, R_7	R_1, R_4, R_9, R_{11}	R_8, R_{10}	
Likely	R_5, R_{13}	R_2, R_{14}			
Certain					

C. Risk evaluation (Step 2.3 of ValidKI)

The risk evaluation consists in plotting the risks into the risk evaluation matrix according to their likelihoods and consequences. As indicated in Table III, four out of the 18 risks namely R_8 , R_{10} , R_{12} , and R_{16} are unacceptable with respect to the fulfillment of the precise business objective PBO-A8.

During the risk evaluation, we also decide that some of the risks need to be accumulated since they pull in the same direction. We decide to accumulate the following risks: R_1 and R_2 ; R_3 and R_4 ; R_{15} and R_{17} ; and R_{16} and R_{18} . All of these risks, with the exception of R_{18} , are acceptable when considered in isolation. Risks are accumulated by accumulating their likelihood and consequence values. We accumulate the risks as follows:



Information obtained from a patient's EPR is used for other purposes than providing medical assistance to the patient in question

Fig. 8. The referenced threat scenario “Information obtained from a patient’s EPR is used for other purposes than providing medical assistance to the patient in question,” referred to in Fig. 5

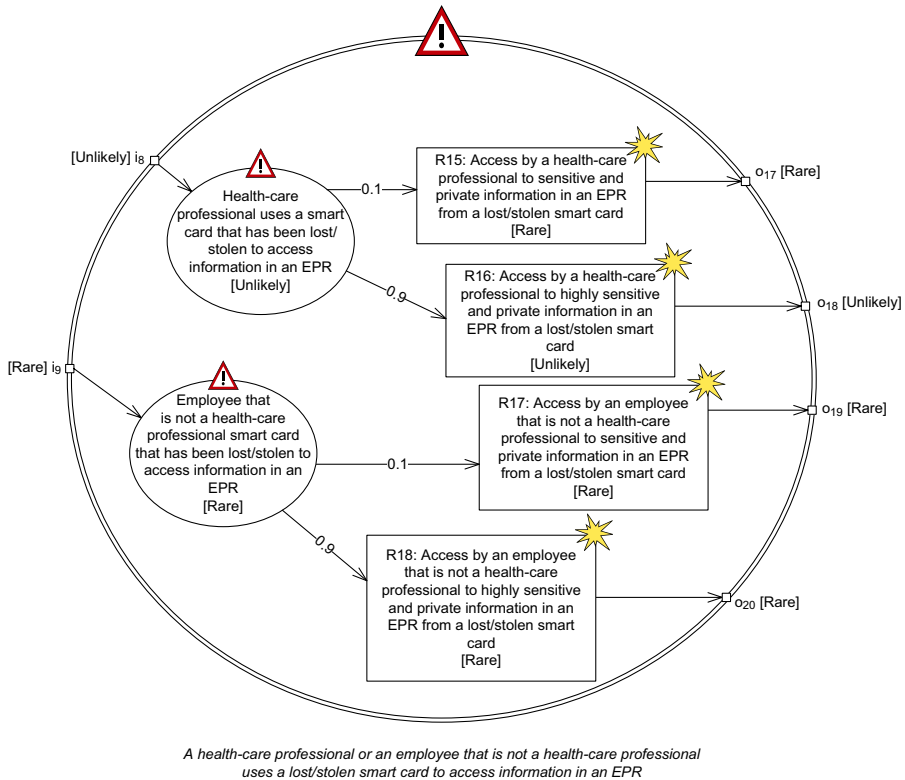


Fig. 9. The referenced threat scenario “A health-care professional or an employee that is not a health-care professional uses a lost/stolen smart card to access information in an EPR,” referred to in Fig. 5

- The accumulated risk “*R1&R2*: Another health-care professional or employee that is not a health-care professional finds a printout of sensitive and private information on the printer.” It occurs with likelihood “Likely” and it impacts the asset with consequence “Moderate.” The accumulated risk is based on:
 - The risk *R1* which occurs with likelihood “Possible,” while it impacts the asset with consequence “Moderate.”
 - The risk *R2* which occurs with likelihood “Likely,” while it impacts the asset with consequence “Minor.”
- The accumulated risk “*R3&R4*: Another health-care professional or employee that is not a health-care professional finds a printout of highly sensitive and private information on the printer.” It occurs with likelihood “Possible” and it impacts the asset with consequence “Major.” The accumulated risk is based on:
 - The risk *R3* which occurs with likelihood “Unlikely,” while it impacts the asset with consequence “Major.”
 - The risk *R4* which occurs with likelihood “Possible,” while it impacts the asset with consequence “Moderate.”
- The accumulated risk “*R15&R17*: Access by a health-care professional or an employee that is not a health-care professional to sensitive and private information in an EPR from a lost/stolen smart card.” It occurs with likelihood “Rare” and it impacts the asset with consequence “Major.” The accumulated risk is based on:
 - The risk *R15* which occurs with likelihood “Rare,” while it impacts the asset with consequence “Major.”
 - The risk *R17* which occurs with likelihood “Rare,” while it impacts the asset with consequence “Major.”
- The accumulated risk “*R16&R18*: Access by a health-care professional or an employee that is not a health-care professional to highly sensitive and private information in an EPR from a lost/stolen smart card.” It occurs

TABLE IV
THE RISK EVALUATION MATRIX FROM TABLE III AFTER RISKS HAVE BEEN ACCUMULATED

Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare				$R15\&R17$	
Unlikely					$R12,$ $R16\&R18$
Possible		$R6, R7$	$R9, R11$	$R3\&R4, R8,$ $R10$	
Likely	$R13$	$R5, R14$	$R1\&R2$		
Certain					

with likelihood “Unlikely” and it impacts the asset with consequence “Catastrophic.” The accumulated risk is based on:

- The risk $R16$ which occurs with likelihood “Unlikely,” while it impacts the asset with consequence “Catastrophic.”
- The risk $R18$ which occurs with likelihood “Rare,” while it impacts the asset with consequence “Catastrophic.”

Since we are operating with a coarse-grained likelihood scale with intervals, we find it sufficient to do a rough aggregation of the likelihoods in order to determine to which likelihood interval the different accumulated risks belong. For the accumulated risk $R15\&R17$ we end up with the likelihood “Rare,” while for each of the other accumulated risks, we end up with an aggregated likelihood that gravitates towards the highest of the two likelihoods. We therefore decide to use the highest likelihood to represent the accumulated likelihood in each of these cases. Moreover, we accumulate consequences by taking the average. In all of the cases where the two consequence values differ, we end up with an average that gravitates towards the highest consequence value. We therefore find it suitable to use the highest consequence value to represent the accumulated consequence in each of these cases.

In Table IV we have plotted the accumulated risks according to their likelihoods and consequences. As we can see from the table, all the accumulated risks with the exception of $R15\&R17$ are unacceptable. Table IV shows that the risks $R1\&R2$, $R3\&R4$, $R8$, $R10$, $R12$, and $R16\&R18$ are unacceptable with respect to the fulfillment of the precise business objective PBO-A8.

VI. IDENTIFY KEY INDICATORS TO MONITOR RISKS

A. Deploy sensors to monitor risks (Step 3.1 of ValidKI)

Fig. 10, which is a detailing of the target description in Fig. 4, specifies the deployment of sensors in the relevant part of business. This specification corresponds to the sensor deployment specification referred to in Fig. 2. An antenna-like symbol is used to represent each sensor in Fig. 10. The different sensors monitor data messages exchanged within the relevant part of business. The results from the monitoring are to be used in the calculation of key indicators.

In Fig. 10, sensor deployments are only shown for “Private hospital *X-ray*.” It should be noticed that “Public hospital *Client H*” and “Private hospital *Blood test analysis*” will have the same sensors as “Private hospital *X-ray*.” The following sensors are deployed in the relevant part of business:

- $S_{CH-REG-MIS-SC}$, $S_{BTA-REG-MIS-SC}$, and $S_{XR-REG-MIS-SC}$ monitor data messages related to the registration of missing smart cards at Client H, Blood test analysis, and X-ray, respectively.
- $S_{CH-AUTH-LIST}$, $S_{BTA-AUTH-LIST}$, and $S_{XR-AUTH-LIST}$ monitor data messages related to the authorization lists employed by the EPR systems at Client H, Blood test analysis, and X-ray, respectively.
- $S_{CH-ACC-INFO-EPR}$, $S_{BTA-ACC-INFO-EPR}$, and $S_{XR-ACC-INFO-EPR}$ monitor data messages where each message is a request issued by health-care professional to access information in an EPR at Client H, Blood test analysis, and X-ray, respectively. It is not necessary to monitor the actual information received, since health-care professionals will always get the information they request.
- $S_{CH-INFO-GP}$, $S_{BTA-INFO-GP}$, and $S_{XR-INFO-GP}$ monitor data messages where each message contains info/feedback from the general public for Client H, Blood test analysis, and X-ray, respectively.

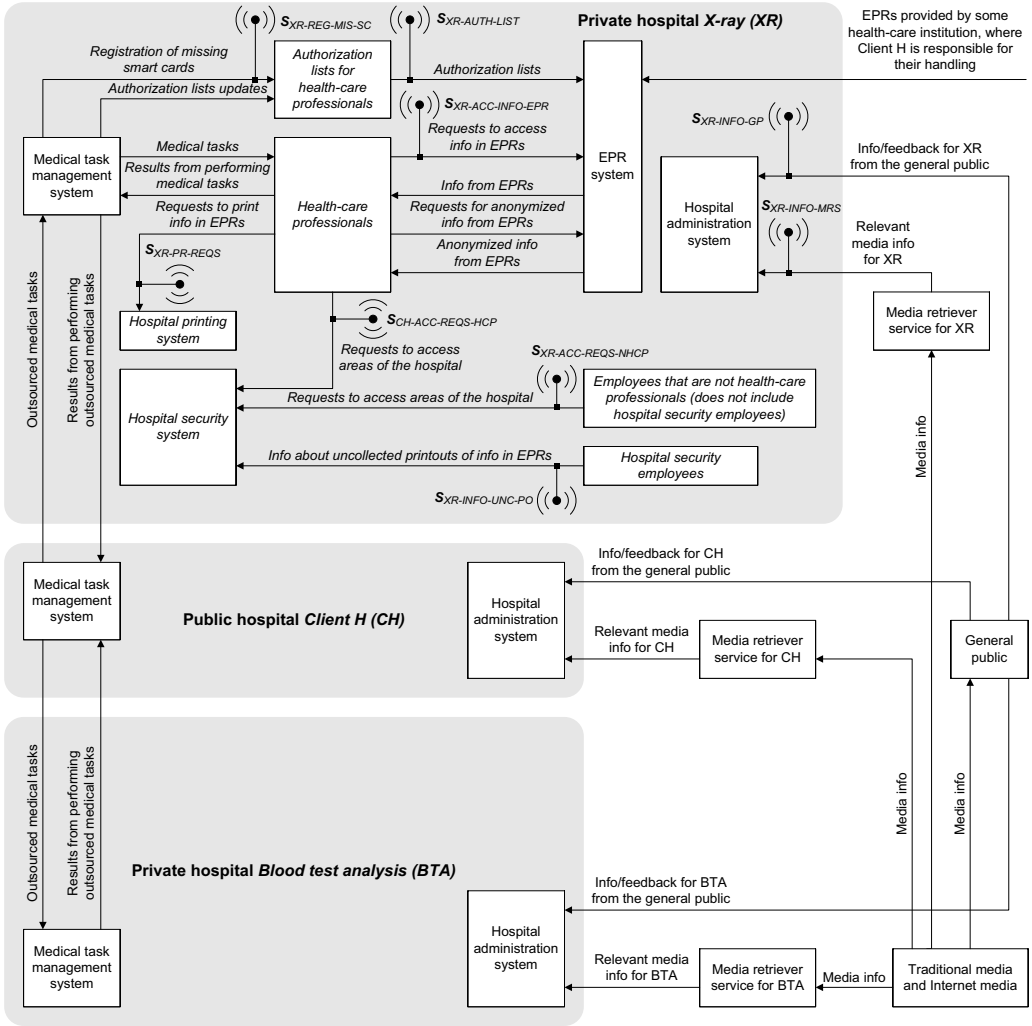


Fig. 10. Deployment of sensors in the relevant part of business

- $S_{CH-INFO-MRS}$, $S_{BTA-INFO-MRS}$, and $S_{XR-INFO-MRS}$ monitor data messages where each message contains relevant information collected by media retriever services from the traditional media or the Internet for Client H, Blood test analysis, and X-ray, respectively.
- $S_{CH-PR-REQS}$, $S_{BTA-PR-REQS}$, and $S_{XR-PR-REQS}$ monitor data messages related to printing of information in EPRs by health-care professionals at Client H, Blood test analysis, and X-ray, respectively.
- $S_{CH-ACC-REQS-HCP}$, $S_{BTA-ACC-REQS-HCP}$, and $S_{XR-ACC-REQS-HCP}$ monitor data messages related to area access requests issued by health-care professionals at Client H, Blood test analysis, and X-ray, respectively.
- $S_{CH-ACC-REQS-NHCP}$, $S_{BTA-ACC-REQS-NHCP}$, and $S_{XR-ACC-REQS-NHCP}$ monitor data messages related to area access requests issued by employees that are not health-care professionals at Client H, Blood test analysis, and X-ray, respectively.
- $S_{CH-INFO-UNC-PO}$, $S_{BTA-INFO-UNC-PO}$, and $S_{XR-INFO-UNC-PO}$ monitor data messages related to registrations of uncol-

TABLE V
KEY INDICATOR REQUIREMENTS SPECIFICATIONS FOR THE COMPOSITE KEY INDICATOR $K_{PR-SP-EPR-INFO}$ AND THE BASIC KEY INDICATORS $K_{CH-PR-SP-EPR-INFO}$, $K_{BTA-PR-SP-EPR-INFO}$, AND $K_{XR-PR-SP-EPR-INFO}$

Requirements for $K_{X-PR-SP-EPR-INFO}$, where $X \in \{CH, BTA, XR\}$	
In:	$S_{X-ACC-REQS-NHCP}, S_{X-ACC-REQS-HCP}, S_{X-PR-REQS}, S_{X-INFO-UNC-PO} : M^*$
Out:	$K_{X-PR-SP-EPR-INFO} : \mathbb{R}$
Description:	$K_{X-PR-SP-EPR-INFO} =$ “The number of times since the monitoring started that health-care professionals or employees that are not health-care professionals have found printouts of sensitive and private information from EPRs on printers at X ”
Requirements for $K_{PR-SP-EPR-INFO}$	
In:	$S_{CH-ACC-REQS-NHCP}, S_{BTA-ACC-REQS-NHCP}, S_{XR-ACC-REQS-NHCP} : M^*$ $S_{CH-ACC-REQS-HCP}, S_{BTA-ACC-REQS-HCP}, S_{XR-ACC-REQS-HCP} : M^*$ $S_{CH-PR-REQS}, S_{BTA-PR-REQS}, S_{XR-PR-REQS} : M^*$ $S_{CH-INFO-UNC-PO}, S_{BTA-INFO-UNC-PO}, S_{XR-INFO-UNC-PO} : M^*$
Out:	$K_{PR-SP-EPR-INFO} : \mathbb{R}$
Description:	$K_{PR-SP-EPR-INFO} = \frac{10 \cdot (K_{CH-PR-SP-EPR-INFO} + K_{BTA-PR-SP-EPR-INFO} + K_{XR-PR-SP-EPR-INFO})}{\text{Number of years since the monitoring started}}$

TABLE VI
KEY INDICATOR REQUIREMENTS SPECIFICATIONS FOR THE COMPOSITE KEY INDICATOR $K_{PR-HSP-EPR-INFO}$ AND THE BASIC KEY INDICATORS $K_{CH-PR-HSP-EPR-INFO}$, $K_{BTA-PR-HSP-EPR-INFO}$, AND $K_{XR-PR-HSP-EPR-INFO}$

Requirements for $K_{X-PR-HSP-EPR-INFO}$, where $X \in \{CH, BTA, XR\}$	
In:	$S_{X-ACC-REQS-NHCP}, S_{X-ACC-REQS-HCP}, S_{X-PR-REQS}, S_{X-INFO-UNC-PO} : M^*$
Out:	$K_{X-PR-HSP-EPR-INFO} : \mathbb{R}$
Description:	$K_{X-PR-HSP-EPR-INFO} =$ “The number of times since the monitoring started that health-care professionals or employees that are not health-care professionals have found printouts of highly sensitive and private information from EPRs on printers at X ”
Requirements for $K_{PR-HSP-EPR-INFO}$	
In:	$S_{CH-ACC-REQS-NHCP}, S_{BTA-ACC-REQS-NHCP}, S_{XR-ACC-REQS-NHCP} : M^*$ $S_{CH-ACC-REQS-HCP}, S_{BTA-ACC-REQS-HCP}, S_{XR-ACC-REQS-HCP} : M^*$ $S_{CH-PR-REQS}, S_{BTA-PR-REQS}, S_{XR-PR-REQS} : M^*$ $S_{CH-INFO-UNC-PO}, S_{BTA-INFO-UNC-PO}, S_{XR-INFO-UNC-PO} : M^*$
Out:	$K_{PR-HSP-EPR-INFO} : \mathbb{R}$
Description:	$K_{PR-HSP-EPR-INFO} = \frac{10 \cdot (K_{CH-PR-HSP-EPR-INFO} + K_{BTA-PR-HSP-EPR-INFO} + K_{XR-PR-HSP-EPR-INFO})}{\text{Number of years since the monitoring started}}$

lected printouts of information from EPRs by security employees at Client H, Blood test analysis, and X-ray, respectively.

B. Specify requirements to key indicators wrt deployed sensors (Step 3.2 of ValidKI)

Two key indicators $K_{PR-SP-EPR-INFO}$ and $K_{PR-HSP-EPR-INFO}$ are identified to monitor the likelihood values of the two unacceptable risks $R1\&R2$ and $R3\&R4$, respectively. In Tables V and VI their requirements are given. The two key indicators calculate likelihoods with respect to a ten year period, because the likelihoods in the likelihood scale in Table I are defined with respect to a ten year period. Both key indicators are composed of basic key indicators. Table V presents the requirements to the basic key indicators that $K_{PR-SP-EPR-INFO}$ is composed of, while Table VI presents the requirements to the basic key indicators that $K_{PR-HSP-EPR-INFO}$ is composed of.

For each key indicator we specify required sensor data. All of the key indicators rely on sequences of data

TABLE VII

KEY INDICATOR REQUIREMENTS SPECIFICATIONS FOR THE COMPOSITE KEY INDICATOR $K_{\text{NOT-APP-UNAUTH-ACC}}$ AND THE BASIC KEY INDICATORS $K_{\text{CH-NOT-APP-UNAUTH-ACC}}$, $K_{\text{BTA-NOT-APP-UNAUTH-ACC}}$, AND $K_{\text{XR-NOT-APP-UNAUTH-ACC}}$

Requirements for $K_{X\text{-NOT-APP-UNAUTH-ACC}}$, where $X \in \{\text{CH, BTA, XR}\}$	
In:	$S_{X\text{-AUTH-LIST}}, S_{X\text{-ACC-INFO-EPR}} : M^*$
Out:	$K_{X\text{-NOT-APP-UNAUTH-ACC}} : \mathbb{N}$
Description:	$K_{X\text{-NOT-APP-UNAUTH-ACC}} =$ “The number of not approved unauthorized accesses at X since the monitoring started to highly sensitive and private information in EPRs, where the owners of the EPRs are not patients of the accessors”
Requirements for $K_{\text{NOT-APP-UNAUTH-ACC}}$	
In:	$S_{\text{CH-AUTH-LIST}}, S_{\text{BTA-AUTH-LIST}}, S_{\text{XR-AUTH-LIST}} : M^*$ $S_{\text{CH-ACC-INFO-EPR}}, S_{\text{BTA-ACC-INFO-EPR}}, S_{\text{XR-ACC-INFO-EPR}} : M^*$
Out:	$K_{\text{NOT-APP-UNAUTH-ACC}} : \mathbb{R}$
Description:	$K_{\text{NOT-APP-UNAUTH-ACC}} = \frac{10 \cdot (K_{\text{CH-NOT-APP-UNAUTH-ACC}} + K_{\text{BTA-NOT-APP-UNAUTH-ACC}} + K_{\text{XR-NOT-APP-UNAUTH-ACC}})}{\text{Number of years since the monitoring started}}$

TABLE VIII

KEY INDICATOR REQUIREMENTS SPECIFICATIONS FOR THE COMPOSITE KEY INDICATOR $K_{\text{SP-EPR-INFO}}$ AND THE BASIC KEY INDICATORS $K_{\text{CH-SP-EPR-INFO}}$, $K_{\text{BTA-SP-EPR-INFO}}$, AND $K_{\text{XR-SP-EPR-INFO}}$

Requirements for $K_{X\text{-SP-EPR-INFO}}$, where $X \in \{\text{CH, BTA, XR}\}$	
In:	$S_{X\text{-ACC-INFO-EPR}}, S_{X\text{-INFO-GP}}, S_{X\text{-INFO-MRS}} : M^*$
Out:	$K_{X\text{-SP-EPR-INFO}} : \mathbb{N}$
Description:	$K_{X\text{-SP-EPR-INFO}} =$ “The number of times since the monitoring started that sensitive and private information from patients’ EPRs have been shared by health-care professionals with others and where this information have ended up in the traditional media or on the Internet”
Requirements for $K_{\text{SP-EPR-INFO}}$	
In:	$S_{\text{CH-ACC-INFO-EPR}}, S_{\text{BTA-ACC-INFO-EPR}}, S_{\text{XR-ACC-INFO-EPR}} : M^*$ $S_{\text{CH-INFO-GP}}, S_{\text{BTA-INFO-GP}}, S_{\text{XR-INFO-GP}} : M^*$ $S_{\text{CH-INFO-MRS}}, S_{\text{BTA-INFO-MRS}}, S_{\text{XR-INFO-MRS}} : M^*$
Out:	$K_{\text{SP-EPR-INFO}} : \mathbb{R}$
Description:	$K_{\text{SP-EPR-INFO}} = \frac{10 \cdot (K_{\text{CH-SP-EPR-INFO}} + K_{\text{BTA-SP-EPR-INFO}} + K_{\text{XR-SP-EPR-INFO}})}{\text{Number of years since the monitoring started}}$

messages (M^*) gathered by the different sensors. We also specify the output type and requirements to output. For a key indicator K we refer to its requirement description as $Req(K)$.

Key indicators have also been identified for monitoring the unacceptable risks $R8$, $R10$, $R12$, and $R16\&R18$. Tables VII, VIII, IX, and X specify requirements to key indicators for monitoring the likelihood values of the risks $R8$, $R10$, $R12$, and $R16\&R18$, respectively.

TABLE IX
KEY INDICATOR REQUIREMENTS SPECIFICATIONS FOR THE COMPOSITE KEY INDICATOR $K_{\text{HSP-EPR-INFO}}$ AND THE BASIC KEY INDICATORS $K_{\text{CH-HSP-EPR-INFO}}$, $K_{\text{BTA-HSP-EPR-INFO}}$, AND $K_{\text{XR-HSP-EPR-INFO}}$

Requirements for $K_{X\text{-HSP-EPR-INFO}}$, where $X \in \{\text{CH, BTA, XR}\}$	
In:	$S_{X\text{-ACC-INFO-EPR}}, S_{X\text{-INFO-GP}}, S_{X\text{-INFO-MRS}} : M^*$
Out:	$K_{X\text{-HSP-EPR-INFO}} : \mathbb{N}$
Description:	$K_{X\text{-HSP-EPR-INFO}} =$ “The number of times since the monitoring started that highly sensitive and private information from patients’ EPRs have been shared by health-care professionals with others and where this information have ended up in the traditional media or on the Internet”
Requirements for $K_{\text{HSP-EPR-INFO}}$	
In:	$S_{\text{CH-ACC-INFO-EPR}}, S_{\text{BTA-ACC-INFO-EPR}}, S_{\text{XR-ACC-INFO-EPR}} : M^*$ $S_{\text{CH-INFO-GP}}, S_{\text{BTA-INFO-GP}}, S_{\text{XR-INFO-GP}} : M^*$ $S_{\text{CH-INFO-MRS}}, S_{\text{BTA-INFO-MRS}}, S_{\text{XR-INFO-MRS}} : M^*$
Out:	$K_{\text{HSP-EPR-INFO}} : \mathbb{R}$
Description:	$K_{\text{HSP-EPR-INFO}} = \frac{10 \cdot (K_{\text{CH-HSP-EPR-INFO}} + K_{\text{BTA-HSP-EPR-INFO}} + K_{\text{XR-HSP-EPR-INFO}})}{\text{Number of years since the monitoring started}}$

TABLE X
KEY INDICATOR REQUIREMENTS SPECIFICATIONS FOR THE COMPOSITE KEY INDICATOR $K_{\text{ILL-ACC-SC}}$ AND THE BASIC KEY INDICATORS $K_{\text{CH-ILL-ACC-SC}}$, $K_{\text{BTA-ILL-ACC-SC}}$, AND $K_{\text{XR-ILL-ACC-SC}}$

Requirements for $K_{X\text{-ILL-ACC-SC}}$, where $X \in \{\text{CH, BTA, XR}\}$	
In:	$S_{X\text{-REG-MIS-SC}}, S_{X\text{-ACC-INFO-EPR}} : M^*$
Out:	$K_{X\text{-ILL-ACC-SC}} : \mathbb{N}$
Description:	$K_{X\text{-ILL-ACC-SC}} =$ “The number of illegal accesses at X since the monitoring started to highly sensitive and private information in EPRs from lost/stolen smart cards”
Requirements for $K_{\text{ILL-ACC-SC}}$	
In:	$S_{\text{CH-REG-MIS-SC}}, S_{\text{BTA-REG-MIS-SC}}, S_{\text{XR-REG-MIS-SC}} : M^*$ $S_{\text{CH-ACC-INFO-EPR}}, S_{\text{BTA-ACC-INFO-EPR}}, S_{\text{XR-ACC-INFO-EPR}} : M^*$
Out:	$K_{\text{ILL-ACC-SC}} : \mathbb{R}$
Description:	$K_{\text{ILL-ACC-SC}} = \frac{10 \cdot (K_{\text{CH-ILL-ACC-SC}} + K_{\text{BTA-ILL-ACC-SC}} + K_{\text{XR-ILL-ACC-SC}})}{\text{Number of years since the monitoring started}}$

VII. EVALUATE INTERNAL VALIDITY

A. Express business objective in terms of key indicators (Step 4.1 of ValidKI)

The precise business objective PBO-A8’ is a reformulation of the precise business objective PBO-A8 expressed in terms of key indicators.

$$\begin{aligned}
 \text{PBO-A8}' = & K_{\text{PR-SP-EPR-INFO}} \in [0, 19] \wedge \text{Req}(K_{\text{SP-PR-EPR-INFO}}) \wedge \\
 & K_{\text{PR-HSP-EPR-INFO}} \in [0, 5] \wedge \text{Req}(K_{\text{HSP-PR-EPR-INFO}}) \wedge \\
 & K_{\text{NOT-APP-UNAUTH-ACC}} \in [0, 5] \wedge \text{Req}(K_{\text{NOT-APP-UNAUTH-ACC}}) \wedge \\
 & K_{\text{SP-EPR-INFO}} \in [0, 5] \wedge \text{Req}(K_{\text{SP-EPR-INFO}}) \wedge \\
 & K_{\text{HSP-EPR-INFO}} \in [0, 1] \wedge \text{Req}(K_{\text{HSP-EPR-INFO}}) \wedge \\
 & K_{\text{ILL-ACC-SC}} \in [0, 1] \wedge \text{Req}(K_{\text{ILL-ACC-SC}})
 \end{aligned}$$

The precise business objective PBO-A8 is fulfilled if the likelihood values of the six unacceptable risks $R1\&R2$, $R3\&R4$, $R8$, $R10$, $R12$, and $R16\&R18$ change in such a way that the six risks become acceptable. The risks become acceptable if their likelihood values change in the following way:

TABLE XI
THE RISK EVALUATION MATRIX WHEN THE PRECISE BUSINESS OBJECTIVE PBO-A8 IS FULFILLED

Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare			$R1 \& R2'$	$R3 \& R4'$, $R8'$, $R10'$, $R15 \& R17$	$R12$, $R16 \& R18$
Unlikely			$R1 \& R2''$	$R3 \& R4''$, $R8''$, $R10''$	
Possible		$R6$, $R7$	$R1 \& R2'''$, $R9$, $R11$		
Likely	$R13$	$R5$, $R14$			
Certain					

- The risk $R1 \& R2$ becomes acceptable if the likelihood changes from “Likely” to “Possible,” “Unlikely,” or “Rare.” The likelihood will change in such a way if the composite key indicator $K_{PR-SP-EPR-INFO}$, monitoring the likelihood, is contained in the interval $[0, 19]$ (interval capturing both “Rare: $[0, 1] : 10$ years,” “Unlikely: $[2, 5] : 10$ years,” and “Possible: $[6, 19] : 10$ years”).
- The risk $R3 \& R4$ becomes acceptable if the likelihood changes from “Possible” to “Unlikely” or “Rare.” The likelihood will change in such a way if the composite key indicator $K_{PR-HSP-EPR-INFO}$, monitoring the likelihood, is contained in the interval $[0, 5]$ (interval capturing both “Rare: $[0, 1] : 10$ years” and “Unlikely: $[2, 5] : 10$ years”).
- The risk $R8$ becomes acceptable if the likelihood changes from “Possible” to “Unlikely” or “Rare.” The likelihood will change in such a way if the composite key indicator $K_{NOT-APP-UNAUTH-ACC}$, monitoring the likelihood, is contained in the interval $[0, 5]$ (interval capturing both “Rare: $[0, 1] : 10$ years” and “Unlikely: $[2, 5] : 10$ years”).
- The risk $R10$ becomes acceptable if the likelihood changes from “Possible” to “Unlikely” or “Rare.” The likelihood will change in such a way if the composite key indicator $K_{SP-EPR-INFO}$, monitoring the likelihood, is contained in the interval $[0, 5]$ (interval capturing both “Rare: $[0, 1] : 10$ years” and “Unlikely: $[2, 5] : 10$ years”).
- The risk $R12$ becomes acceptable if the likelihood changes from “Unlikely” to “Rare.” The likelihood will change in such a way if the composite key indicator $K_{HSP-EPR-INFO}$, monitoring the likelihood, is contained in the interval $[0, 1]$ (interval capturing “Rare: $[0, 1] : 10$ years”).
- The risk $R16 \& R18$ becomes acceptable if the likelihood changes from “Unlikely” to “Rare.” The likelihood will change in such a way if the composite key indicator $K_{ILL-ACC-SC}$, monitoring the likelihood, is contained in the interval $[0, 1]$ (interval capturing “Rare: $[0, 1] : 10$ years”).

Moreover, the different composite key indicators need to measure the likelihoods correctly in order to measure the fulfillment of PBO-A8. This can be determined based on the requirements to the different composite key indicators. These requirements are captured by $Req(K_{PR-SP-EPR-INFO})$, $Req(K_{PR-HSP-EPR-INFO})$, etc.

The reformulated precise business objective can also be used to determine to what degree the precise business objective is fulfilled. For instance, if $K_{PR-SP-EPR-INFO}$ equals 20 while the other composite key indicators equal 0, then PBO-A8 is close to being fulfilled. On the other hand, if $K_{PR-SP-EPR-INFO}$ equals 25 instead, then PBO-A8 is far from being fulfilled.

B. Evaluate criteria for internal validity (Step 4.2 of ValidKI)

To evaluate the internal validity of the set of key indicators, we need to show that the reformulated precise business objective PBO-A8' measures the fulfillment of the precise business objective PBO-A8. We evaluate the internal validity of each composite key indicator based on the criteria given in Section III-D.

To evaluate attribute validity we need to compare the definitions of the six risks with the requirements to the composite key indicators. The definitions of the risks $R8$, $R10$, and $R12$ are given in Figs. 7 and 8, while the definitions of the accumulated risks $R1 \& R2$, $R3 \& R4$, and $R16 \& R18$ are given in Section V-C. Moreover, the

requirements to the composite key indicators are given by $Req(K_{PR-SP-EPR-INFO})$, $Req(K_{PR-HSP-EPR-INFO})$, etc. In all six cases there is a match between the definition of the risk and the requirements to the composite key indicator. We therefore conclude that the composite key indicators correctly exhibit the likelihood attributes of the six risks that the composite key indicators intend to measure. In addition, based on the requirements specified for the six composite key indicators it is clear that the six composite key indicators are not restricted to only producing values that are always contained or not contained in the intervals mentioned above. Thus, both acceptable and unacceptable risks can be detected.

Moreover, all the composite key indicators have factor independence. Each composite key indicator is calculated based on three basic key indicators. These are independent of each other, since they are computed by three different health-care institutions. The six composite key indicators do also have internal consistency, since the three basic key indicators employed by each composite key indicator measure the same thing, but at different health-care institutions. The three basic key indicators are therefore conceptually related.

We continue the evaluation of internal validity by evaluating whether the composite key indicators have appropriate continuity. All are discontinuous if “*Number of years since the monitoring started*” equals zero. Client H does not consider this to be a problem, since the denominator will in all six cases be a real number that is never zero. We also show that the six composite key indicators have dimensional consistency. Each composite key indicator adds three likelihoods, where each is for the period of “*Number of years since the monitoring started*” years, and transforms the resulting likelihood into a likelihood which is for a period of ten years. Thus, no information is lost when constructing the composite key indicators from their respective basic key indicators. The six composite key indicators do also have unit validity. All six use the unit “likelihood per ten years,” which is appropriate for measuring the six likelihood attributes of the risks.

Based on the evaluation of the different internal validity types of criteria above, we conclude that the set of key indicators is internally valid. When the precise business objective PBO-A8 is fulfilled, we get the risk evaluation matrix in Table XI. In this situation, all of the risks $R1\&R2$, $R3\&R4$, $R8$, $R10$, $R12$, and $R16\&R18$ are acceptable. Moreover, the risks will have the following likelihood values when acceptable:

- The risk $R1\&R2$ will either have the likelihood “Rare” ($R1\&R2'$), “Unlikely” ($R1\&R2''$), or “Possible” ($R1\&R2'''$).
- The risk $R3\&R4$ will either have the likelihood “Rare” ($R3\&R4'$) or “Unlikely” ($R3\&R4''$).
- The risk $R8$ will either have the likelihood “Rare” ($R8'$) or “Unlikely” ($R8''$).
- The risk $R10$ will either have the likelihood “Rare” ($R10'$) or “Unlikely” ($R10''$).
- The risk $R12$ will have the likelihood “Rare”.
- The risk $R16\&R18$ will have the likelihood “Rare”.

VIII. SPECIFY KEY INDICATOR DESIGNS

We use the UML [6] sequence diagram notation for the key indicator design specifications, but one may of course also use other languages depending on the problem in question. In the following sub-sections, we specify the designs of the six composite key indicators and their respective basic key indicators.

A. Key indicator designs for $K_{PR-SP-EPR-INFO}$ and its basic key indicators

The sequence diagram in Fig. 11 specifies how the key indicator $K_{PR-SP-EPR-INFO}$ is calculated. Each entity in the sequence diagram is either a component, a sensor, or an employee at Client H, and it is represented by a dashed, vertical line called a lifeline, where the box at its top specifies which entity the lifeline represents. The entities interact with each other through the transmission and reception of messages, which are shown as horizontal arrows from the transmitting lifeline to the receiving lifeline. We can also see that a lifeline can be both the sender and receiver of a message.

The sequence diagram contains one reference (ref) to another sequence diagram. This reference can be replaced by the content of the sequence diagram that it refers to. The reference refers to the sequence diagram given in Fig. 12, which describes the calculation of the basic key indicator $K_{CH-PR-SP-EPR-INFO}$ at Client H. We do not present sequence diagrams describing the calculations of the two other basic key indicators, since these calculations are performed in the same way as the calculation of $K_{CH-PR-SP-EPR-INFO}$, and since these calculations involve the same

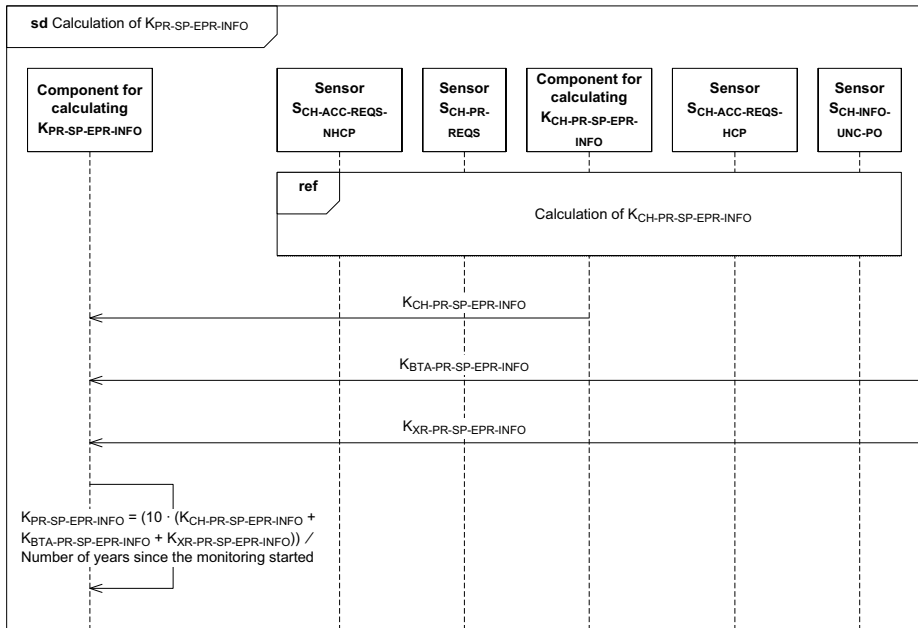


Fig. 11. The sequence diagram “Calculation of $K_{PR-SP-EPR-INFO}$ ”

types of lifelines as the ones described in Fig. 12. For the two other basic key indicators we only show that they are sent to “Component for calculating $K_{PR-SP-EPR-INFO}$,” and that they are used in the calculation of $K_{PR-SP-EPR-INFO}$.

The sequence diagram in Fig. 12 shows that the basic key indicator $K_{CH-PR-SP-EPR-INFO}$ is updated each week. The first thing that happens is that “Component for calculating $K_{CH-PR-SP-EPR-INFO}$ ” retrieves the value that was computed for the basic key indicator in the previous week. Afterwards, the component counts for each printout the number of health-care professionals and employees that are not health-care professionals that accessed the printer room between $TIME_1$ (the time the print job was completed) and $TIME_2$ (the time when the health-care professional collected his/hers printout or the time when the printout was collected by a security employee). The number NUM is the number of other health-care professionals and employees that are not health-care professionals that may have seen the printout of sensitive and private information.

Client H is of the opinion that between 10% and 30% of the other health-care professionals and employees that are not health-care professionals that accessed the printer rooms between $TIME_1$ and $TIME_2$ have seen the printouts of sensitive and private information from patients’ EPRs. Thus, the number $TOTAL_NUM$ is multiplied by $[0.1, 0.3]$. In the end, the component stores the basic key indicator before sending it to “Component for calculating $K_{PR-SP-EPR-INFO}$,” as illustrated in the sequence diagram in Fig. 11.

B. Key indicator designs for $K_{PR-HSP-EPR-INFO}$ and its basic key indicators

The sequence diagram in Fig. 13 specifies how the key indicator $K_{PR-HSP-EPR-INFO}$ is calculated, while the sequence diagram in Fig. 14 describes the calculation of the basic key indicator $K_{CH-PR-HSP-EPR-INFO}$ at Client H. We use the same argument as the one given in Section VIII-A for not presenting sequence diagrams for the two other basic key indicators.

The sequence diagram in Fig. 14 shows that the basic key indicator $K_{CH-PR-HSP-EPR-INFO}$ is updated each week. This sequence diagram is almost identical to the one in Fig. 12. Thus, we do not give any further explanations for the sequence diagram.

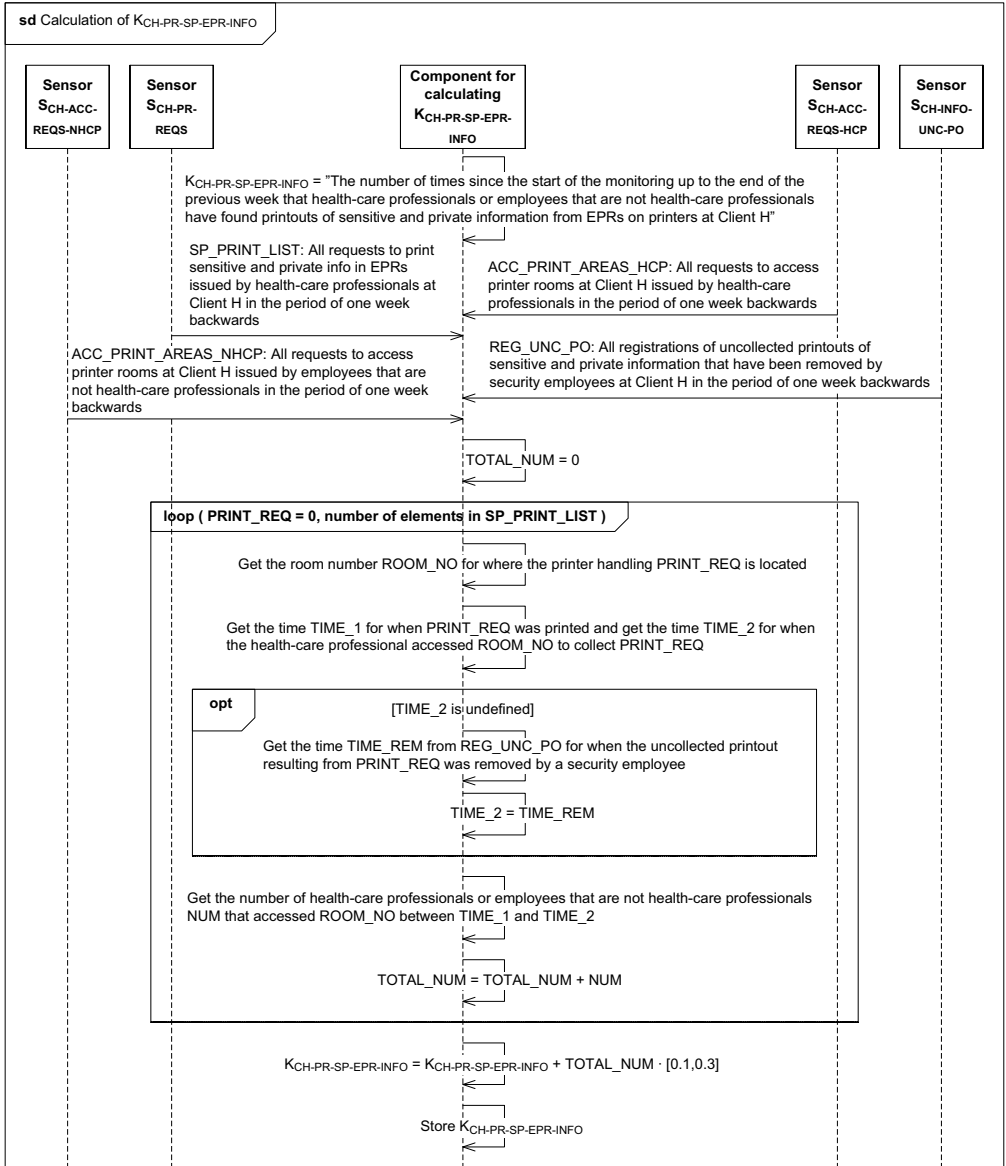


Fig. 12. The sequence diagram "Calculation of $K_{CH-PR-SP-EPR-INFO}$ "

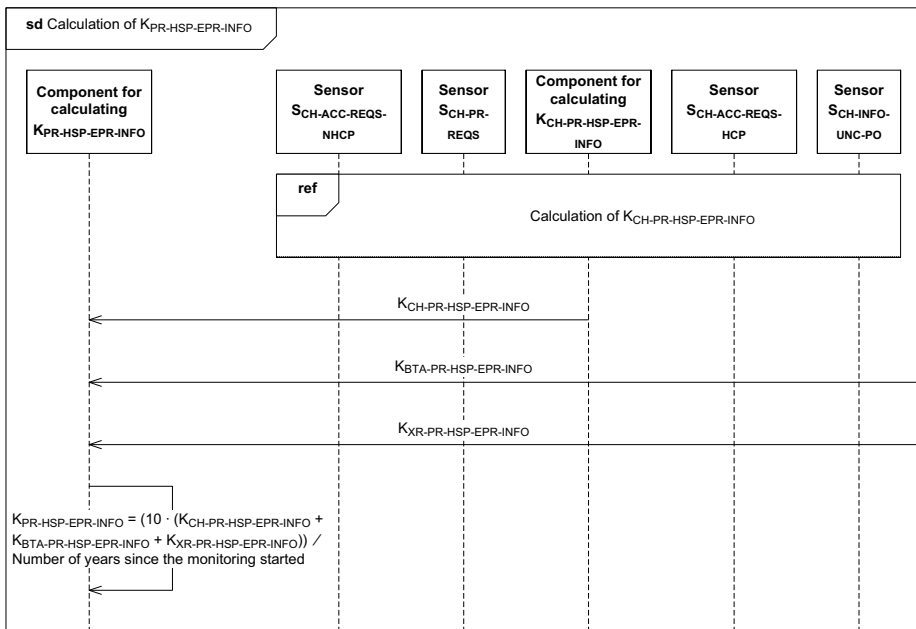


Fig. 13. The sequence diagram “Calculation of $K_{PR-HSP-EPR-INFO}$ ”

C. Key indicator designs for $K_{NOT-APP-UNAUTH-ACC}$ and its basic key indicators

The sequence diagram in Fig. 15 specifies how the key indicator $K_{NOT-APP-UNAUTH-ACC}$ is calculated, while the sequence diagram in Fig. 16 describes the calculation of the basic key indicator $K_{CH-NOT-APP-UNAUTH-ACC}$ at Client H. We use the same argument as the one given in Section VIII-A for not presenting sequence diagrams for the two other basic key indicators.

The sequence diagram in Fig. 16 shows that the basic key indicator $K_{CH-NOT-APP-UNAUTH-ACC}$ is updated each week. The first thing that happens is that “Component for calculating $K_{CH-NOT-APP-UNAUTH-ACC}$ ” sends the value that was computed for the basic key indicator in the previous week to “Employee at Client H.” Afterwards, the component identifies “All unauthorized accesses at Client H in the period of one week backwards to highly sensitive and private information in EPRs, where the owners of the EPRs are not patients of the accessors” based on input from the entities representing the sensors. The “Employee at Client H” performs a manual inspection of each of these unauthorized accesses, and classifies each as approved or not approved. If the unauthorized access is classified as not approved, then the basic key indicator is incremented by one. After all the unauthorized accesses have been inspected and classified, “Employee at Client H” sends the basic key indicator to the component which stores it. Afterwards, the component sends the basic key indicator to “Component for calculating $K_{NOT-APP-UNAUTH-ACC}$,” as illustrated in the sequence diagram in Fig. 15.

D. Key indicator designs for $K_{SP-EPR-INFO}$ and its basic key indicators

The sequence diagram in Fig. 17 specifies how the key indicator $K_{SP-EPR-INFO}$ is calculated, while the sequence diagram in Fig. 18 describes the calculation of the basic key indicator $K_{CH-SP-EPR-INFO}$ at Client H. We use the same argument as the one given in Section VIII-A for not presenting sequence diagrams for the two other basic key indicators.

The sequence diagram in Fig. 18 shows that the basic key indicator $K_{CH-SP-EPR-INFO}$ is updated each week. The first thing that happens is that “Component for calculating $K_{CH-SP-EPR-INFO}$ ” sends the value that was computed for the basic key indicator in the previous week to “Employee at Client H”. Afterwards, the component receives

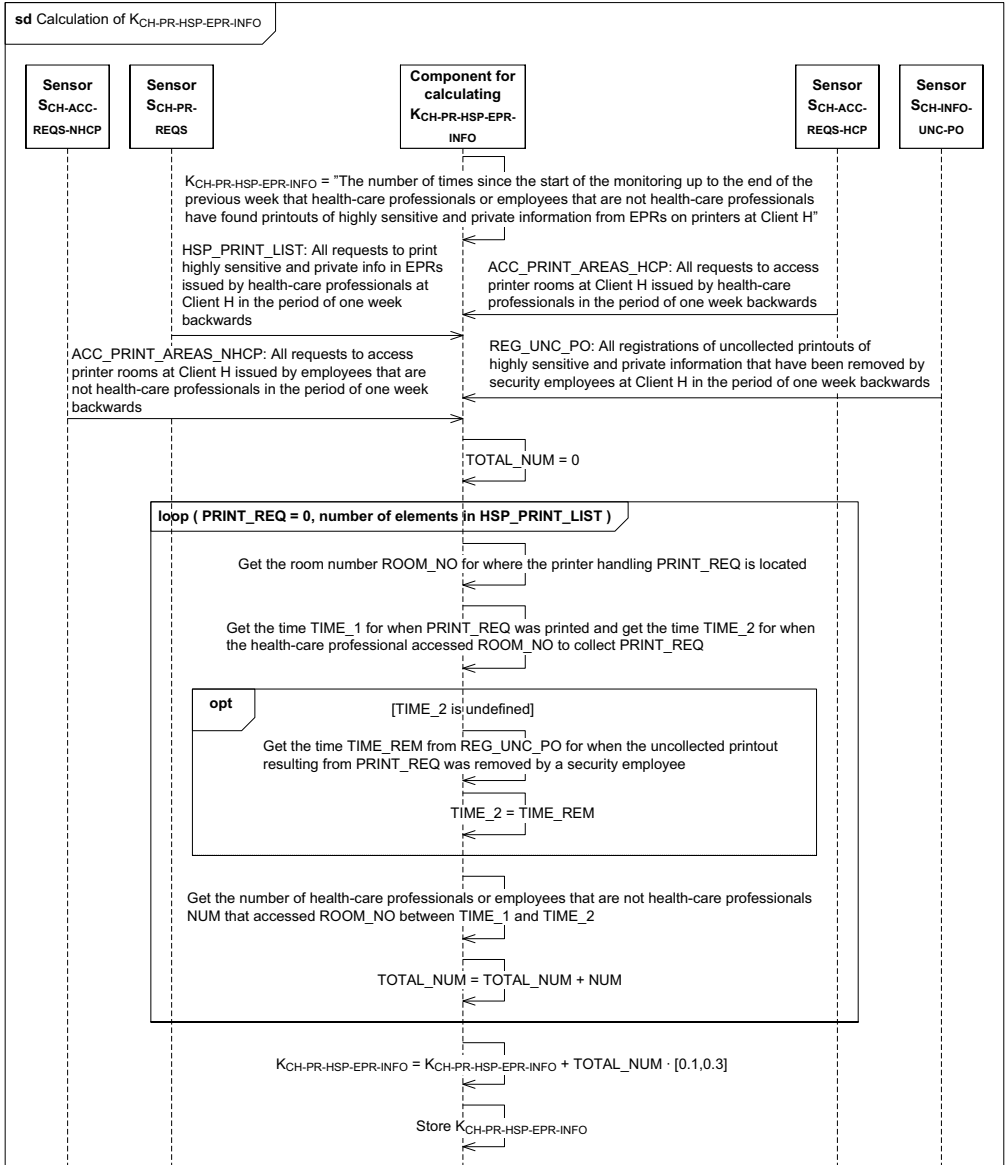


Fig. 14. The sequence diagram "Calculation of $K_{CH-PR-HSP-EPR-INFO}$ "

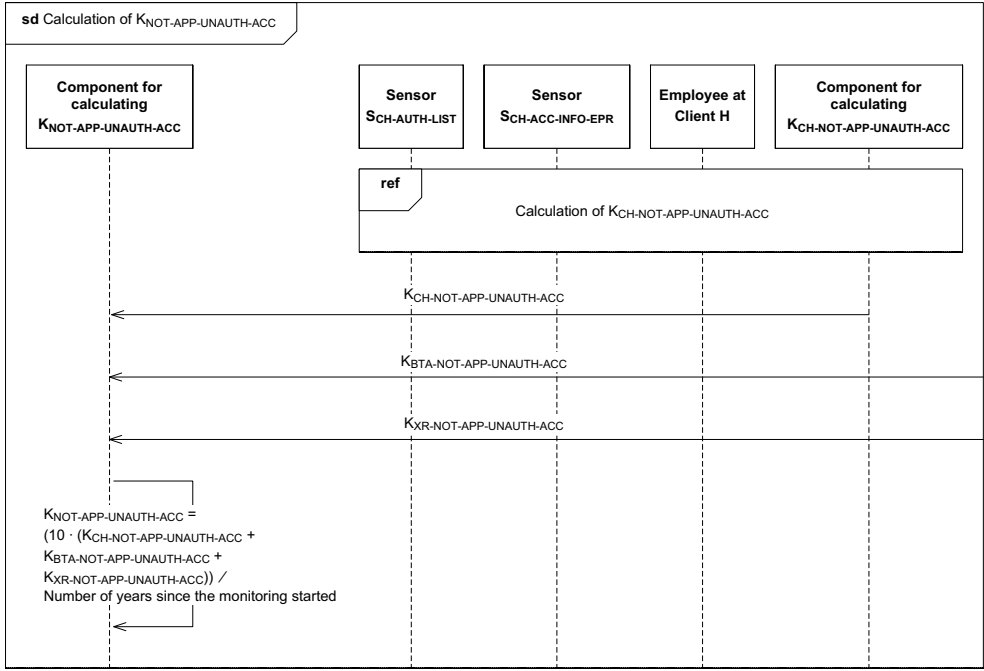


Fig. 15. The sequence diagram “Calculation of $K_{NOT-APP-UNAUTH-ACC}$ ”

different kinds of data from the three sensors, where this data is used in the sequence diagram “Comparison of data” in Fig. 19 for updating $K_{CH-SP-EPR-INFO}$.

In the sequence diagram in Fig. 19, “Component for calculating $K_{CH-SP-EPR-INFO}$ ” extracts all accesses to sensitive and private information in EPRs that have occurred in the period of one week backwards. The component also extracts all information items from `INFO_LIST_1` and `INFO_LIST_2` that both refer to Client H and the medical history of a person. Since information retrieved from the traditional media or the Internet will refer to patients by name, the different accesses are grouped with respect to patient names. In addition, duplicate accesses are removed, since we are not interested in how many times some information has been accessed, but rather whether it has been accessed or not. As can be seen in the sequence diagram, the different items of information retrieved from the traditional media or the Internet are grouped in the same way as for accesses to information in EPRs.

After having grouped the different data, we check for the different information items whether they match information that is retrieved when performing different accesses to information in EPRs. We use software to identify potential matches, while an employee at Client H performs a manual check of the potential matches to determine whether the sensitive and private information obtained from performing an access to information in an EPR is really the source of the information that has been retrieved from the traditional media or the Internet. When evaluating the potential matches, the employee needs to consider other potential sources for the information leakage, such as the patient itself. The employee also needs to consider whether the information retrieved from the traditional media or the Internet really refers to the same patient as the information obtained from an EPR does. If the employee is confident that the information from the EPR is the source, then the basic key indicator $K_{CH-SP-EPR-INFO}$ is incremented by one. In the end, the employee sends the updated basic key indicator to “Component for calculating $K_{CH-SP-EPR-INFO}$,” as illustrated in Fig. 18. The component stores the updated basic key indicator before sending it to “Component for calculating $K_{SP-EPR-INFO}$,” as illustrated in the sequence diagram in Fig. 17.

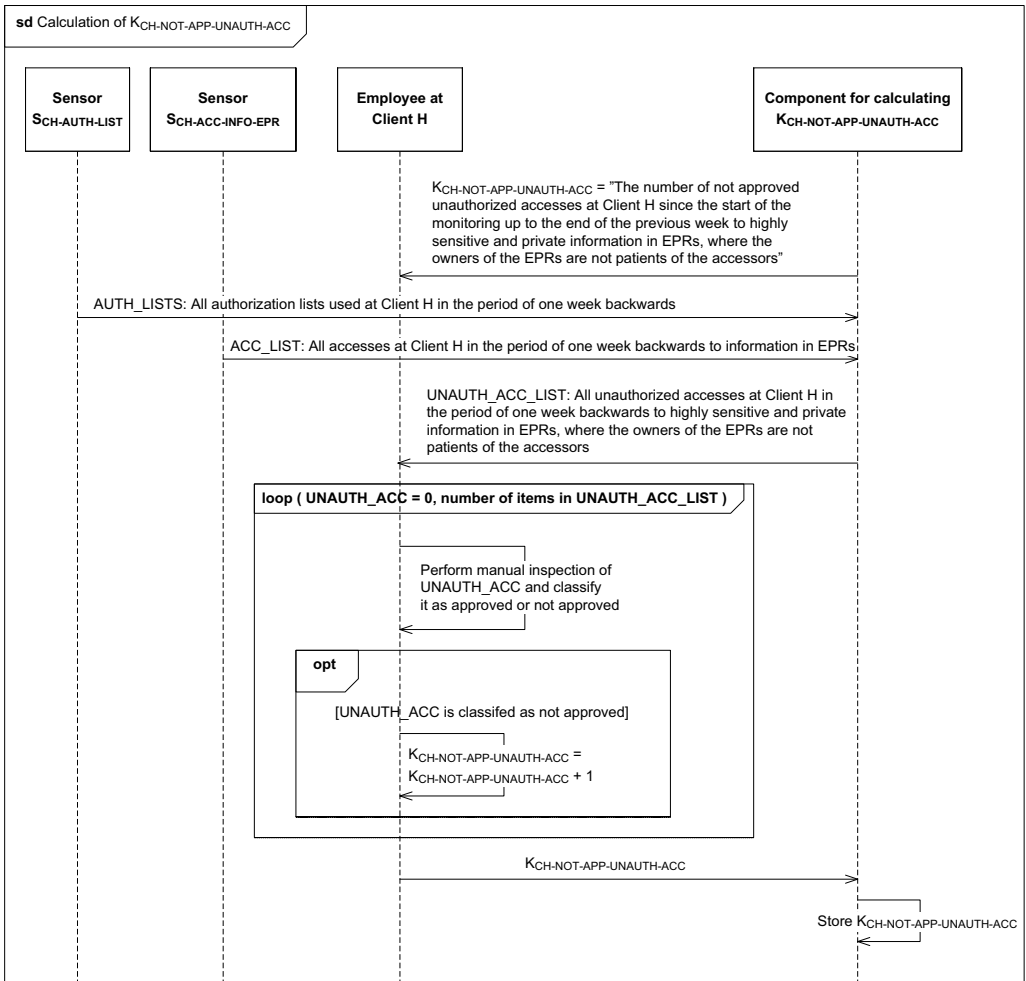


Fig. 16. The sequence diagram “Calculation of $K_{CH-NOT-APP-UNAUTH-ACC}$ ”

E. Key indicator designs for $K_{HSP-EPR-INFO}$ and its basic key indicators

The sequence diagram in Fig. 20 specifies how the key indicator $K_{HSP-EPR-INFO}$ is calculated, while the sequence diagram in Fig. 21 describes the calculation of the basic key indicator $K_{CH-HSP-EPR-INFO}$ at Client H. We use the same argument as the one given in Section VIII-A for not presenting sequence diagrams for the two other basic key indicators.

The sequence diagram in Fig. 21 shows that the basic key indicator $K_{CH-HSP-EPR-INFO}$ is updated each week. This sequence diagram is almost identical to the one in Fig. 18, while the sequence diagram “Comparison of data” in Fig. 22, which is referred to in Fig. 21, is almost identical to the one in Fig. 19. Thus, we do not give any further explanations for the two sequence diagrams.

E. Key indicator designs for $K_{ILL-ACC-SC}$ and its basic key indicators

The sequence diagram in Fig. 23 specifies how the key indicator $K_{ILL-ACC-SC}$ is calculated, while the sequence diagram in Fig. 24 describes the calculation of the basic key indicator $K_{CH-ILL-ACC-SC}$ at Client H. We use the same

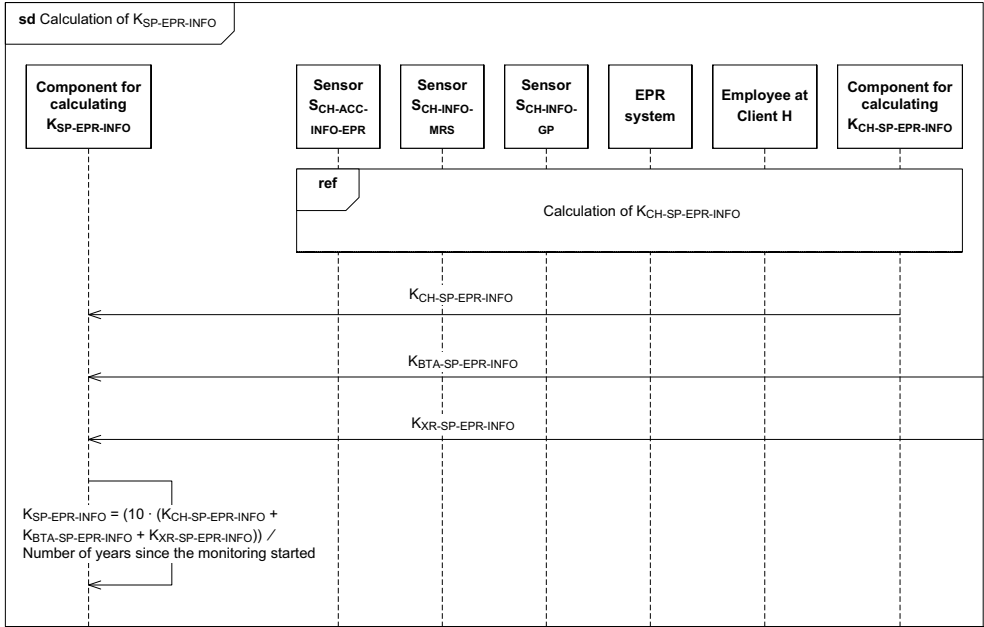


Fig. 17. The sequence diagram “Calculation of $K_{SP-EPR-INFO}$ ”

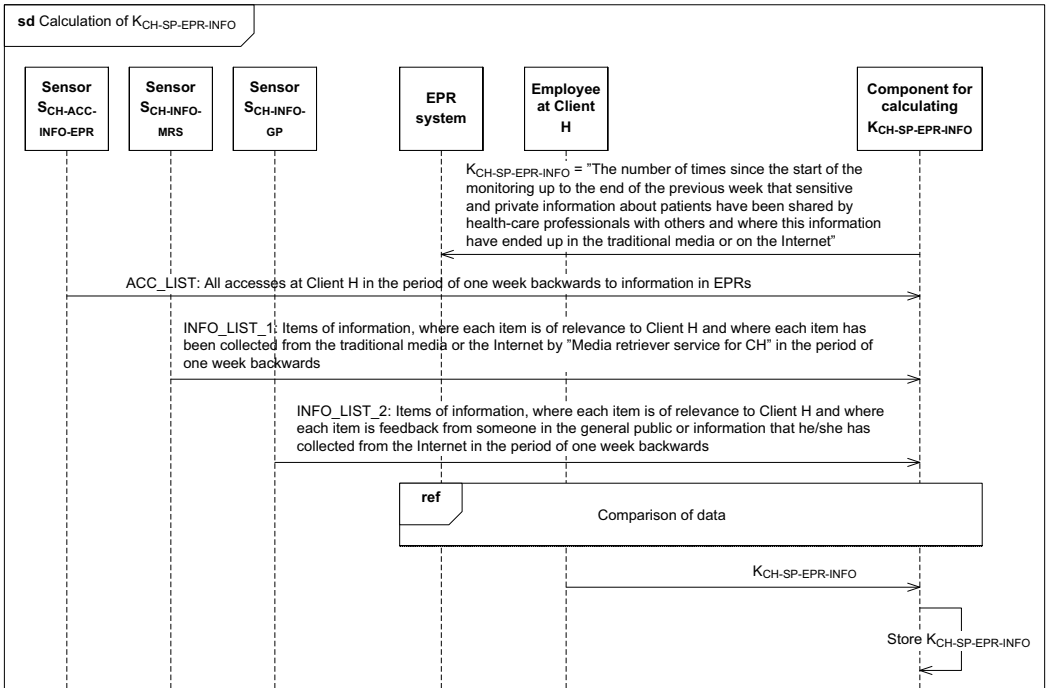


Fig. 18. The sequence diagram “Calculation of $K_{CH-SP-EPR-INFO}$ ”

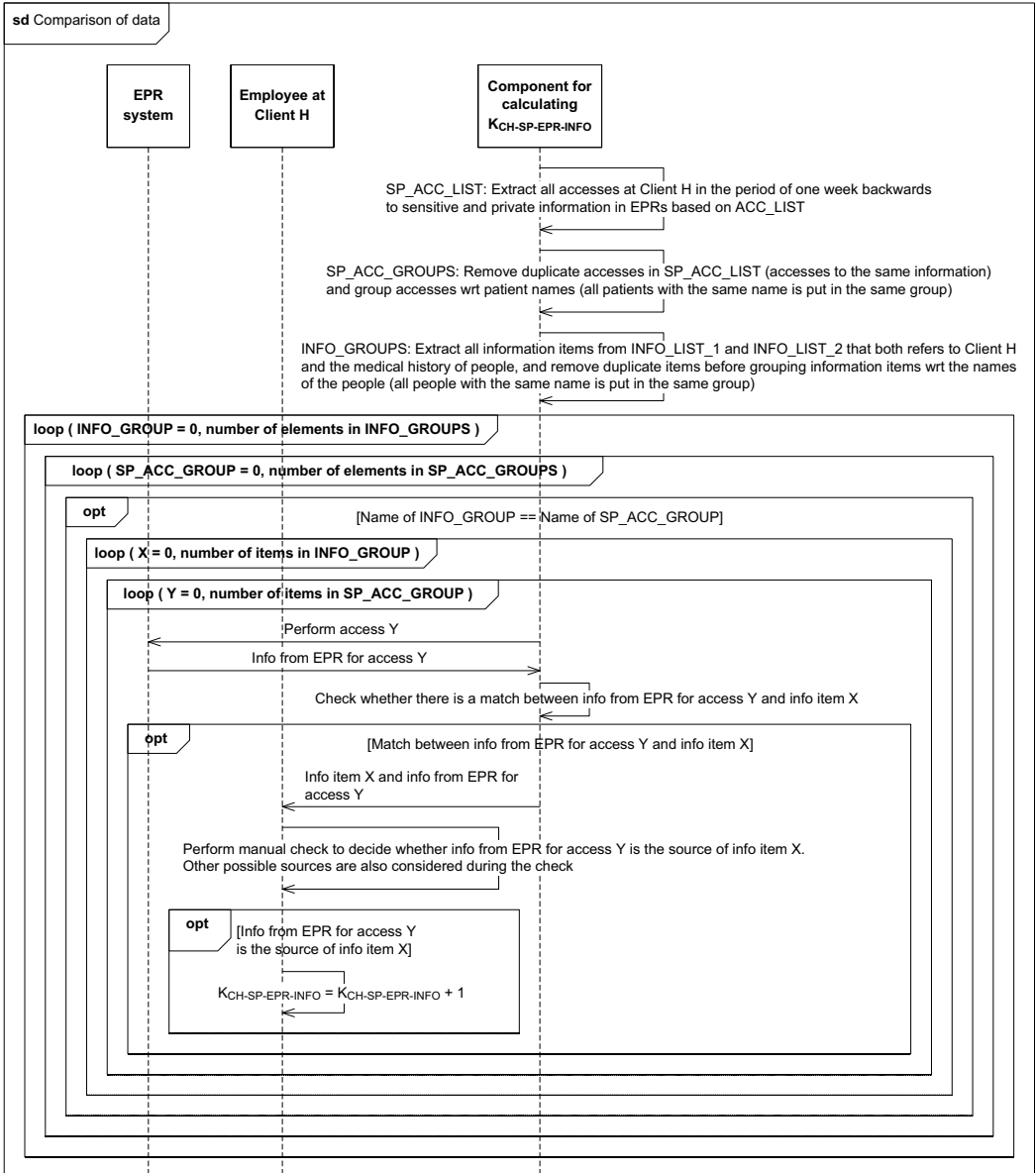


Fig. 19. The sequence diagram “Comparison of data”

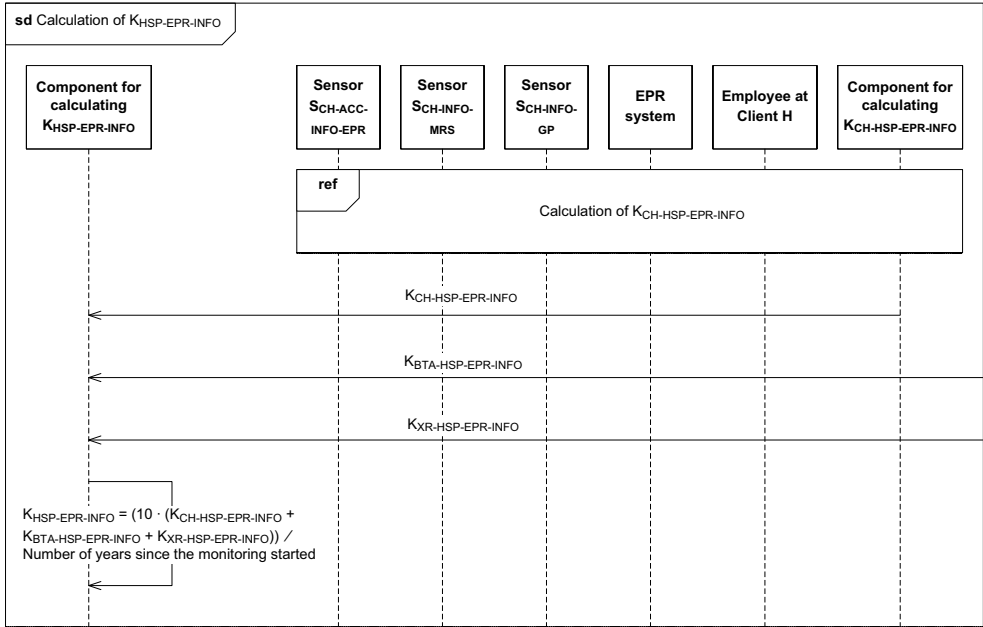


Fig. 20. The sequence diagram “Calculation of $K_{HSP-EPR-INFO}$ ”

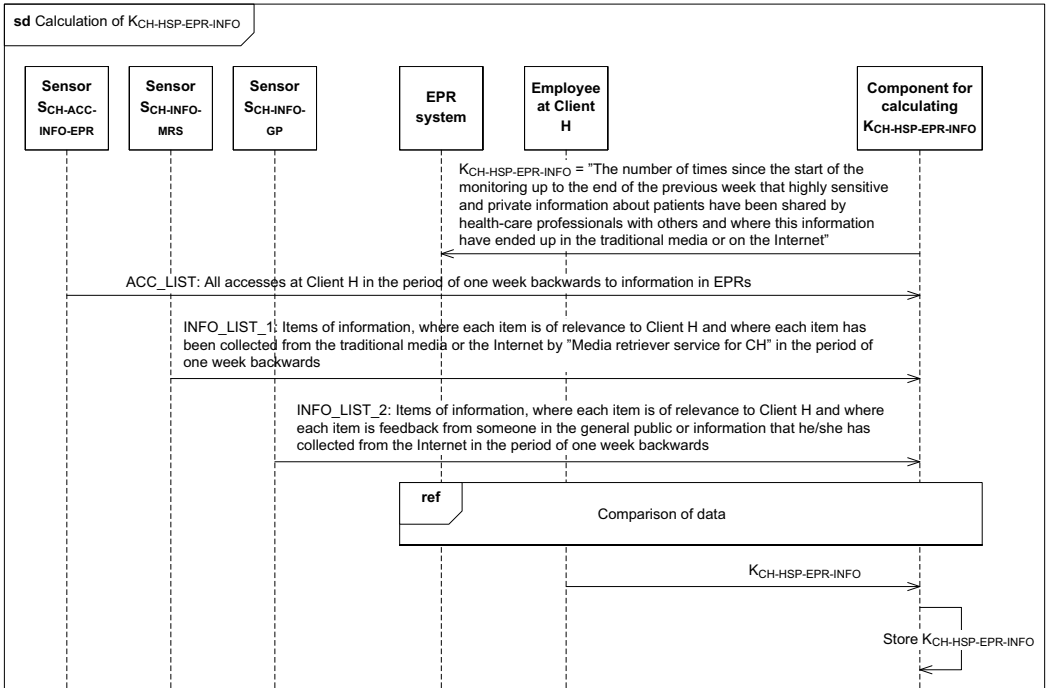


Fig. 21. The sequence diagram “Calculation of $K_{CH-HSP-EPR-INFO}$ ”

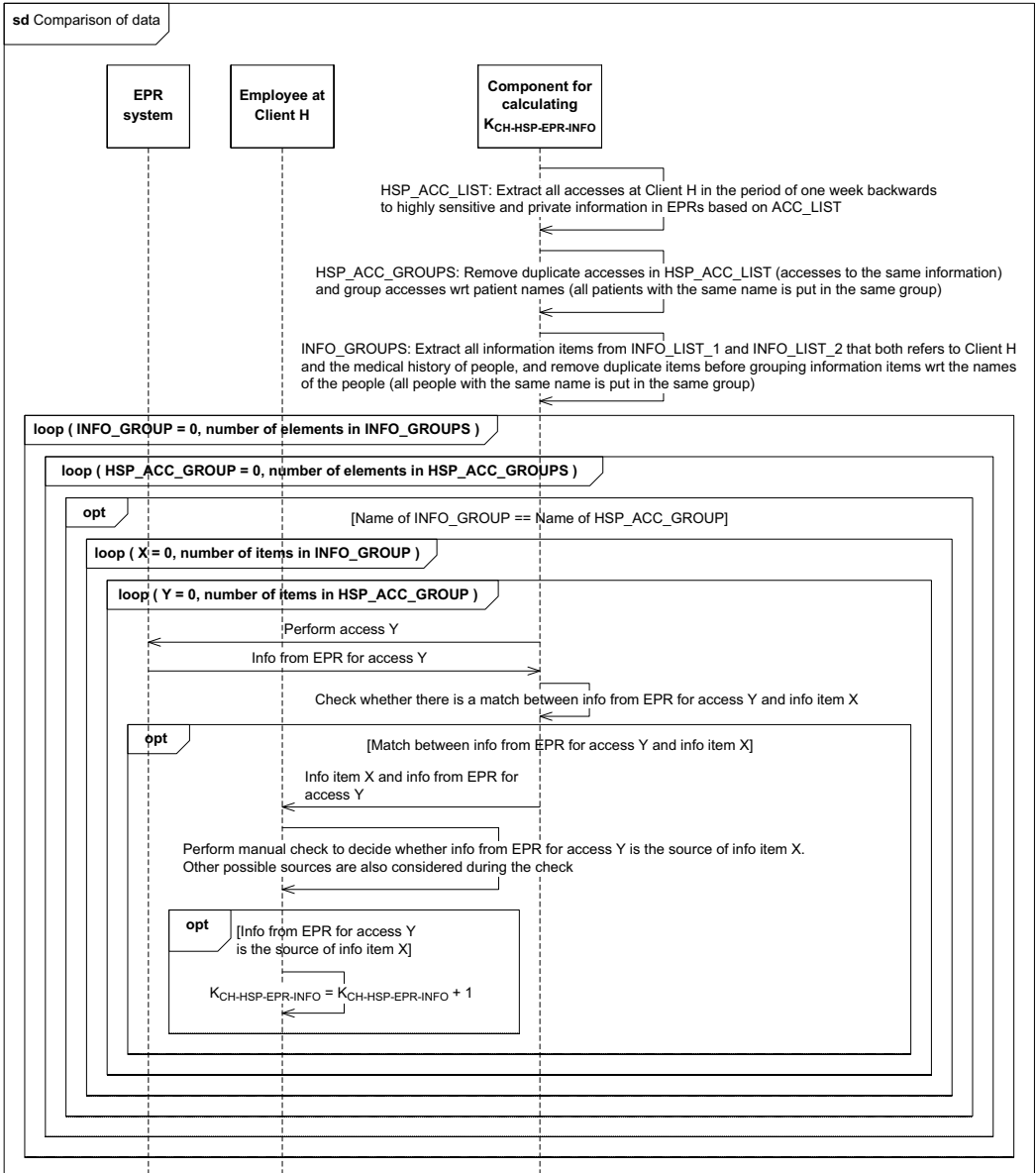


Fig. 22. The sequence diagram “Comparison of data”

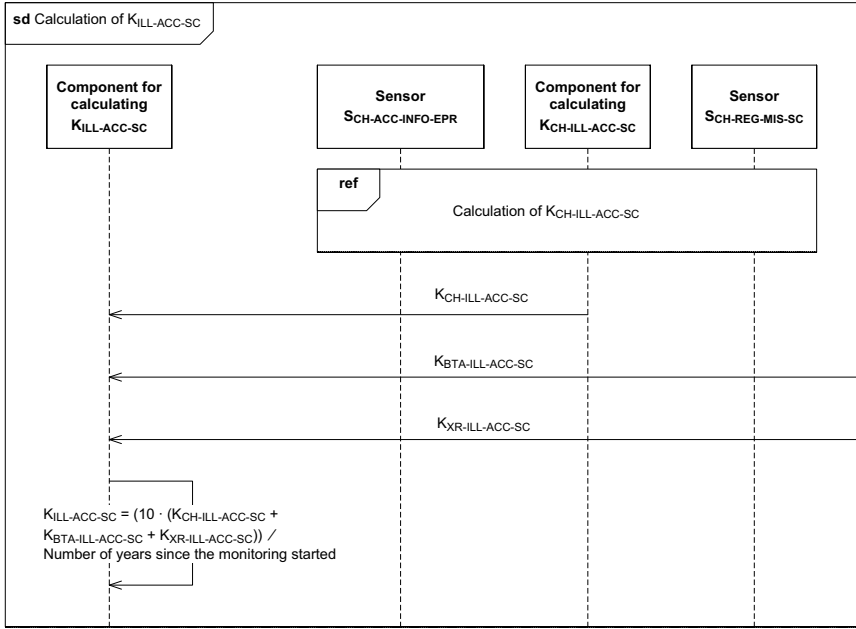


Fig. 23. The sequence diagram “Calculation of $K_{ILL-ACC-SC}$ ”

argument as the one given in Section VIII-A for not presenting sequence diagrams for the two other basic key indicators.

The sequence diagram in Fig. 24 shows that the basic key indicator $K_{CH-ILL-ACC-SC}$ is updated each week. The first thing that happens is that “Component for calculating $K_{CH-ILL-ACC}$ ” retrieves the value that was computed for the basic key indicator in the previous week. Afterwards, the component counts for each of the lost/stolen smart cards the number of accesses that have occurred between T_{IME_1} (the time the smart card’s owner used it the last time before noticing that it was missing) and T_{IME_2} (the time when the smart card was registered as missing). In the end, the component stores the basic key indicator $K_{CH-ILL-ACC-SC}$, and sends it to “Component for calculating $K_{ILL-ACC-SC}$,” as illustrated in the sequence diagram in Fig. 23.

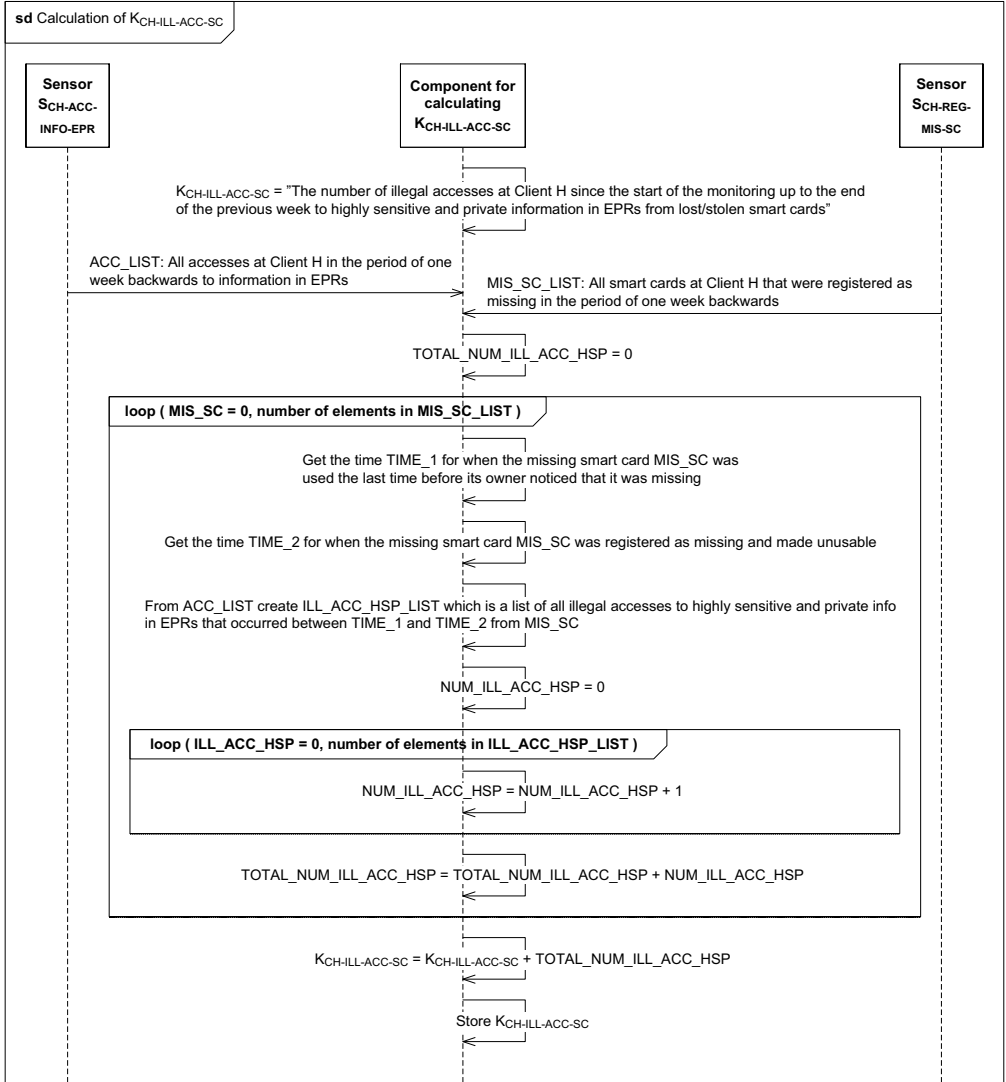


Fig. 24. The sequence diagram "Calculation of $K_{CH-ILL-ACC-SC}$ "

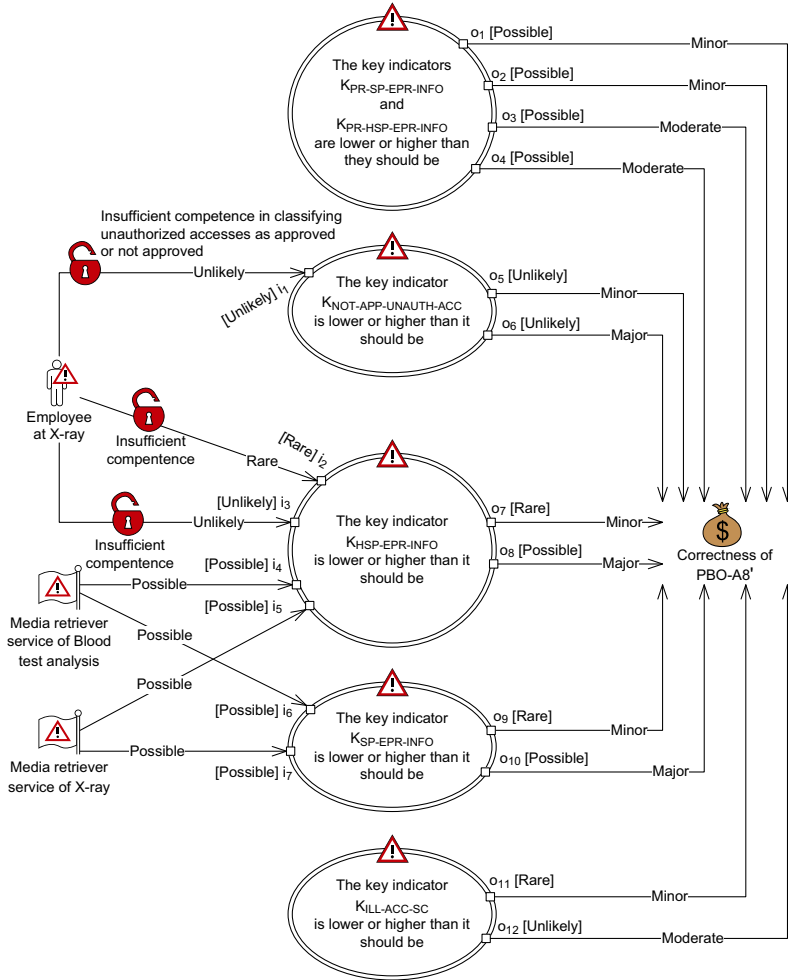
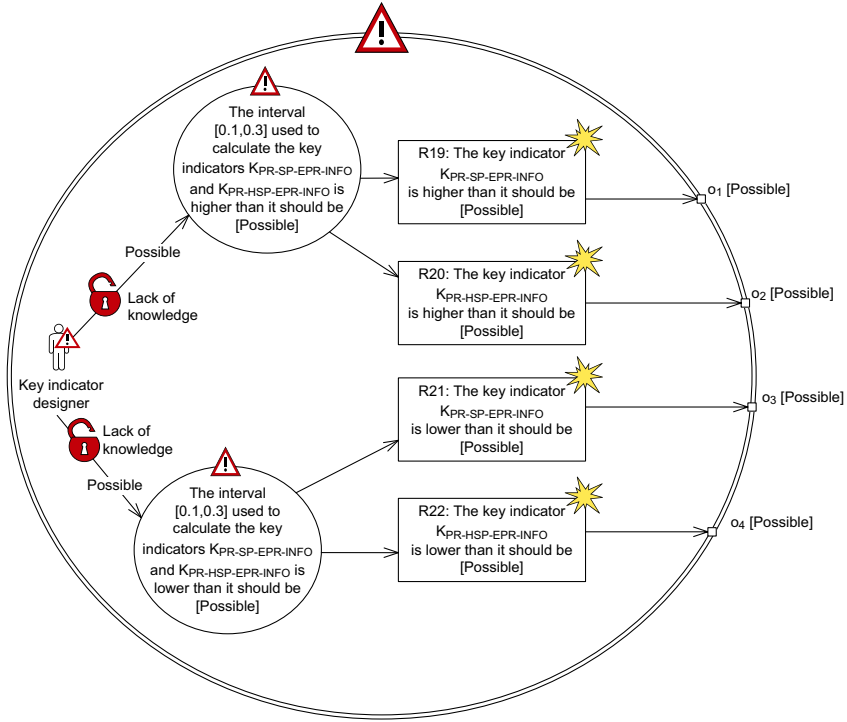


Fig. 25. CORAS threat diagram providing a high-level overview of the impact of the proposed implementation of the monitoring infrastructure for the different composite key indicators on the correctness of PBO-A8'

IX. EVALUATE CONSTRUCT VALIDITY

To evaluate whether the composite key indicators have construct validity, we re-do the risk analysis from Step 2.2 with the asset “Fulfillment of PBO-A8” replaced by the asset “Correctness of PBO-A8’.” We have established that the monitoring infrastructure described in Step 2–4 is suitable for monitoring the relevant part of business. With the designs of the key indicators specified in the previous step, we want to identify in this step whether the proposed implementation of the monitoring infrastructure results in any new unacceptable risks. More precisely, we want to identify unacceptable risks towards the correctness of the reformulated precise business objective that are the result of threats to criteria for construct validity that the different composite key indicators need to fulfill.

We evaluate the construct validity of the composite key indicators based on the criteria given in Section III-F. A high-level overview of the result of the risk analysis is given in the CORAS threat diagram in Fig. 25. In the referenced threat scenarios in Figs. 26 – 30, risk to the correctness of the different composite key indicators have been documented. For the key indicators $K_{PR-SP-EPR-INFO}$ and $K_{PR-HSP-EPR-INFO}$, Client H is of the opinion that their



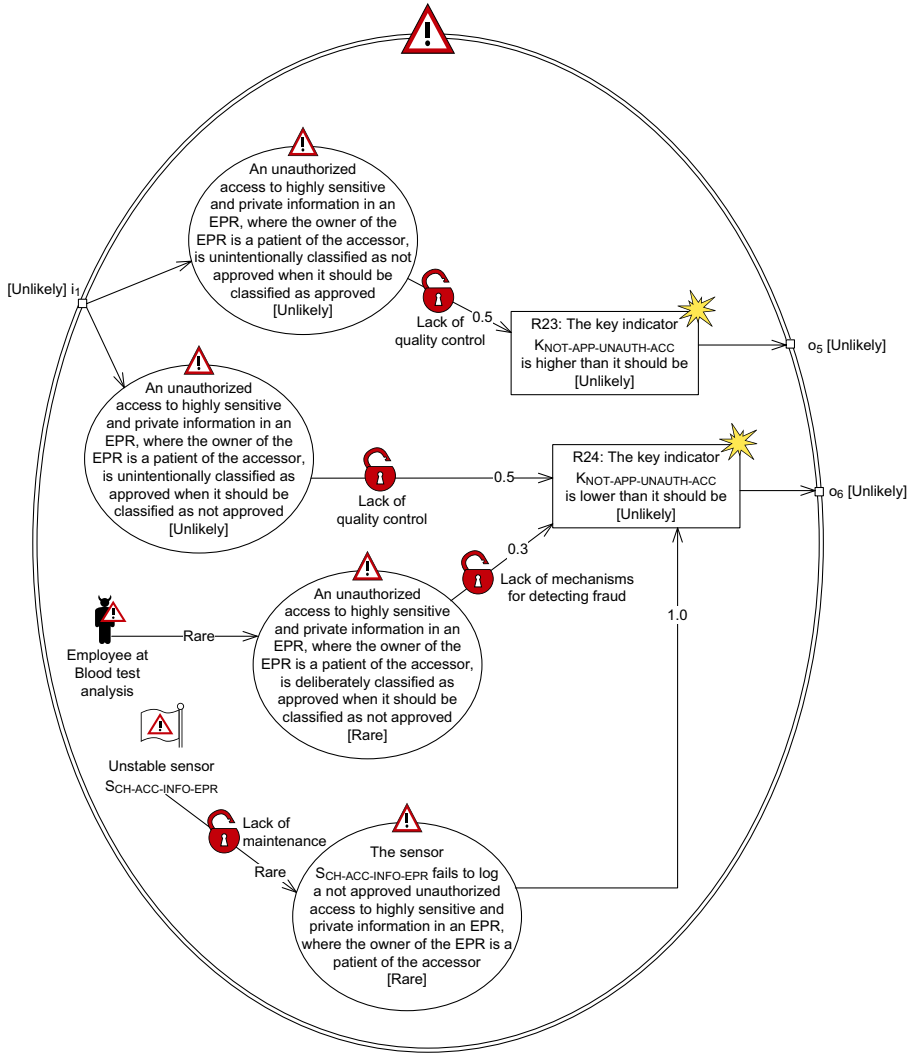
The key indicators $K_{PR-SP-EPR-INFO}$ and $K_{PR-HSP-EPR-INFO}$ are lower or higher than they should be

Fig. 26. The referenced threat scenario “The key indicators $K_{PR-SP-EPR-INFO}$ and $K_{PR-HSP-EPR-INFO}$ are lower or higher than they should be,” referred to in Fig. 25

correctness may be affected if the interval $[0.1, 0.3]$ used to calculate the two key indicators is either too low or too high. This is an example of violation of the stability criterion, since the selection of the interval is the result of human decisions, i.e., expert judgments. For the two composite key indicators, no threats towards the definition and instrument validity of the composite key indicators are identified.

In the case of the key indicator $K_{NOT-APP-UNAUTH-ACC}$, Client H is of the opinion that its correctness may be affected if the employees who classify unauthorized accesses as approved or not approved at X-ray and Blood test analysis are incompetent and fraudulent, respectively. Both these cases are examples of violation of the stability criterion, since the classification of unauthorized accesses as approved or not approved involves human decisions. Moreover, Client H is worried that the sensor $S_{CH-ACC-INFO-EPR}$ (represented as a non-human threat in Fig. 27) may be unstable with respect to logging of accesses to information in EPRs. This is an example of violation of the instrument validity criterion. Besides the stability and instrument validity criteria, definition validity should also be evaluated. In our case, we say that a key indicator has definition validity if its design is clear and unambiguous so that the key indicator can be implemented correctly. The only thing that is not clear and unambiguous with respect to the design of $K_{NOT-APP-UNAUTH-ACC}$ is how unauthorized accesses should be classified as approved or not approved. Since this has already been covered during the evaluation of the stability criterion, we do not pursue this issue further.

In the case of the key indicators $K_{SP-EPR-INFO}$ and $K_{HSP-EPR-INFO}$, Client H is worried that the correctness of $K_{SP-EPR-INFO}$ may be affected if employees at Blood test analysis either fail to identify data leakages of sensitive and private information from EPRs or incorrectly classify sensitive and private information obtained from EPRs as the sources of data leakages, when no such data leakages have occurred. Moreover, Client H is worried that



The key indicator $K_{NOT-APP-UNAUTH-ACC}$ is lower or higher than it should be

Fig. 27. The referenced threat scenario “The key indicator $K_{NOT-APP-UNAUTH-ACC}$ is lower or higher than is should be,” referred to in Fig. 25

the correctness of $K_{HSP-EPR-INFO}$ may be affected if employees at X-ray commit the same errors when it comes to highly sensitive and private information in EPRs. Both these cases are examples of violation of the stability criterion. In the case of instrument validity, Client H is worried that the media retriever services employed by Blood test analysis and X-ray are not able to collect the information necessary for detecting data leakages. Client H is also worried that the two composite key indicators may violate the definition validity criterion. The design specifications of the two composite key indicators are not clear and unambiguous with respect to how data leakages should be identified. In both specifications, it is up to the employees investigating potential data leakages to decide. Since this has already been covered during the evaluation of the stability criterion, we do not pursue this issue further.

In the case of the key indicator $K_{ILL-ACC-SC}$, Client H is worried that its correctness may be affected by health-

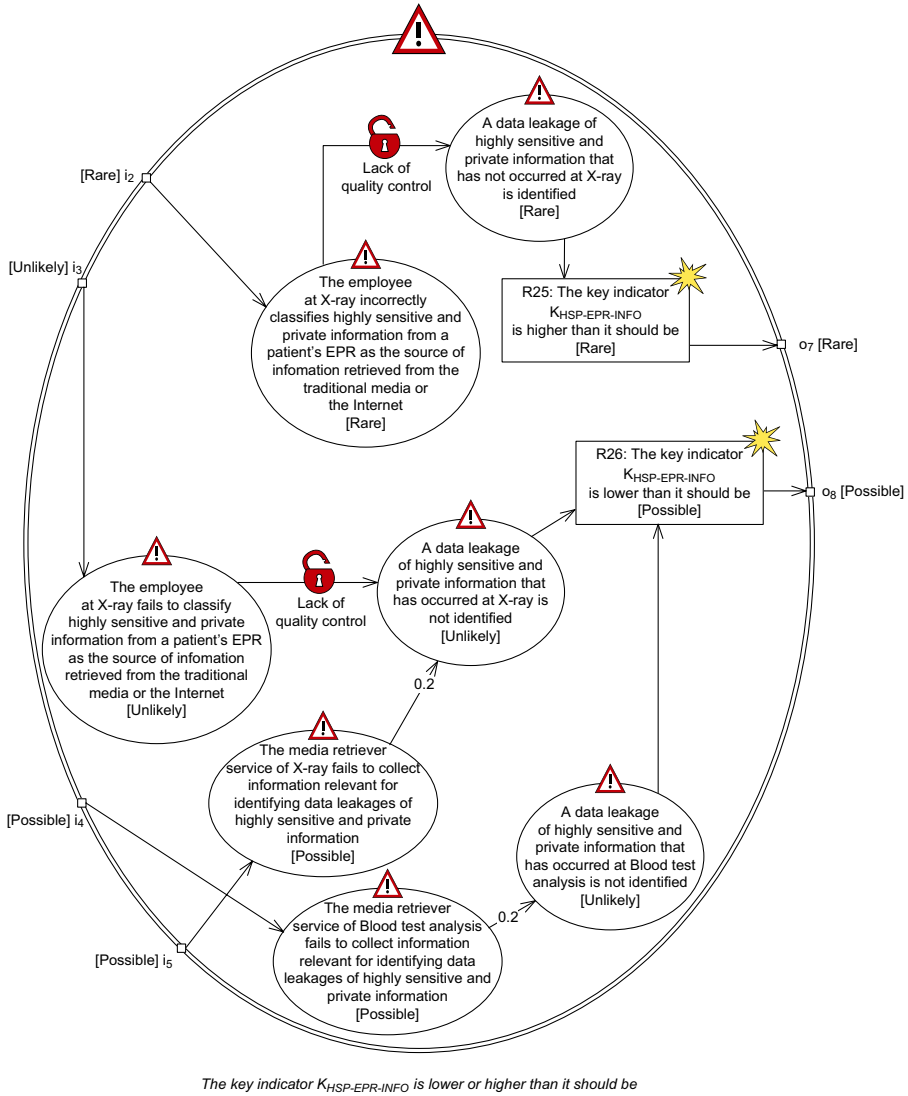
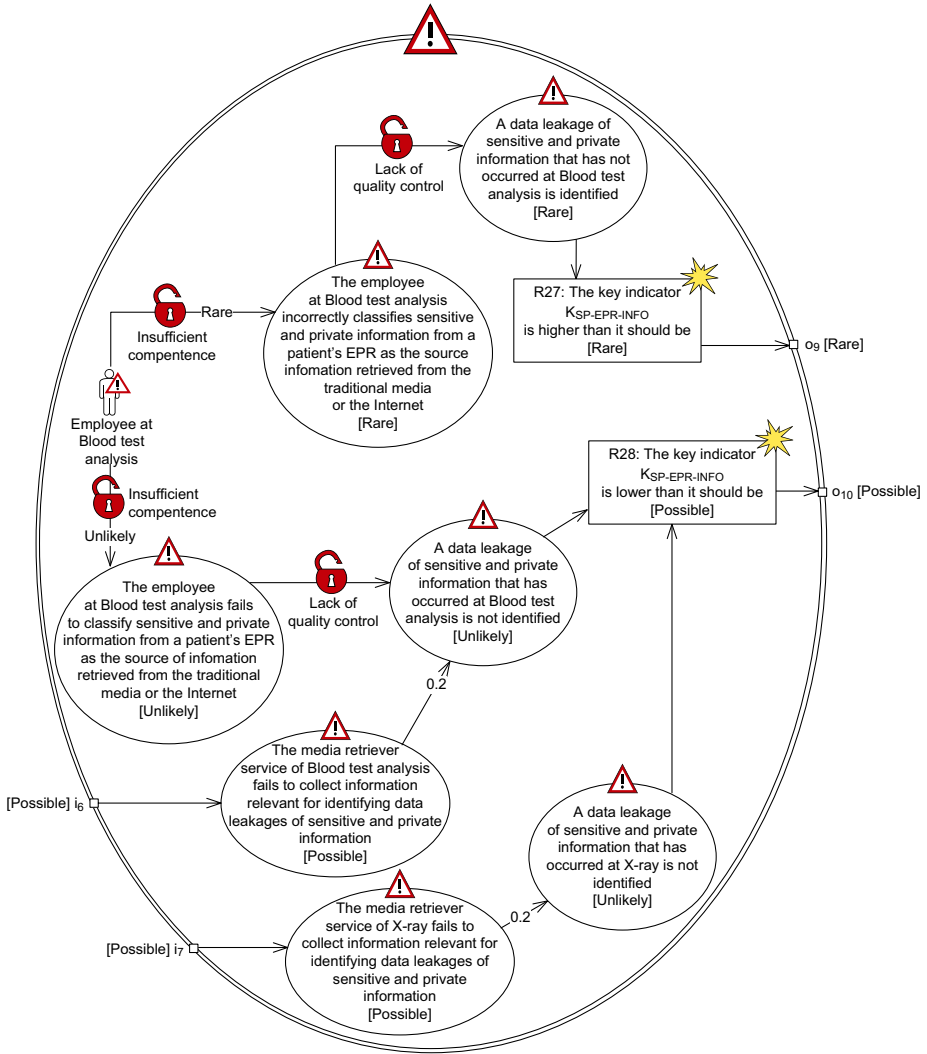


Fig. 28. The referenced threat scenario “The key indicator $K_{HSP-EPR-INFO}$ is lower or higher than is should be,” referred to in Fig. 25

care professionals not having a perfect recollection of when they used their smart cards the last time before losing it. By not having a perfect recollection, accesses to information in EPRs may incorrectly be classified as legal or illegal accesses. This is an example of violation of the stability criterion. For the composite key indicator, no threats towards the definition and instrument validity of the composite key indicator are identified.

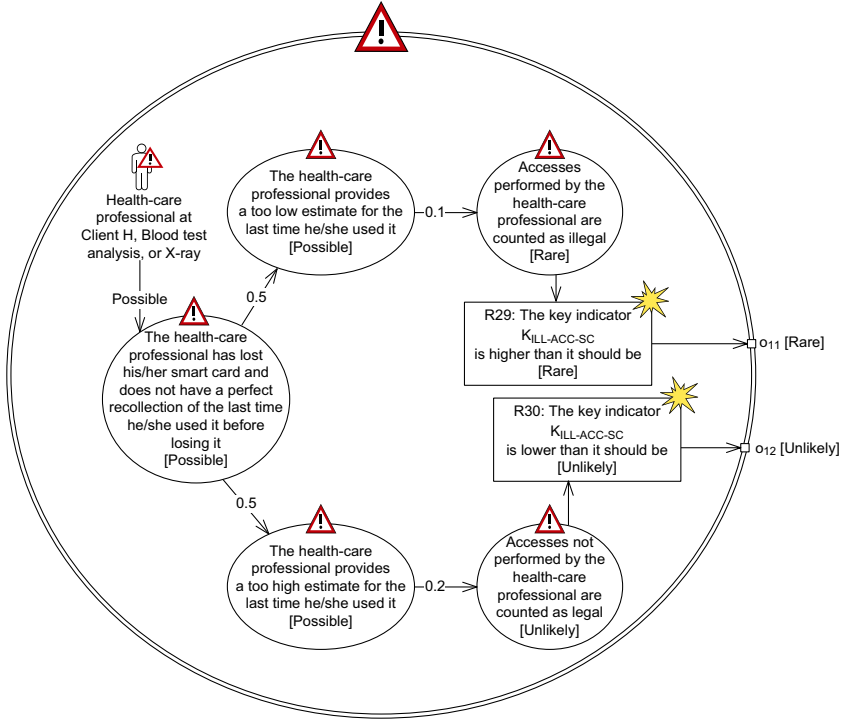
In Table XII the risks $R19 - R30$ have been plotted according to their likelihoods and consequences. As we can see from the table, the two risks $R26$ and $R28$ are unacceptable. This means that all the composite key indicators with the exceptions of $K_{SP-EPR-INFO}$ and $K_{HSP-EPR-INFO}$ have construct validity. As a first step to making these two risks acceptable, Client H finds it necessary to gain more knowledge on the suitability of the two media retriever services. If the two risks do not become acceptable as a result of this, further treatment will be necessary in order



The key indicator $K_{SP-EPR-INFO}$ is lower or higher than it should be

Fig. 29. The referenced threat scenario “The key indicator $K_{SP-EPR-INFO}$ is lower or higher than is should be,” referred to in Fig. 25

for the two key indicators $K_{SP-EPR-INFO}$ and $K_{HSP-EPR-INFO}$ to achieve construct validity. Such treatments may involve replacing the media retriever services of Blood test analysis and X-ray, or introducing an additional media retriever service for each of the two hospitals. In the latter case this means that Blood test analysis and X-ray will each identify data leakages based on information which combines results from two media retriever services.



The key indicator $K_{ILL-ACC-SC}$ is lower or higher than it should be

Fig. 30. The referenced threat scenario “The key indicator $K_{ILL-ACC-SC}$ is lower or higher than is should be,” referred to in Fig. 25

TABLE XII
THE RISK EVALUATION MATRIX FROM TABLE XI WITH THE RISKS $R19 - R30$ INSERTED

Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare		$R25, R27, R29$	$R1 \& R2'$	$R3 \& R4', R8', R10', R15 \& R17$	$R12, R16 \& R18$
Unlikely		$R23$	$R1 \& R2'', R30$	$R3 \& R4'', R8'', R10'', R24$	
Possible		$R6, R7, R19, R20$	$R1 \& R2''', R9, R11, R21, R22$	$R26, R28$	
Likely	$R13$	$R5, R14$			
Certain					

X. RELATED WORK

To the best of our knowledge, there exists no other method for the design of valid key indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of key indicators. There is a tool-framework called Mozart [23] that uses a model-driven approach to create monitoring applications that employs key performance indicators. We do not focus on the implementation of key indicators, but we specify what is needed for implementing them. The work in [23] also differs from our work by not designing indicators from scratch, but by mining them from a data repository during the design cycle.

An important part of our method is the assessment of the validity of the key indicators we design. Our approach to assessing validity is inspired by research conducted within the software engineering domain. As previously explained, there is however no agreement upon what constitutes a valid software metric [8]. A number of the software metrics validation approaches advocate the use of measurement theory [24][25][26] in the validation (see e.g., [9][27][28]). Measurement theory is a branch of applied mathematics that is useful in measurement and data analysis. The fundamental idea of this theory is that there is a difference between measurements and the attribute being measured. Thus, in order to draw conclusions about the attribute, there is a need to understand the nature of the correspondence between the attribute and the measurements. In [29], an approach that relies on measurement theory for the validation of indicators is presented. This approach uses measurement theory to validate the meaningfulness of IT security risk indicators.

Measurement theory has been criticized of being too rigid and restrictive in a practical measurement setting. Briand et al. [27] advocate a pragmatic approach to measurement theory in software engineering. The authors show that even if their approach may lead to violations of the strict prescriptions and proscriptions of measurement theory, the consequences are small compared to the benefits. Another approach that takes a pragmatic approach to measurement theory is [28]. Here, the authors propose a framework for evaluating software metrics. The applicability of the framework is demonstrated by applying it on a bug count metric.

There exist also approaches that assess the validity of specific sets of key indicators. For instance, in [30] the validity of indicators of firm technological capability is assessed, while the validity of indicators of patent value is assessed in [31].

There are several approaches that focus on measuring the achievement of goals. One example is COBIT [32], which is a framework for IT management and IT governance. The framework provides an IT governance model that helps in delivering value from IT and understanding and managing the risks associated with IT. In the governance model, business goals are aligned with IT goals, while metrics, in the form of leading and lagging indicators [33], and maturity models are used to measure the achievement of the IT goals. In our approach we do not focus on the value that the use of IT has with respect to the business objectives. On the other hand, the risk that the use of IT has with respect to the business objectives is important. In our context, IT is relevant in the sense of providing the infrastructure necessary for monitoring the part of business that needs to fulfill the business objectives. In Step 6 of our method we identify risks that may result from the use of the monitoring infrastructure with respect to the business objectives.

Another way to measure the achievement of goals is by the use of the Goal-Question-Metric [34][35] (GQM) approach. Even though GQM originated as an approach for measuring achievement in software development, it can also be used in other contexts where the purpose is to measure achievement of goals. In GQM, business goals are used to drive the identification of measurement goals. These goals do not necessarily measure the fulfillment of the business goals, but they should always measure something that is of interest to the business. Each measurement goal is refined into questions, while metrics are defined for answering each question. No specific method, beyond reviews, is specified for validating whether the correct questions and metrics have been identified. The data provided by the metrics are interpreted and analyzed with respect to the measurement goal in order to conclude whether it is achieved or not. One of the main differences between our method and GQM is that we characterize precisely what it means to achieve a goal/objective. In GQM, however, this may be a question of interpretation.

In the literature, key indicators are mostly referred to in the context of measuring business performance. There exist numerous approaches to performance measurement. Some of these are presented in [36]. Regardless of the approach being used, the organization must translate their business objectives/goals into a set of key performance indicators in order to measure performance. An approach that is widely used [37] is balanced scorecard [5]. This approach translates the company's vision into four financial and non-financial perspectives. For each perspective a set

of business objectives (strategic goals) and their corresponding key performance indicators are identified. However, the implementation of a balanced scorecard is not necessarily straight forward. In [38], Neely and Bourne identify several reasons for the failure of measurement initiatives such as balanced scorecards. One problem is that the identified measures do not measure fulfillment of the business objectives, while another problem is that measures are identified without putting much thought into how the data must be extracted in order to compute the measures. The first problem can be addressed in Step 4 of our method, while the second problem can be addressed in Step 3 and Step 5 of our method. In Step 3 we identify the sensors to be deployed in the relevant part of business, while in Step 5 we present the kinds of data that needs to be extracted from these sensors in order to compute the measures.

Much research has been done in the field of data quality. The problem of data quality is also recognized within the field of key indicators [39][40]. In [41] a survey on how data quality initiatives are linked with organizational key performance indicators in Australian organizations is presented. This survey shows that a number of organizations do not have data quality initiatives linked to their key indicators. Data quality should be taken into account when designing key indicators, since the use of key indicators based on poor quality data may lead to bad business decisions, which again may greatly harm the organization.

In [42][43] the problem of key indicators computed from uncertain events is investigated. The motivation for this work is to understand the uncertainty of individual key indicators used in business intelligence. The authors use key indicators based on data from multiple domains as examples. In these papers a model for expressing uncertainty is proposed, and a tool for visualizing the uncertain key indicators is presented.

XI. CONCLUSION

In [1] we presented the method *ValidKI* (Valid Key Indicators) for designing key indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of key indicators. *ValidKI* facilitates the design of a set of key indicators that is valid with respect to a business objective. In this report we have presented the improved and consolidated version of the method.

To the best of our knowledge, there exists no other method for the design of valid key indicators to monitor the fulfillment of business objectives with particular focus on quality and ICT-supported monitoring of key indicators. The applicability of our method has been demonstrated on a large, realistic example case addressing the use of electronic patient records in a hospital environment.

Even though *ValidKI* has been demonstrated on a large, realistic example case there is still a need to apply *ValidKI* in a real-world industrial setting in order to evaluate properly to what extent it has the characteristics specified in the introduction. By applying *ValidKI* in such a setting we will for instance gain more knowledge regarding whether it is time and resource efficient.

ACKNOWLEDGMENTS

The research on which this report describes has been carried out within the DIGIT project (180052/S10), funded by the Research Council of Norway, and the MASTER and NESSoS projects, both funded from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements FP7-216917 and FP7-256980, respectively.

REFERENCES

- [1] O. S. Ligaarden, A. Refsdal, and K. Stølen, "ValidKI: A Method for Designing Key Indicators to Monitor the Fulfillment of Business Objectives," in *Proceedings of First International Conference on Business Intelligence and Technology (BUSTECH'11)*. Wilmington, DE: IARIA, 2011, pp. 57–65.
- [2] A. Hammond, A. Adriaanse, E. Rodenburg, D. Bryant, and R. Woodward, *Environmental Indicators: A Systematic Approach to Measuring and Reporting on Environmental Policy Performance in the Context of Sustainable Development*. Washington, DC: World Resources Institute, 1995.
- [3] International Organization for Standardization, International Electrotechnical Commission, and Institute of Electrical and Electronics Engineers, "ISO/IEC/IEEE 24765 Systems and Software Engineering – Vocabulary," 2010.
- [4] B. Ragland, "Measure, Metrics or Indicator: What's the Difference?" *Crosstalk: The Journal of Defense Software Engineering*, vol. 8, no. 3, 1995.
- [5] R. S. Kaplan and D. P. Norton, "The Balanced Scorecard – Measures That Drive Performance," *Harvard Business Review*, vol. 70, no. 1, pp. 71–79, 1992.

- [6] Object Management Group, "Unified Modeling Language Specification, Version 2.0," 2004.
- [7] International Organization for Standardization and International Electrotechnical Commission, "ISO/IEC 9126 Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for their Use," 1991.
- [8] A. Meneely, B. Smith, and L. Williams, "Software Metrics Validation Criteria: A Systematic Literature Review," Department of Computer Science, North Carolina State University, Raleigh, NC, Tech. Rep. TR-2010-2, 2010.
- [9] A. L. Baker, J. M. Bieman, N. E. Fenton, D. A. Gustafson, A. Melton, and R. W. Whitty, "A Philosophy for Software Measurement," *Journal of Systems and Software*, vol. 12, no. 3, pp. 277–281, 1990.
- [10] B. Kitchenham, S. L. Pfleeger, and N. Fenton, "Towards a Framework for Software Measurement Validation," *IEEE Transactions on Software Engineering*, vol. 21, no. 12, pp. 929–944, 1995.
- [11] J. M. Roche, "Software Metrics and Measurement Principles," *ACM SIGSOFT Software Engineering Notes*, vol. 19, no. 1, pp. 77–85, 1994.
- [12] B. Curtis, "Measurement and Experimentation in Software Engineering," *Proceedings of the IEEE*, vol. 68, no. 9, pp. 1144–1157, 1980.
- [13] B. Henderson-Sellers, "The Mathematical Validity of Software Metrics," *ACM SIGSOFT Software Engineering Notes*, vol. 21, no. 5, pp. 89–94, 1996.
- [14] N. E. Fenton, "Software Measurement: A Necessary Scientific Basis," *IEEE Transactions on Software Engineering*, vol. 20, no. 3, pp. 199–206, 1994.
- [15] K. El-Emam, "A Methodology for Validating Software Product Metrics," National Research Council of Canada, Ottawa, ON, Tech. Rep. NCR/ERC-1076, 2000.
- [16] J. P. Cavano and J. A. McCall, "A Framework for the Measurement of Software Quality," in *Proceedings of the Software Quality Assurance Workshop on Functional and Performance Issues*. New York, NY: ACM Press, 1978, pp. 133–139.
- [17] R. Lincke and W. Lowe, "Foundations for Defining Software Metrics," in *Proceedings of 3rd International Workshop on Metamodels, Schemas, Grammars, and Ontologies (ateM'06) for Reverse Engineering*. Mainz: Johannes Gutenberg-Universität Mainz, 2006.
- [18] M. E. Bush and N. E. Fenton, "Software Measurement: A Conceptual Framework," *Journal of Systems and Software*, vol. 12, no. 3, pp. 223–231, 1990.
- [19] Council of Europe, "Convention for the Protection of Human Rights and Fundamental Freedoms," 1954.
- [20] European Court of Human Rights, "Press Release – Chamber Judgments 17.07.08," 17. July 2008.
- [21] Helsedirektoratet, "Code of Conduct for Information Security – The Healthcare, Care, and Social Services Sector," <http://www.helsedirektoratet.no/publikasjoner/norm-for-informasjonsikkerhet/Publikasjoner/code-of-conduct-for-information-security.pdf>, Accessed: 2012-06-21, 2. June 2010.
- [22] M. S. Lund, B. Solhaug, and K. Stølen, *Model-Driven Risk Analysis: The CORAS Approach*, 1st ed. Berlin/Heidelberg: Springer-Verlag, 2010.
- [23] M. Abe, J. Jeng, and Y. Li, "A Tool Framework for KPI Application Development," in *Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE'07)*. Los Alamitos, CA: IEEE Computer Society, 2007, pp. 22–29.
- [24] D. H. Krantz, R. D. Luce, P. Suppes, and A. Tversky, *Foundations of Measurement, Vol. I: Additive and Polynomial Representations*. New York, NY: Academic Press, 1971.
- [25] P. Suppes, D. H. Krantz, R. D. Luce, and A. Tversky, *Foundations of Measurement, Vol. II: Geometrical, Threshold, and Probabilistic Representations*. New York, NY: Academic Press, 1989.
- [26] R. D. Luce, D. H. Krantz, P. Suppes, and A. Tversky, *Foundations of Measurement, Vol. III: Representation, Axiomatization, and Invariance*. New York, NY: Academic Press, 1990.
- [27] L. Briand, K. El-Emam, and S. Morasca, "On the Application of Measurement Theory in Software Engineering," *Empirical Software Engineering*, vol. 1, no. 1, pp. 61–88, 1996.
- [28] C. Kaner and W. P. Bond, "Software Engineering Metrics: What Do They Measure and How Do We Know?" in *Proceedings of 10th International Software Metrics Symposium (METRICS'04)*. Los Alamitos, CA: IEEE Computer Society, 2004.
- [29] A. Morali and R. Wieringa, "Towards Validating Risk Indicators Based on Measurement Theory," in *Proceedings of First International Workshop on Risk and Trust in Extended Enterprises*. Los Alamitos, CA: IEEE Computer Society, 2010, pp. 443–447.
- [30] T. Schoenecker and L. Swanson, "Indicators of Firm Technological Capability: Validity and Performance Implications," *IEEE Transactions on Engineering Management*, vol. 49, no. 1, pp. 36–44, 2002.
- [31] M. Reitzig, "Improving Patent Valuations for Management Purposes – Validating New Indicators by Analyzing Application Rationales," *Research Policy*, vol. 33, no. 6-7, pp. 939–957, 2004.
- [32] IT Governance Institute, "COBIT 4.1," 2007.
- [33] W. Jansen, *Directions in Security Metrics Research*. Darby, PA: DIANE Publishing, 2010.
- [34] V. R. Basili and D. M. Weiss, "A Methodology for Collecting Valid Software Engineering Data," *IEEE Transactions on Software Engineering*, vol. SE-10, no. 6, pp. 728–738, 1984.
- [35] R. V. Solingen and E. Berghout, *The Goal/Question/Metric method: A Practical Guide for Quality Improvement of Software Development*. New York, NY: McGraw-Hill International, 1999.
- [36] A. Neely, J. Mills, K. Platts, H. Richards, M. Gregory, M. Bourne, and M. Kennerley, "Performance Measurement System Design: Developing and Testing a Process-based Approach," *International Journal of Operation & Production Management*, vol. 20, no. 10, pp. 1119–1145, 2000.
- [37] T. Lester, "Measure for Measure," <http://www.ft.com/cms/s/2/31e6b750-16e9-11d9-a89a-00000e2511c8.html#axzz1ImHJOLmg>, Accessed: 2012-06-21, 5. October 2004.
- [38] A. Neely and M. Bourne, "Why Measurement Initiatives Fail," *Measuring Business Excellence*, vol. 4, no. 4, pp. 3–6, 2000.
- [39] S. M. Bird, D. Cox, V. T. Farewell, H. Goldstein, T. Holt, and P. C. Smith, "Performance Indicators: Good, Bad, and Ugly," *Journal Of The Royal Statistical Society. Series A (Statistics in Society)*, vol. 168, no. 1, pp. 1–27, 2005.
- [40] D. M. Eddy, "Performance Measurement: Problems and Solutions," *Health Affairs*, vol. 17, no. 4, pp. 7–25, 1998.

- [41] V. Masayna, A. Koronios, and J. Gao, "A Framework for the Development of the Business Case for the Introduction of Data Quality Program Linked to Corporate KPIs & Governance," in *Proceedings of the 2009 Fourth International Conference on Cooperation and Promotion of Information Resources in Science and Technology (COINFO'09)*. Los Alamitos, CA: IEEE Computer Society, 2009, pp. 230–235.
- [42] C. Rodríguez, F. Daniel, F. Casati, and C. Cappiello, "Computing Uncertain Key Indicators from Uncertain Data," in *Proceedings of 14th International Conference on Information Quality (ICIQ'09)*. Potsdam/Cambridge, MA: HPI/MIT, 2009, pp. 106–120.
- [43] C. Rodríguez, F. Daniel, F. Casati, and C. Cappiello, "Toward Uncertain Business Intelligence: The Case of Key Indicators," *Internet Computing*, vol. 14, no. 4, pp. 32–40, 2010.



Technology for a better society

www.sintef.no

Chapter 10

Paper B: Using indicators to monitor risk in interconnected systems: How to capture and measure the impact of service dependencies on the quality of provided services

Chapter 11

Paper C: An architectural pattern for enterprise level monitoring tools

Chapter 12

Paper D: Experiences from using a UML-based method for trust analysis in an industrial project on electronic procurement

Experiences from using a UML-based method for trust analysis in an industrial project on electronic procurement

Tormod V. Håvaldsrud · Olav S. Ligaarden ·
Per Myrseth · Atle Refsdal · Ketil Stølen ·
Jon Ølnes

Published online: 24 August 2010

© The Author(s) 2010. This article is published with open access at Springerlink.com

Abstract This paper reports on experiences from using a UML-based method for trust analysis in an industrial project. The overall aim of the trust analysis method is to provide a sound basis for making trust policy decisions. The method makes use of UML sequence diagrams extended with constructs for probabilistic choice and subjective belief, as well as the capture of policy rules. The trust analysis method is evaluated with respect to a set of criteria. The industrial project focused on the modeling and analysis of a public electronic procurement (eProcurement) system

The research on which this paper reports has been carried out within the DIGIT project (180052/S10), funded by the Research Council of Norway, and the MASTER project, funded from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement FP7-216917.

T.V. Håvaldsrud · O.S. Ligaarden (✉) · A. Refsdal · K. Stølen
SINTEF ICT, Oslo, Norway
e-mail: olav.ligaarden@sintef.no

T.V. Håvaldsrud
e-mail: tormod.havaldsrud@sintef.no

A. Refsdal
e-mail: atle.refsdal@sintef.no

K. Stølen
e-mail: ketil.stolen@sintef.no

T.V. Håvaldsrud · O.S. Ligaarden · K. Stølen
Department of Informatics, University of Oslo, Oslo, Norway

P. Myrseth · J. Ølnes
DNV Research and Innovation, Høvik, Norway

P. Myrseth
e-mail: per.myrseth@dnv.com

J. Ølnes
e-mail: jon.olnes@dnv.com

making use of a validation authority service for validating electronic certificates and signatures.

Keywords Trust management · Modeling · Electronic certificates · Electronic procurement

1 Introduction

Trust is often linked to the notion of subjective probability. For example, inspired by [5, 10], [11] defines trust as the subjective probability by which an actor, the trustor, expects that another entity, the trustee, performs a given transaction on which its welfare depends. Unless you are a psychologist, subjective probabilities (or beliefs) are not very interesting as long as they are studied in isolation. In computer science we are interested in trust or the more general notion of belief only as long as it has an impact on the factual (or objective) behavior of a computer system or computer-based facility. Moreover, within computer science we are often more interested in the trust of an organization than the individual trust of a human being.

In order to analyze something we need a clear understanding of this “something” (or target) to be analyzed. This target is often captured in the form of a model. Unfortunately, modeling approaches of industrial maturity targeting the computer industry (like UML [13]) do not have the expressiveness required to fully cover the aspects of relevance for a trust analysis. This motivated us to develop a method for trust analysis based on UML sequence diagrams extended with constructs for capturing

1. beliefs of agents (humans or organizations) in the form of subjective probabilities;
2. factual (or objective) probabilities of systems which may contain agents whose behavior is described in terms of subjective probabilities;
3. trust decisions in terms of policy rules with the deontic modalities obligation, prohibition and permission.

This paper reports on experiences from using this method for trust analysis (first proposed in [17]) in an industrial project focusing on the modeling and analysis of a public electronic (eProcurement) system making use of a validation authority service for validating electronic certificates and signatures. The trust analysis was conducted on behalf of Det Norske Veritas (DNV) in the autumn of 2008. DNV’s goal was to obtain a better understanding of the potential usefulness of a service they offered for supporting trust-based decisions in systems which rely on electronically signed documents. The performance of the trust analysis is evaluated with respect to a set of evaluation criteria.

The rest of the paper is organized as follows: Sect. 2 introduces our modeling approach which is UML sequence diagrams extended with constructs for probabilistic choice and belief. It also gives a brief introduction on how to specify trust policies to ensure and enforce the desirable behavior of a system. Section 3 presents the method for trust analysis, that builds on the modeling and policy specification approaches introduced in Sect. 2. Section 4 outlines the industrial case we used to test the feasibility of the trust analysis method. It also presents a set of evaluation criteria. Section 5

presents the use of the trust analysis method in the industrial case. Section 6 presents the results from the evaluation based on the criteria identified in Sect. 4. And finally, in Sect. 7 we draw the main conclusion and present related work.

2 Modeling approach

UML 2.1 sequence diagrams [13] are widely used for the modeling and specification of information systems; in particular to capture communication or interaction between system entities.

In this section we give a brief introduction to UML 2.1 sequence diagrams and the constructs for probabilistic choice and belief proposed in [18]. We also explain how sequence diagrams can be enriched to capture policy rules with deontic modalities. We use a running example: Alice purchases items on the Internet. For many of the purchases, Alice needs to send advance payment to the seller of the item. In these cases Alice runs a risk of not receiving the item after paying for it. The challenge is to model the trust considerations made by Alice, as well as their impact on the observable behavior.

2.1 Basic constructs as in the UML standard

The diagram **purchase** in Fig. 1 address the situation where Alice has found an item, with an acceptable price, that she might be interested in purchasing. The keyword `sd` (sequence diagram) in front of the diagram name marks the diagram as a sequence diagram. Each entity modeled by the diagram is represented by a dashed, vertical line called a lifeline, where the box at its top specifies which entity the lifeline represents, its name as well as its type separated by a colon. If one lifeline represents several entities with different names but of the same type, we only specify the type. Entities interact with each other through the transmission and reception of messages, which are shown as horizontal arrows from the transmitting lifeline to the receiving lifeline. For each message we distinguish between two events; a transmission event, represented by the arrow tail, and a reception event, represented by the arrow head. The transmission events occur, of course, before the corresponding receive events. The events on each single lifeline are ordered in time from top to bottom.

According to Fig. 1, Alice starts by requesting a tender from the seller. The seller sends in response a tender, signed with his electronic ID (eID) to Alice. Based on this signed tender, Alice decides whether she trusts the seller to send the item after she has sent the advance payment. Alice may behave in two alternative ways, with respect to this decision. She can either send the advance payment, or she can cancel the deal. This is represented by the outermost `alt` operator, which specifies alternative behavior. A dashed horizontal line separates the alternative behaviors. If Alice trusts the seller and sends the advance payment, the scenario continues in two alternative ways. This is represented by the innermost `alt` operator. Either the seller sends the item to Alice, or the seller does not. In the latter case Alice is forced to write off the money she paid for the item.

Fig. 1 The sequence diagram **purchase**

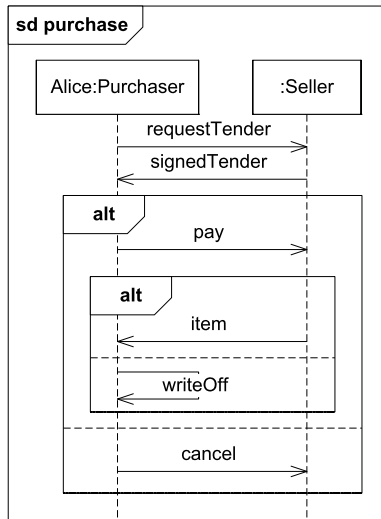
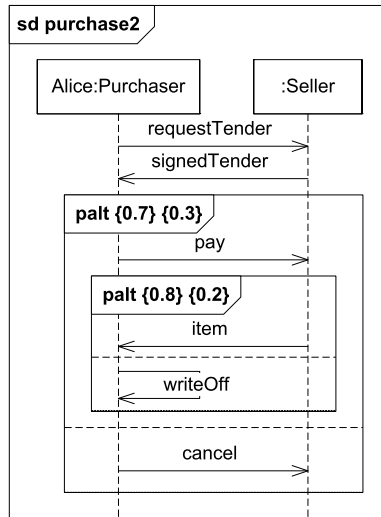


Fig. 2 The probabilistic sequence diagram **purchase2**

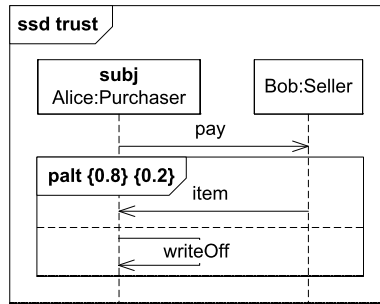


2.2 Construct for probabilistic choice

The diagram **purchase** in Fig. 1 specifies behaviors that may occur, but not how often or with what probability. Alice interacts with several sellers, and the scenario in Fig. 1 will therefore be repeated several times. We want to model how Alice behaves with respect to all these sellers. For this purpose we introduce the `palt` construct for probabilistic choice. By using the `palt` construct we can say something about the probability of a specific behavior.

The diagram **purchase2** in Fig. 2 is a probabilistic version of **purchase** in Fig. 1. The numbers occurring after `palt` in the upper corner of the operator frame specify the probabilities of the various alternatives. In the outermost `palt` the first number

Fig. 3 The subjective sequence diagram **trust**



states that the scenario of the first operand occurs with a probability of 0.7, which means that this behavior occurs in 70% of the cases, while the second number states that the scenario of the second operand occurs with a probability of 0.3, which means that this behavior occurs in 30% of the cases. Furthermore, for the innermost `palt` operator the first operand has a probability of 0.8 of occurring, while the second operand has a probability of 0.2 of occurring. The probabilities in this diagram are factual in the sense that they are meant to reflect observable probabilistic behavior of the system.

2.3 Belief construct

It is clear that Alice behaves based on how much she trusts the seller of the item, but this notion of trust is not really reflected in the diagrams we have seen so far. They capture that she makes a choice, but not whether this choice is based on trust or something else. We want to model explicitly to what degree she needs to trust a seller before she sends advance payment.

Trust is the belief of a trustor that a trustee will perform a specific transaction on which the welfare of the trustor depends. Often the trustor will only expect the trustee to perform the transaction if another event has already occurred. In our example this event would be the sending of advance payment from Alice to the seller. Here, Alice is the trustor, while the seller is the trustee. Alice believes that there is a certain probability that the seller will send the item.

To model trust considerations we use so-called subjective sequence diagrams. Subjective sequence diagrams captures the subjective belief of an actor. Syntactically, subjective sequence diagrams differ from the ordinary sequence diagrams in two respects. Firstly, `ssd` (subjective sequence diagram) is used instead of `sad` to mark the diagram. Secondly, we annotate exactly one lifeline head with the keyword `subj`. This identifies the annotated entity as the subject, meaning that the diagram is used to capture this entity's subjective belief. According to the subjective sequence diagram **trust** in Fig. 3 Alice believes that the probability of receiving the item after sending the payment to the seller Bob is 0.8, and that the probability of not receiving the item is 0.2.

Semantically, a subjective sequence diagram aims to capture the belief of some entity like a person, an organization, or even a computer to the extent a computer may be said to believe. An ordinary sequence diagram, on the other hand, aims to capture

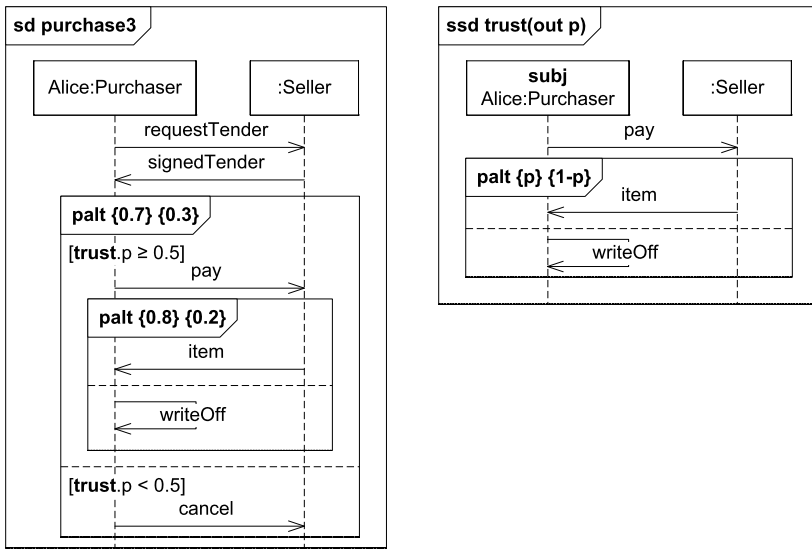


Fig. 4 The objective sequence diagram **purchase3** and the subjective sequence diagram **trust**

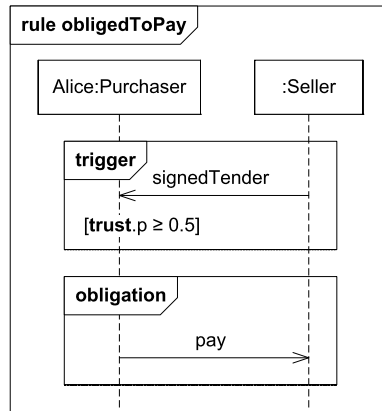
the factual reality, i.e. how things really are, independent of subjective beliefs, and such diagrams are in the following often referred to as objective sequence diagrams.

2.4 Combining objective and subjective diagrams

In the previous section we showed how we can model Alice’s trust in a seller using subjective sequence diagrams. In this section we explain how subjective diagrams relate to the objective ones. Alice will only send payment if her trust in the seller is sufficiently high, i.e. if it reaches a certain threshold. The threshold says how much trust Alice needs to have in the seller in order to send him advance payment. In the diagram **purchase3** in Fig. 4, the threshold is represented as guards. A guard is a Boolean expression within square brackets. It constrains the choice of operand. An operand can only be chosen if its guard evaluates to true. We can see that the two guards refer to the variable *trust.p*. Here, *trust* refers to the subjective diagram **trust** in Fig. 4. Unlike the diagram in Fig. 3, which captures the trust with respect to one specific seller, this diagram uses the variable *p* for the probability of the *palt* operands. This variable can be referred to in the objective diagram, since it is an out parameter of the subjective diagram. The *trust.p* expression in the objective diagram refers to this output value. The guards in the objective diagram specify that Alice sends advance payment if she believes that the probability of receiving the item is greater than or equal to 0.5. This will happen in 70% of the cases, since the operand where this guard holds has the probability of 0.7.

2.5 Policy specification

A policy is a set of rules that determines choices in the behavior of a system [19], and is used in policy based management. Each rule determines a system choice of

Fig. 5 Example of a policy rule

behavior, where a given trust level is a decisive factor for each choice. Enforcement of the given rules aims to ensure the optimal balance of the risks and opportunities that are imposed by trust based decisions within the system.

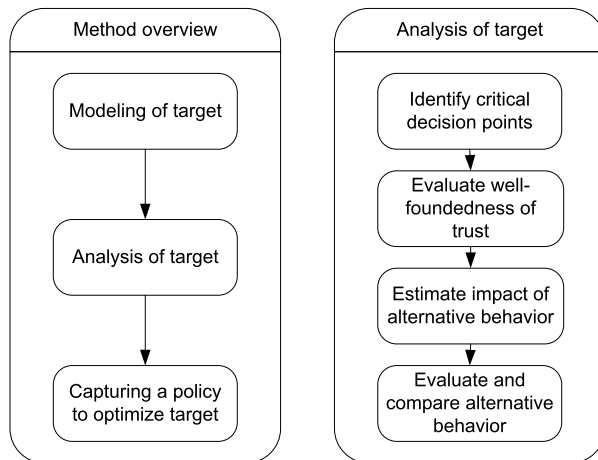
To formalize the policy the trust analysis method, proposed in [17], uses Deontic STAIRS [21], which is a language for expressing policies, and based on UML sequence diagrams. Deontic STAIRS has the expressiveness to specify constraints in the form of obligations, prohibitions, and permissions, corresponding to the expressiveness of standard deontic logic [12]. Such constraints are normative rules that describe the desired system behavior. This reflects a key feature of policies, namely that they “define choices in behavior in terms of the conditions under which predefined operations or actions can be invoked rather than changing the functionality of the actual operations themselves” [20]. Furthermore, Deontic STAIRS supports the specification of triggers that define the circumstances under which the various rules apply. In particular, the policy triggers can specify the required trust levels for a particular choice of behavior to be constrained.

Figure 5 shows an example of a policy rule in Deontic STAIRS for the scenario described in this section. The keyword `rule` in the upper left corner indicates that the diagram specifies a policy rule, while **obligedToPay** is the name of the rule. The diagram consists of two parts, a trigger and an interaction that is the operand of a deontic modality.

The first operator with keyword `trigger` specifies the circumstances under which the rule applies and consists of an interaction and a condition. The former refers to a scenario such that when it occurs, the rule applies. In this case the scenario is the reception by Alice of a signed tender. The condition of the trigger limits the applicability of the rule to a set of system states. In this case it refers to the states in which the relevant trust level is 0.5 or higher.

The second operator with keyword `obligation` shows the modality of the rule, while its operand specifies the behavior that is constrained by the rule. In this case, the relevant behavior is that Alice sends payment to the seller. According to **obligedToPay**, she is obliged to do so, given that the trigger is fulfilled. On the other hand, if the keyword had been `prohibition` then Alice would have been prohibited from sending payment, while if the keyword had been `permission` then Alice could choose whether or not to send payment.

Fig. 6 Overview of the method proposed in [17]. The *right-hand side of the figure* shows the sub-steps of the second step



3 The trust analysis method

In this section we give a brief overview of the trust analysis method that was introduced in [17]. For further details we refer to [17] and of course Sect. 5 of this paper which describes how the method was used in the industrial project on which this paper reports.

Figure 6 shows an overview of the method. There are three major steps. The first step is to model the target, the second step is to analyze the target and the third step is to capture policies to optimize the behavior of the target based on the knowledge acquired in the first two steps.

Step 1. Modeling of target. In order to analyze a system, we first need to understand the system under analysis, including the behavior of its users. A major goal of the first step and the resulting models is to provide such an understanding. However, as most systems are highly complex, it is neither feasible nor desirable to take every detail into account. Therefore the target should be modeled at a level of abstraction suitable for the analysis to come. Thus, the models should only capture the aspects of the system that enhances our understanding of the decisions that are taken on the basis of trust and the considerations that lie behind these decisions, as well as the resulting system behavior and outcomes that are relevant.

As explained in Sect. 2, the modeling approach is based on UML sequence diagrams. The reason is that trust is mainly of relevance in the context of interactions between different entities, and sequence diagrams are well suited for modeling interactions. Moreover, UML sequence diagrams are fairly easy to understand at an intuitive level. This is important, as the models developed in the first step should serve as a point of focus for discussions and as an aid in communication between the analysts and participants throughout the analysis. The extensions of UML sequence diagrams provided by subjective STAIRS [18] ensures that the trust considerations behind decisions can be captured in the models, as well as the resulting system behavior.

Step 2. Analysis of target. After a suitable model of the target has been established, the next step is to conduct the actual analysis. This involves investigating the current system behavior and the way in which trust-based decisions are being made, as well as potential alternative behaviors. The aim is to obtain a good understanding of the risks and opportunities involved. The analysis is divided into four sub-steps.

Step 2.1. Identify critical decision points. In this sub-step critical decision points that will be further investigated are identified. This will typically be points where actors in the system make trust-based decisions. But it may also be points where one could benefit from introducing new trust-based decisions. For example, if time is a critical factor, it may be more important to make a quick decision than to make the optimal decision. In such cases, it may be better to allow actors to make decisions based on trust than to insist on more time-consuming decision procedures.

Step 2.2. Evaluate well-foundedness of trust. Trust involves a subjective estimate of the potential behavior of another entity. The second sub-step of Step 2 consists of evaluating to what degree the subjective estimates reflect reality. In the industrial project on which this paper reports this step was not relevant since our task was not to evaluate an existing trust solution, but rather to develop a policy from scratch.

Step 2.3. Estimate impact of alternative behavior. In this sub-step the impact of various alternative behaviors that the system may potentially perform is investigated with respect to risks and opportunities. The goal is to get an understanding not only of the current “as-is” system behavior, which may not be optimal, but also of potential alternatives. Typically, this involves asking “what if” questions about the system, and capturing the answers in models. For example: what would be the overall effect on the system behavior if a certain actor was more (or less) willing to engage in interactions with other entities? What happens if a different policy is applied when making a certain decision?

Step 2.4. Evaluate and compare alternative behavior. In the final sub-step, the different alternative behaviors that were identified and investigated in the previous step are evaluated and compared. The purpose is to identify behaviors that should be sought or avoided.

Step 3. Capturing a policy to optimize target. The final step of the method proposed in [17] consists of using the obtained knowledge about preferred behavior to form policies to ensure and enforce the desirable behavior.

4 The industrial project on electronic procurement

We now present the industrial project in which the trust analysis method outlined in Sect. 3 was applied. The trust analysis method was used to model and analyze a public eProcurement system, which makes use of a Validation Authority (VA) service for validating electronic certificates and signatures. We first present the public eProcurement system, before describing how this system can make use of the VA service. Then we present criteria for evaluating the trust analysis method.

4.1 Public electronic procurement

Public eProcurement is used by public authorities within the EU to award public work contracts, public supply contracts, and public service contracts to economic operators [16]. We consider only the open procedure for individual contracts as specified in [3]. In the open procedure, any interested economic operator may submit a tender. The procedure consists of three phases: eNotification, eTendering, and eAwarding. In the eNotification phase a procurement officer¹ creates a call for tenders. This call specifies the requirements of the contracting authority for the goods/services/works to be procured. In the eTendering phase, interested economic operators will create tenders containing legal, financial, and technical information. Before submitting the tender electronically, one or more persons representing the economic operator need to sign the tender with their electronic IDs (eIDs), issued by Certificate Authorities (CAs). When received by the system, the system will examine whether the tender is compliant with the requirements defined in the call, including examining whether the digital signatures in the tender are valid. The eAwarding phase begins after the deadline for submission has expired. In this phase the contract is awarded based on an evaluation of the received tenders.

4.2 The validation authority service

For the eProcurement system it is important to be able to accept electronically signed tenders from electronic operators from all over Europe, regardless of the eID used by the operator. Due to the potential large number of CAs, the technical validation of eIDs and digital signatures has some challenges with respect to scaling [14], but the real problem is the assessment of the risk implied by accepting a digital signature. Here, one particular concern is that an economic operator can refute the validity of the offer stated in the submitted tender, if awarded the contract. The eProcurement system can ensure that this risk is acceptable by making an assessment of the signature quality and accepting only those of a certain quality. The higher the quality is, the harder it would be for an economic operator to refute the validity of a submitted tender. The quality of a signature [15] can be decided from the quality of the eID, which is derived from the certificate policy of the CA, and the cryptography used. A certificate policy may be written in a foreign language and may refer to a foreign legislation, so with a large number of CAs, the contracting authorities will have a hard time determining the quality of digital signatures. Thus, it will be hard if not impossible for the contracting authorities to have agreements with all the CAs on which it may want to rely, which again limits the number of economic operators that can submit tenders. A solution to this, as proposed in [15], is to use a VA as the single trust anchor, as shown in Fig. 7. In the figure we can see that the VA supports a number of CAs. For each CA that it supports, the VA is able to assess the quality of the eIDs issued by this CA and the signatures produced with those eIDs. A relying party, in this case the eProcurement system, can then validate and assess the quality of the

¹Representative for the contracting authorities.

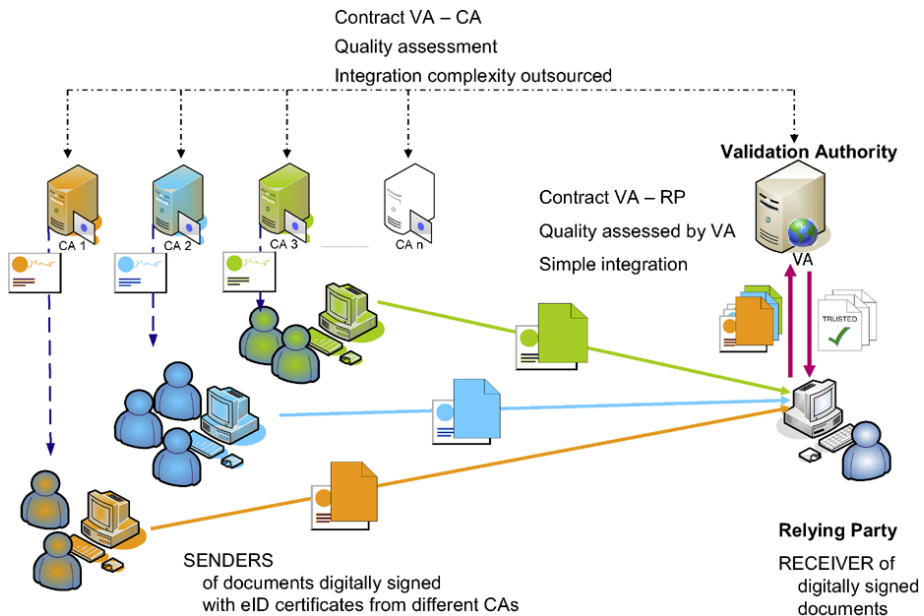


Fig. 7 Figure from [15]. The figure shows how a relying party (in this case a public eProcurement system) may use a VA service to validate and assess the quality of signatures in signed documents (in this case signed tenders)

signature² in a signed tender by issuing a request to the VA. If the eID used to create the signature has been issued by a supported CA, the VA will use the CA to validate the eID, while the quality of the signature is computed by the VA by applying the formula

$$\text{Signature Quality} = \text{eID Quality} + \text{Hash Quality} \\ + \text{Public Key Crypto Key Length Quality},$$

which will assign a Signature Quality value from 0 to 20 according to criteria further specified in [15]. This value is then compared to the minimum required quality level requested by the eProcurement system. If the signature is valid, meaning that the technical validation of the eID and the signature was successful, and the signature has sufficient quality, the VA will give the tender a trusted verdict. Otherwise, the VA will give the tender a not trusted verdict.³ By trusting the VA and its assessments, the eProcurement system is able to trust any CA that the VA handles. Hence, the eProcurement system can support a large number of CAs and it gets a one-stop shopping service for verification of digital signatures and eIDs and quality assessment of digital signatures.

²It is possible to sign a tender with more than one signature. The VA is able to make an overall quality assessment of all these signatures.

³The VA can also give an inconclusive verdict. This will happen in the cases when the VA cannot validate the eID and/or signature and/or cannot assess the quality of the signature.

4.3 Evaluation criteria

A major objective with applying the trust analysis method in the industrial project was to get an idea of how well the method performs in a practical setting. To do so we need a set of evaluation criteria and they are characterized and motivated in the following. The criteria are general in the sense that they do not target the eProcurement system or VA service specifically; they are equally valid for other kinds of trust-related infrastructures on which the trust analysis method may be applied.

When evaluating a method for trust analysis there are of course many concerns. To make sure that we covered the most important concerns we started by identifying groups of stakeholders of relevance for the method. What is important for one group may of course be less so for another group. We identified three main groups of stakeholders, and the evaluation criteria are based on the point of view for each of these groups. First, the *customers* are those who pay for the analysis. Typically, this will be managers and decision makers. They do not necessarily take part in the analysis process themselves, but will use the results of the analysis as a basis for making policy decisions. Second, the *analysts* are those who will conduct the analysis (process) and document results. They know the analysis method, but cannot be assumed to know the particular target system at the start of the analysis. Third, the *participants* are people such as decision makers, system users, developers, or engineers with whom the analysts interact during the analysis process. We now present evaluation criteria classified according to the stakeholder group for which they are most relevant.

For the *customer* of a trust analysis, the overall goal is to make the right trust policy decisions. This requires a good understanding of the outcome of potential alternative trust policies. Hence:

EC1: The trust analysis should provide the customers with a good basis for making trust policy decisions. This means that sufficient information about the impact of the potential alternatives must be provided.

Clearly, the cost of the analysis needs to be justified with respect to the benefit for the customer. Hence:

EC2: The trust analysis should be cost effective.

The task of the *analyst* is to conduct the trust analysis and to document the findings within the allotted time and cost frame. This means that the trust analysis method should be sufficiently simple to be carried out within a reasonable time frame. However, as this is implied by the requirement expressed in **EC2**, we do not include this as a separate criterion. On the other hand, the analyst would like to document the findings, and in particular all assumptions and constraints on which their validity depends, to cover him/herself as much as possible. Hence:

EC3: The modeling approach should be sufficiently expressive to capture the information, assumptions, and constraints of relevance.

The *participant* is supposed to communicate her or his knowledge in such a way that the analysis will result in correct models of the target, and the models should serve as a means of communication between the analysts and participants. It is therefore important that the models are comprehensible for the participants when properly

assisted by an analyst. Otherwise, it will be hard for them to identify shortcomings and errors. Hence:

EC4: The models should be comprehensible for the participants of the analysis.

5 Trust analysis in the industrial project

In this section we present how the trust analysis method as described in Sect. 3 was applied in the industrial case outlined in Sect. 4.

5.1 Step 1. Modeling the target

The trust analysis focused on the scenarios where the eProcurement system makes decisions based on trust, i.e. where it is decided whether a received tender should be trusted to be authentic or not. On the one hand, it was an objective to minimize the risk that non-authentic tenders were accepted for further evaluation in the eAwarding phase, as contracts should not be awarded based on non-authentic tenders. On the other hand, it was also an objective to avoid authentic tenders being rejected without further evaluation.

There was some discussion on whether non-authentic tenders actually represent a real problem. Although this may not be the case today, it was agreed that this may easily become a problem in the future, as the use of eProcurement increases. It is easy to imagine cases where economic operators submit false tenders in a competitor's name. The motivation for this could be, for example, to ensure that minimum requirements on the number of received tenders to enable eAwarding are fulfilled, or to bind the competitor to unfavorable obligations, or to make the operator's own tender appear more attractive compared to a false costly tender.

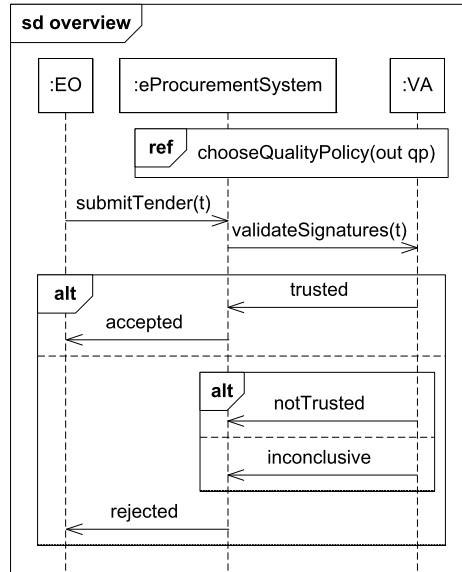
The decision on whether to trust the authenticity of a tender is made in the eTendering phase, while the selection of the best tender based on price, quality, and so on from the tenders judged to be authentic is made in the eAwarding phase. Thus, for the purpose of the trust analysis we are only interested in the behavior related to submission of tenders in the eTendering phase. The task in Step 1 is therefore to model this behavior. Figure 8 shows the resulting overview diagram.

First the eProcurement system needs to find the minimum quality level the signatures have to comply with to be accepted to the eAwarding phase. This particular process is described in more detail in the diagram **chooseQualityPolicy** in Fig. 9.⁴ Choosing the quality policy and communicating the choice to the VA is typically done even before the eNotification phase, since the economic operators must be informed about the requirements for the submission. Note that the eNotification phase is not captured in the models, as it is of little relevance for the trust analysis.

After the quality policy has been set an economic operator may submit a tender t to the eProcurement system. This is represented by the message `submitTender(t)` in the diagram. The eProcurement system will validate the signatures of this tender by

⁴The `ref` construct is a reference to another diagram. Its meaning is the same as we would get by inserting the contents of the referred diagram at the place of the reference.

Fig. 8 A simplified description of how submitted tenders are handled by the eProcurement system



using the VA service `validateSignatures(t)`. The VA will then use the required minimum quality level to decide whether the signature should be trusted or not.

The first operand of the outermost `alt` operator describes the case where the tender is reported `trusted` by the VA, and therefore is accepted for further evaluation by the eProcurement system. The second operand describes the case where the tender is reported as `notTrusted` or as `inconclusive`, and therefore rejected by the eProcurement system.

We now explain how the choice of quality policy level performed by the eProcurement system is captured in the models. Intuitively, the process of choosing one of the 21⁵ quality policy levels can be described as follows: the eProcurement system uses a threshold value that specifies the least amount of trust that is needed to accept the risk of accepting a non-authentic tender. In order to balance the risk of accepting a non-authentic tender against the desire not to reject authentic tenders, the eProcurement system chooses the lowest quality policy level needed to ensure that its trust exceeds the threshold.

Figure 9 describes the process of choosing a suitable quality policy based on trust. The diagram `chooseQualityPolicy` shows that there are 21 alternative quality policies from which the eProcurement system may choose. After choosing the quality policy in terms of an assignment, the eProcurement system communicates to the VA service the chosen policy, as shown by the `setQualityPolicy(qp)` message at the bottom of the diagram.

The trust level which the `threshold` is compared to, is captured by expressions of the form `trust(j).p`, where `j` is one of the 21 quality policy levels; this value is bound to the input parameter `qp` of the subjective diagram `trust` in Fig. 9. So,

⁵Remember that the quality policy scale is from 0 to 20.

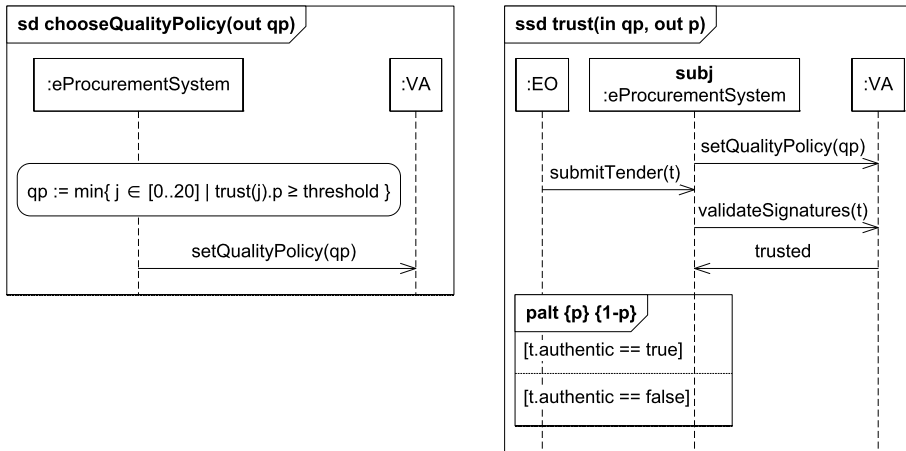


Fig. 9 `chooseQualityPolicy` and `trust` describe how the quality policy is found and set

for example, `trust(7).p` yields the return value of the subjective sequence diagram **trust**, assuming quality policy level 7 is used. As shown by the lifeline head, the eProcurement system is the trustor. Therefore, the expression `trust(j).p` represents the trust of the eProcurement system that a tender that receives a trusted verdict from the VA service is indeed authentic, given that quality policy level j is used. Note that the diagrams in Fig. 9 do not refer to one specific VA. Hence, `trust(j).p` may yield different values for different VAs for the same tender.

5.2 Step 2. Analyzing the target

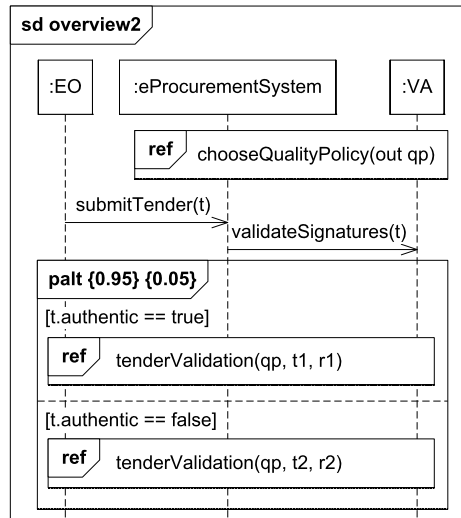
We now explain how the sub-steps of Step 2 were performed.

Step 2.1. Identify critical decision points. The only critical decision point that was identified was the point where the signature quality policy is chosen by the eProcurement system. The reason for this was that the analysis was performed from the point of view of the eProcurement system, with the purpose of setting the right trust threshold. This decision point is represented by the assignment in the diagram **chooseQualityPolicy** in Fig. 9.

Step 2.2. Evaluate well-foundedness of trust. As explained in Sect. 3, this step was not relevant in the project on which this paper reports. Our task was not to evaluate one particular trust solution, but rather come up with a policy for how to choose quality policy level.

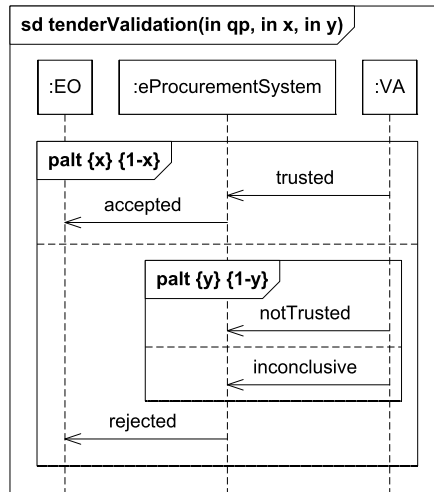
Step 2.3. Estimate impact of alternative behavior. As the decision point under analysis was the choice of quality policy level, the task of this sub-step was to estimate the impact of choosing different levels with respect to how many of the authentic or non-authentic tenders would receive the different verdicts from the VA service. This would serve as a basis for the evaluation and comparison in the next sub-step. To do so, we obviously needed to collect data. No historical data were available, and we

Fig. 10 A simplified description of how submitted tenders are handled by the eProcurement system, where probabilities have been assigned to alternatives



had to base ourselves on expert judgments. The experts were all employees of DNV including the two co-authors from DNV. The data they provided is documented in Figs. 10, 11, and Table 1. Figure 10 is a refinement of Fig. 8. It is unchanged above the `palt` operator. The first operand of the `palt` operator represents the cases where tenders are authentic, while the second alternative represents the cases where they are non-authentic. We consider a tender to be authentic if it actually comes from the company (EO) in whose name it has been submitted. This applies even if the employees who have signed the tender electronically is not strictly speaking authorized by the company to do so, as long as the company acknowledges the tender and intends to honor its commitment. To emphasize the fact that the EO is the only entity who actually knows whether the tender is authentic, we have used guards on the EO lifeline in the model to represent whether a tender is authentic or not; the two cases are represented by the guards `t.authentic==true` and `t.authentic==false`, respectively. The probability 0.95 assigned to the first `palt` operand in Fig. 10 captures that 95% of received tenders are authentic. The remaining 5% are non-authentic and the second `palt` operand is therefore assigned the probability 0.05. The two operands of the `palt` operator in Fig. 10 refer to the same parameterized diagram (i.e. **tenderValidation** in Fig. 11), as the behavior of the system for the two cases only differ with respect to the probabilities for the alternatives. The **tenderValidation** diagram has three input parameters; namely the quality policy (`qp`), the probability (x) of being judged as trusted by the VA with respect to the selected quality policy, and similarly the probability (y) of being judged as not trusted as opposed to inconclusive in the other case. The first operand of the outermost `palt` operator in **tenderValidation** gives the probability for the tender being reported `trusted` by the VA, and therefore is accepted for further evaluation by the eProcurement system. The second operand shows the case where the tender is reported as `notTrusted` or as `inconclusive`, and therefore rejected by the system; this will occur with probability $1 - x$. Within this alternative, the `notTrusted` verdict from the VA will occur with probability y , while the `inconclusive` verdict will occur with probability $1 - y$.

Fig. 11 tenderValidation
shows how tenders are handled



The actual values of x and y depend on whether the tender is authentic or not. Therefore the references to **tenderValidation** in the operands of the `palt` in Fig. 10 bind x and y to different entities. In the first operand representing the cases where tenders are authentic, x is bound to t_1 and y is bound to r_1 . In the second operand representing the cases where tenders are non-authentic, x is bound to t_2 and y is bound to r_2 . The intuitive meaning of t_1 , t_2 , r_1 , and r_2 for a given quality policy qp can be summarized as follows: t_1 denotes the probability of assigning a trusted verdict to an authentic tender; r_1 denotes the conditional probability of assigning a not trusted verdict (as opposed to inconclusive) for an authentic tender given that a trusted verdict is not assigned; t_2 denotes the probability of assigning a trusted verdict to a non-authentic tender; r_2 denotes the conditional probability of assigning a not trusted verdict (as opposed to inconclusive) for a non-authentic tender given that a trusted verdict is not assigned.

Table 1 shows how the representatives from DNV estimate that the probabilities will vary according to the quality policy. Note that even though the VA service offers a quality scale from 0 to 20, it was deemed sufficient to analyze only five different quality policy levels for the purpose of this analysis. Based on a consideration of criteria for assigning quality levels, the following steps on the scale were selected for analysis: 0, 5, 7, 10, and 20. The first line in the table provides the values of the parameters t_1 , r_1 , t_2 , and r_2 , in the diagram in Fig. 10, if the quality policy 0 ($qp = 0$) is chosen. The second line provides the values in the case where the quality policy 5 is chosen and so on.

Given the data captured by Table 1, we calculated the probabilities for the possible combinations of authenticity and verdicts, which amounted to a simple multiplication of the probabilities assigned in the objective diagrams in Figs. 10 and 11. For example, the probability that a given tender is authentic and receives a trusted verdict is obtained by $0.95 \times t_1$, while the probability that it is non-authentic and receives an inconclusive verdict is obtained by $0.05 \times (1 - t_2) \times (1 - r_2)$. Table 2 shows the result from these calculations (when inserting the values from Table 1 for the variables t_1 , t_2 , r_1 , and r_2).

Table 1 The probabilities corresponding to the quality policy

Quality policy	t1	r1	t2	r2
0	0.65	0.05	0.05	0.10
5	0.80	0.15	0.01	0.20
7	0.95	0.40	0.005	0.50
10	0.75	0.70	0.002	0.80
20	0.80	0.80	0.001	0.90

Table 2 Probabilities for authentic and non-authentic tenders

Quality policy	Authentic			Non-authentic		
	Trusted	Not trusted	Inconclusive	Trusted	Not trusted	Inconclusive
0	0.61750	0.01663	0.31589	0.00250	0.00475	0.04275
5	0.76000	0.02850	0.16150	0.00050	0.00990	0.03960
7	0.90250	0.01900	0.02850	0.00025	0.02488	0.02488
10	0.71250	0.16625	0.07125	0.00010	0.03992	0.00998
20	0.76000	0.15200	0.03800	0.00005	0.04955	0.00500

Figure 12 shows the left-hand part of Table 2 as a trend-graph, i.e. it shows the probability that a tender is authentic and receives each of the three possible verdicts, depending on the chosen quality level.⁶ On the left-hand side of the graph, we see that the probability of getting a trusted verdict is relatively low (but increasing), while the probability of getting an inconclusive verdict is correspondingly high (but decreasing). According to the domain experts providing the data, the reason for this is that when a request for tenders is announced with low certificate and signature requirements, relatively many of the received tenders will use certificates from CAs that are not supported by a VA; there are many CAs offering low quality eIDs, and a VA service is primarily aimed at supporting CAs with higher quality eIDs. After quality policy level 7, we see a decrease in the probability of receiving a trusted verdict. According to the same experts, this is due to the fact that when higher quality policy levels are used, more of the received tenders will use certificates from CAs that are supported by the VA, but of insufficient quality.

Figure 13 shows the right-hand part of Table 2 as a trend-graph, i.e. it shows the probability that a tender is non-authentic and receives each of the three possible verdicts. Here we are operating with very small scales, due to the small amount (5%) of non-authentic tenders, and the probability for getting the trusted verdict is almost non-existing for all quality policy levels. Not surprisingly, the probability for a non-authentic tender getting the not trusted verdict increases with increasing quality policy level. Furthermore, the probability of an inconclusive verdict decreases with increasing quality level, as more of the received tenders will use certificates from CAs that are supported by VA when high quality policies are used.

⁶Recall that 95% of tenders are assumed to be authentic in all cases.

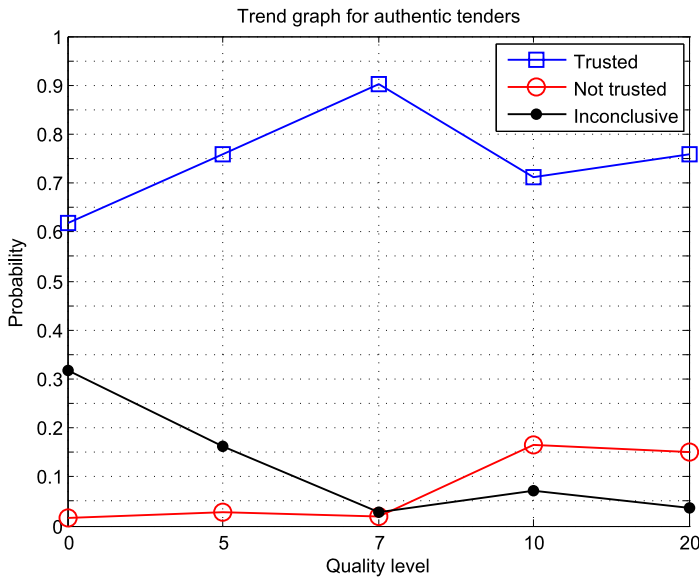


Fig. 12 Trend graph for authentic tenders

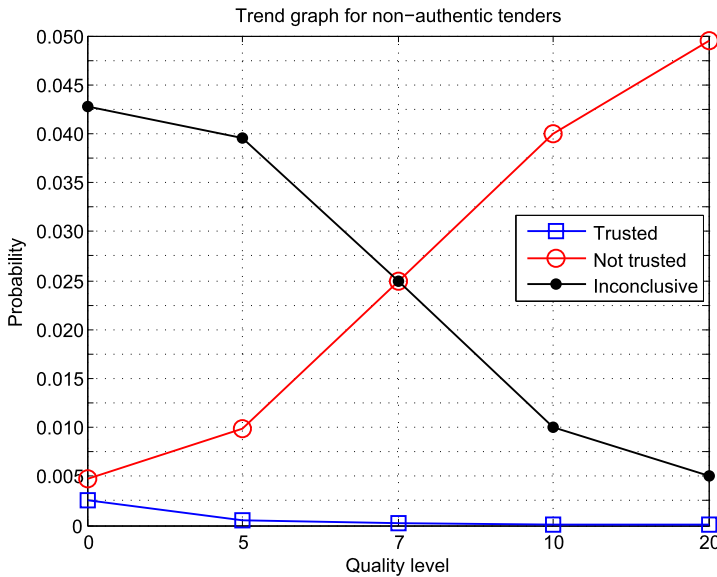


Fig. 13 Trend graph for non-authentic tenders

Step 2.4. Evaluate and compare alternative behavior. In order to decide which of the quality policies that will give the optimal behavior of the system, we looked at the probabilities for desirable and undesirable outcomes (opportunities and risks) for the

different quality levels, with the goal of finding the optimal balance. For each quality level, the desirable outcomes are as follows:

- A tender that has been accepted by the system, due to a trusted verdict from the VA, is authentic.
- A tender that has been rejected by the system, due to a not trusted or an inconclusive verdict from the VA, is non-authentic.

We seek to maximize the probabilities of these outcomes, since these outcomes represent opportunities. The probabilities are given as follows:

- $P(\text{aut}|\text{acc})$ —The conditional probability of a tender being authentic, given that it is accepted by the system.
- $P(\text{not aut}|\text{not acc})$ —The conditional probability of a tender being non-authentic, given that it is rejected by the system.

On the other hand, for each quality level, the undesirable outcomes are as follows:

- A tender that has been accepted by the system, due to a trusted verdict from the VA, is non-authentic.
- A tender that has been rejected by the system, due to a not trusted or an inconclusive verdict from the VA, is authentic.

We seek to minimize the probabilities of these outcomes, since these outcomes represent risks. The probabilities are given as follows:

- $P(\text{not aut}|\text{acc})$ —The conditional probability of tender being non-authentic, given that it is accepted by the system.
- $P(\text{aut}|\text{not acc})$ —The conditional probability of a tender being authentic, given that it is rejected by the system.

From the diagrams in Figs. 10 and 11 we get the following values directly: the probability $P(\text{aut}) = a$ for a tender being authentic is 0.95, irrespective of the quality policy, while the probability for a tender being non-authentic is $P(\text{not aut}) = 1 - a$. We also have the probabilities $P(\text{acc}|\text{aut}) = \tau_1$ and $P(\text{not acc}|\text{aut}) = 1 - \tau_1$ for a tender being accepted and not accepted by the system, given that it is authentic, while $P(\text{acc}|\text{not aut}) = \tau_2$ and $P(\text{not acc}|\text{not aut}) = 1 - \tau_2$ give the probabilities for a tender being accepted and not accepted, given that it is non-authentic. Values for τ_1 and τ_2 , depending on the chosen quality level, are taken from Table 1. The probabilities for a tender being accepted and not accepted are obtained as follows:

$$\begin{aligned} P(\text{acc}) &= P(\text{aut}) \times P(\text{acc}|\text{aut}) + P(\text{not aut}) \times P(\text{acc}|\text{not aut}) \\ &= a \times \tau_1 + (1 - a) \times \tau_2 \end{aligned} \quad (1)$$

$$P(\text{not acc}) = 1 - P(\text{acc}) \quad (2)$$

The conditional probabilities, mentioned above, were calculated for the different quality levels by applying Bayes' theorem as follows:

$$P(\text{aut}|\text{acc}) = \frac{P(\text{acc}|\text{aut}) \times P(\text{aut})}{P(\text{acc})} = \frac{\tau_1 \times a}{a \times \tau_1 + (1 - a) \times \tau_2} \quad (3)$$

Table 3 Table showing the probabilities related to opportunity (column 2 and 5) and risk (column 3 and 4) for the different quality policies

Quality policy	$P(\text{aut} \text{acc})$	$P(\text{aut} \text{not acc})$	$P(\text{not aut} \text{acc})$	$P(\text{not aut} \text{not acc})$
0	0.995968	0.875000	0.004032	0.125000
5	0.999343	0.793319	0.000657	0.206681
7	0.999723	0.488432	0.000277	0.511568
10	0.999860	0.826374	0.000140	0.173626
20	0.999934	0.791832	0.000066	0.208168

$$P(\text{aut}|\text{not acc}) = \frac{P(\text{not acc}|\text{aut}) \times P(\text{aut})}{P(\text{not acc})} = \frac{(1 - t_1) \times a}{1 - (a \times t_1 + (1 - a) \times t_2)} \quad (4)$$

$$P(\text{not aut}|\text{acc}) = 1 - P(\text{aut}|\text{acc}) \quad (5)$$

$$P(\text{not aut}|\text{not acc}) = 1 - P(\text{aut}|\text{not acc}) \quad (6)$$

The results of the calculations are shown in Table 3. For $P(\text{aut}|\text{acc})$, which we want to maximize, there is little difference between the values of $P(\text{aut}|\text{acc})$ for the different quality policies. On the other hand, for $P(\text{not aut}|\text{not acc})$, which we also want to maximize, we see that for level 7 we have a much higher value than for the others.

5.3 Step 3. Capturing a policy to optimize target

The numbers in Table 3 provide useful input, assuming of course that the expert judgments are sound. However, they do not take the nature of the call for tender into consideration, which of course is an essential factor when formulating a policy. After all, the significance of the numbers in Table 3 depends heavily on what is to be procured. If the cost of goods to be procured is low (e.g. pencils for the administration), we would probably worry only about $P(\text{aut}|\text{acc})$ and based on that choose quality policy 0. This is partly because the difference in $P(\text{aut}|\text{acc})$ for the quality policy levels does not matter much when the cost of goods to be procured is low, and partly because a higher quality level might frighten off potential submitters of tenders.

On the other hand, if the cost of the goods to be procured is very high (e.g. new fighter planes in the extreme case) the procurer would probably want as much legal coverage as possible and use quality policy level 20, since this gives the best value for $P(\text{aut}|\text{acc})$. Moreover, if the goods to be procured are so costly that it is important to avoid disqualifying authentic tenders as well as obtaining a high level of trust in certificates, quality policy level 7 seems to be the best option.

Based on these considerations we ended up with the policy rules specified in Figs. 14–16. We make the assumption that the eProcurement system trusts the VA and its assessments. This is important since an eProcurement system cannot make use of the VA service if it is not trustable. The trigger of each rule contains a condition which limits the applicability of the rule to a set of system states. For rule **qp0** in Fig. 14 the condition is that the cost of the goods to be procured is low, while for rule **qp20** in Fig. 15 it is that the cost is very high. For rule **qp7** in Fig. 16 the condition

Fig. 14 Policy rule for the selection of quality policy level 0

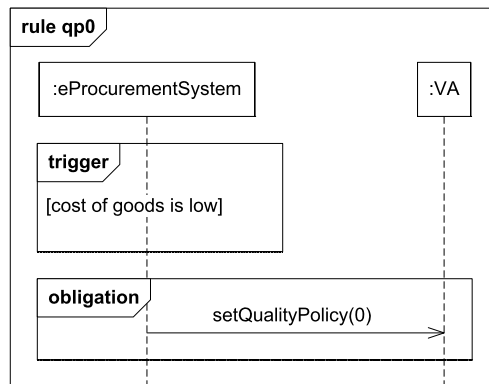


Fig. 15 Policy rule for the selection of quality policy level 20

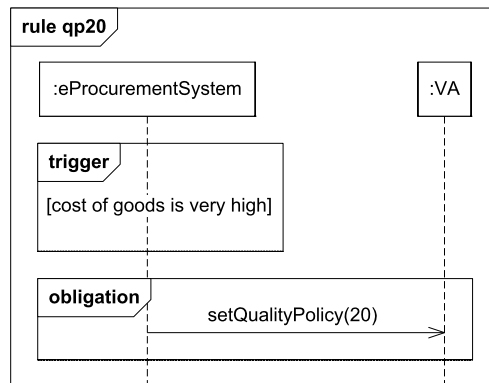
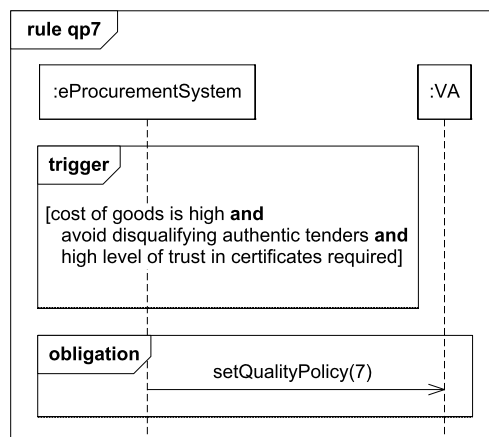


Fig. 16 Policy rule for the selection of quality policy level 7



is that the cost is high, that disqualifying authentic tenders should be avoided, and a high level of trust in certificates is required. Depending on which one of these three conditions that is satisfied, the eProcurement system *must* use either quality policy level 0, 7, or 20.

Table 4 The number of hours used on the trust analysis, not including writing of a final report

	Meetings	Preparations
Analysts	27	100
Participants	20	–

6 Evaluation of the trust analysis method

In this section we evaluate the performance of the trust analysis method in the industrial project with respect to the criteria presented in Sect. 4.3.

EC1: The trust analysis should provide the customers with a good basis for making trust policy decisions. This means that sufficient information about the impact of the potential alternatives must be provided.

The project gave strong indications that the trust analysis method is feasible in practice. We went through the various steps of the method (with exception of Step 2.2) in close interaction with the industrial representatives of the customer and delivered a result in the form of a policy that we believe gives an institution making use of eProcurement system with a VA service useful input on selecting the right quality policy level. Based on models developed in the modeling step of the method we collected expert judgments and documented them in the models.

Of course an industrial case like this can never give solid repeatable evidence of anything. There are too many factors influencing what happens. In the case of our project it may be argued that we should have used historical data rather than expert judgments, but such data were not available. It may also for example be argued that we should have had involvement of representatives of an eProcurement institution, and having the inventors of the trust analysis method in the analyst team is of course also rather extraordinary.

EC2: The trust analysis should be cost effective.

The trust analysis was carried out in a series of five meetings, each of which took about 1.5 hours. Typically, four analysts and two to four participants/representatives of the customer took part in the meetings. In addition, the analysts spent time between the meetings developing models and preparing the next meeting. Table 4 shows an estimate of the total amount of time spent on the trust analysis. Note that time spent on writing a final report is not included in the numbers—this depends heavily on the type of report the customer wants. There are some issues that must be taken into consideration when evaluating these numbers. Firstly, this was the first time the trust analysis method was applied to a real industrial case. Hence, even though the analysis team included authors of the paper [17] proposing the trust analysis method, none of the analysts had any experience with applying the method in a realistic setting. It can reasonably be assumed that the process will be more effective as the analysts gain experience with applying the trust analysis method. Furthermore, the reason for having as many as four analysts was a desire to learn as much as possible from this first application of the method. Normally, we believe two analysts would be enough.

Based on the experience gained, we believe that it should be possible to carry out this kind of analysis with within a time frame of ca. 80 man-hours spent by analysts

(not including writing a final report) and ca. 20 man-hours spent by participants. Whether this counts as being cost effective has to be evaluated in light of the values at stake in the target of analysis.

EC3: The modeling approach should be sufficiently expressive to capture the information, assumptions, and constraints of relevance.

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 models. The diagrams in Figs. 8–11 contain all the information that was finally used (and deemed relevant) for finding the best trust policy. These diagrams have been abstracted from more detailed models of the target. We also formalized the policy we recommended in the end.

EC4: The models should be comprehensible for the participants of the trust analysis.

During the meetings, models were presented and explained by an analyst in order to validate the correctness of the models. There were many instances where the participants pointed out parts of a model that did not correctly represent the target, provided additional information, or asked relevant questions about some detail in a model. This indicates that the models were in general comprehensible for the participants, and our experience is that the models served well as an aid in establishing a common understanding of the target between the participants and analysts. 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 a more diverse group. Note also that we do not know whether the models would have been well understood by the participants without any guidance or explanation, as all the models were presented by an analyst. In particular with respect to subjective diagrams and their relation to the objective diagrams, we believe it is necessary to have an analyst explain the diagrams in order for them to be understood by the participants if they have no prior experience with the notation, other than some background in UML.

There was one aspect of the models that proved hard to understand for the participants. This occurred when the operands of `par` operators contained more `par` operators. In Fig. 10 the `par` operator contains references to the diagram in Fig. 11, which again contains a `par` operator with another `par` operator inside one of its operands. This nesting of operators made it hard for the participants to understand exactly what each of the alternatives represented. In order to explain, one of the analysts drew a tree-structure where the root represented the outermost `par` operator and each branch represented a `par` operand. Based on this experience, we believe that the presentation style of UML interaction overview diagrams are better suited than sequence diagrams to present cases with nested alternatives. Interaction overview diagrams have the same kind of semantics as sequence diagrams and are often used in combination with sequence diagrams, but nested alternatives are represented syntactically by a branching point (the operator) with branches (the operands), rather than boxes inside boxes.

7 Conclusion and related work

The paper has presented experiences from using a UML-based method for trust analysis in an industrial project focusing on the modeling and analysis of a public eProcurement system making use of a validation authority service for validating electronic certificates and signatures. The contributions of the paper include:

1. a detailed account of how the method proposed in [17] scales in an industrial context; in particular, we have illustrated the specific constructs for capturing
 - beliefs of agents (humans or organization) in the form of subjective probabilities;
 - factual (or objective) probabilities of systems which may contain agents whose behavior is described in the form of subjective probabilities;
 - trust decisions in the form of policy rules;
2. an evaluation of the feasibility of the method in an industrial context; in particular, it is claimed that
 - the project gave strong indications that the trust analysis method is feasible in practice;
 - this kind of trust analysis can be carried within the frame of 100 man-hours (not including writing of a final report);
 - there were no instances where the analysts were not able to capture the relevant information in the models;
 - the models to a large extent were comprehensible for the industrial participant with some experience in UML but no background in the specific extensions used by the method.

The method for trust analysis makes use of models that capture the subjective trust considerations of actors, as well as their resulting behavior. We are not aware of other approaches that combine these elements in this way. However, the issues of uncertainty, belief, and trust have received much attention in the literature. We now present a small selection of the proposed approaches.

Giorgini et al. [6] presents a formal framework for modeling and analyzing trust and security requirements. Here, the focus is on modeling organizations, which may include computer systems as well as human actors. The approach is based on a separation of functional dependencies, trust, and delegation relationships. Trust and security requirements can be captured without going into details about how these will be realized, and the formal framework supports automatic verification of the requirements.

An interesting approach to modeling and reasoning about subjective belief and uncertainty is subjective logic [7, 8], which is a probabilistic logic that captures uncertainty about probability values explicitly. The logic operates on subjective belief about the world. Different actors have different subjective beliefs, and these beliefs are associated with uncertainty. The approach makes it possible, for example, to calculate to what degree an actor believes that a system will work based on the actor's beliefs about the subsystems, or to calculate the consensus opinion of a group of actors. Subjective logic deals strictly with the actors' beliefs and reasoning, and does not address the question of how their beliefs affect their behavior.

The belief calculus of subjective logic can be applied in risk analysis to capture the uncertainty associated with such analysis, as shown in [9]. This is achieved by using subjective beliefs about threats and vulnerabilities as input parameters to the analysis. Through application of the belief calculus, the computed risk assessments provides information about the uncertainty associated with the result of the analysis.

With respect to situations in which the outcome of a choice of one actor depends on the subsequent choice of another actor, the field of game theory [4] is highly relevant. Game theory provides strategies for making rational choices with respect to desirable and undesirable outcomes from the point of view of the different players/actors. These potential outcomes are described by a payoff structure in terms of the loss and gain to which the various players are exposed; a rational player will seek the outcome with the best payoff for herself. Not surprisingly, game theory can also be applied to analyze trust, as shown by Bacharach and Gambetta [1]. They explain how the trustor's choice to trust or not, and the trustee's subsequent choice to deceive or not, can be modeled in terms of this rational choice theory.

A formal model for trust in dynamic networks based on domain theory is proposed by Carbone et al. in [2]. Here, trust is propagated through delegation in a "web of trust", where the trust of one actor is affected by the trust of other actors. An important contribution of the approach is the distinction between a trust ordering and an information ordering. The former represents degrees of trust, while the latter represents degrees of precision of information from which trust is formed. An interval construction is introduced to capture uncertainty, and a simple trust policy language is proposed based on the formal model.

Acknowledgements We would like to thank Aida Omerovic (SINTEF ICT) and Anette Andresen (DNV) for participating in discussions and commenting on the work in the industrial project.

Open Access This article is distributed under the terms of the Creative Commons Attribution Noncommercial License which permits any noncommercial use, distribution, and reproduction in any medium, provided the original author(s) and source are credited.

References

1. Bacharach, M., & Gambetta, D. (2001). Trust in signs. In *The Russel Sage Foundation Series on Trust: Vol. 2. Trust in society* (pp. 148–184). Thousand Oaks: Russel Sage Foundation.
2. Carbone, M., Nielsen, M., & Sassone, V. (2003). A formal model for trust in dynamic networks. In *Proceedings of the international conference on software engineering and formal methods (SEFM'03)* (pp. 54–61). New York: IEEE.
3. European Dynamics S.A. Functional requirements for conducting electronic public procurement under the EU framework (vol. 1). <http://ec.europa.eu/idabc/servlets/Doc?id=22191>, January 2005. Accessed: January 19, 2009.
4. Fudenberg, D., & Tirole, J. (1991). *Game theory*. Cambridge: MIT Press.
5. Gambetta, D. (1988). Can we trust? In *Trust: making and breaking cooperative relations* (pp. 213–237). Oxford: Blackwell.
6. Giorgini, P., Massacci, F., Mylopoulos, J., & Zannone, N. (2004). Requirements engineering meets trust management: Model, methodology, and reasoning. In *LNCS: Vol. 2995. Trust management* (pp. 176–190). Berlin: Springer.
7. Jøsang, A. (2001). A logic for uncertain probabilities. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(3), 279–311.

8. Jøsang, A. (2007). Probabilistic logic under uncertainty. In *Proceedings of the thirteenth computing: the Australasian theory symposium (CATS2007). Conferences in research and practice in information technology (CRPIT)* (Vol. 65, pp. 101–110). Australian Computer Society.
9. Jøsang, A., Bradley, D., & Knapkog, S. J. (2004). Belief-based risk analysis. In *Proceedings of the Australasian information security workshop (AISW). Conferences in research and practice in information technology (CRPIT)* (Vol. 32, pp. 63–68). Australian Computer Society.
10. Jøsang, A., Keser, C., & Dimitrakos, T. (2005). Can we manage trust? In *Proceedings of the third international conference on trust management (iTrust), versailles* (pp. 93–107). Berlin: Springer.
11. Lysemose, T., Mahler, T., Solhaug, B., Bing, J., Elgesem, D., & Stølen, K. (2007). ENFORCE conceptual framework. Technical Report A1209, SINTEF ICT.
12. McNamara, P. (2006). Deontic logic. In *Handbook of the History of Logic: Vol. 7. Logic and the modalities in the twentieth century* (pp. 197–288). Amsterdam: Elsevier.
13. Object Management Group. UML Superstructure, V2.1.2. <http://www.omg.org/spec/UML/2.1.2/Superstructure/PDF>, November 2007.
14. Ølnes, J. (2006). PKI interoperability by an independent, trusted validation authority. In *Proceedings of the 5th annual PKI R&D workshop: making PKI easy to use* (pp. 68–78). NIST.
15. Ølnes, J., Andresen, A., Buene, L., Cerrato, O., & Grindheim, H. (2007). Making digital signatures work across national borders. In N. Pohlmann, H. Reimer, & W. Schneider (Eds.), *ISSE/SECURE 2007 securing electronic business processes: highlights of the information security solutions Europe/SECURE 2007 conference* (pp. 287–296). Wirsbaden: Vieweg.
16. The European Parliament and the Council. Directive 2004/18/EC of the European Parliament and of the Council of 31 March 2004 on the coordination of procedures for the award of public works contracts, public supply contracts and public service contracts. <http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2004:134:0114:0240:EN:PDF>, March 2004. Accessed: January 19, 2009.
17. Refsdal, A., Solhaug, B., & Stølen, K. (2008). A UML-based method for the development of policies to support trust management. In *Proceedings of the 2nd joint iTrust and PST conferences on privacy, trust management and security (IFIPTM'2008)* (pp. 33–49). Berlin: Springer.
18. Refsdal, A., & Stølen, K. (2008). Extending UML sequence diagrams to model trust-dependent behavior with the aim to support risk analysis. *Science of Computer Programming*, 74(1–2), 34–42.
19. Sloman, M. (1994). Policy driven management for distributed systems. *Journal of Network and Systems Management*, 2(4), 333–360.
20. Sloman, M., & Lupu, E. (2002). Security and management policy specification. *IEEE Network*, 16(2), 10–19.
21. Solhaug, B., & Stølen, K. (2009). Compositional refinement of policies in UML – Exemplified for access control. Technical Report A11359, SINTEF ICT.

Chapter 13

Paper E: Experiences from using indicators to validate expert judgments in security risk analysis

