A Graphical Approach to Security Risk Analysis

Doctoral Dissertation by

Ida Hogganvik

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

October 2007

Series of dissertations submitted to the
Faculty of Mathematics and Natural Sciences, University of Oslo.
No. 662

ISSN 1501-7710

Cover: Inger Sandved Anfinsen.
Printed in Norway: AiT e-dit AS, Oslo, 2007.

Produced in co-operation with Unipub AS.
The thesis is produced by Unipub AS 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.

Unipub AS is owned by
The University Foundation for Student Life (SiO)

Acknowledgements

First of all, I would like to thank my supervisor Ketil Stølen for many years of collaboration. Ketil Stølen has spent endless time on reading draft versions of papers and reports, providing detailed and highly valuable feedback and always being available to his students. Without all the effort Ketil Stølen has put in his supervising, this thesis work would never have become what it is today, and I am enormously grateful for this.

Jan Heim has in his role as co-supervisor provided valuable help and guidance, particularly related to my empirical investigations.

I would like to thank all my former colleagues at SINTEF Cooperative and Trusted Systems who have provided support, participated in data collection, brought up good ideas and been great discussion partners. This goes especially to Mass Soldal Lund and Heidi E. I. Dahl who both have been actively involved in this thesis work. I would also like to thank Folker den Braber, Gyrd Brændeland, Fredrik Seehusen, Bjørnar Solhaug, Ida Solheim and Fredrik Vraalsen.

I am very grateful to the people from the Institute for Energy Technology, the students from Østfold University College, UNIK and Institute for Informatics, and also former student fellows and others who have participated in my empirical investigations and thereby provided me with the necessary data for this thesis work.

This thesis work could not have been accomplished without the funding from the Norwegian Research Council.

I am also thankful to my current employer at Scandpower Risk Management who has supported me during the finalization of this thesis.

I would like to thank my family and friends who have supported me through all these years. Finally, I give my warmest thanks to my dear Nils Henrik. Without him encouraging me to keep on working, this thesis might never have been completed.

List of Original Publications

I. Ida Hogganvik and Ketil Stølen. ***On the Comprehension of Security Risk Scenarios***. In proceedings of the 13th International Workshop on Program Comprehension (IWPC'04), pages 115-124. IEEE Computer Society, 2004.

II. Ida Hogganvik and Ketil Stølen. ***Risk Analysis Terminology for IT-systems: Does it match intuition?*** In proceedings of the 4th International Symposium on Empirical Software Engineering (ISESE'05), pages 13-23. IEEE Computer Society, 2005.

III. Ida Hogganvik and Ketil Stølen. ***A Graphical Approach to Risk Identification, Motivated by Empirical Investigations***. In proceedings of ACM/IEEE 9th International Conference on Model Driven Engineering Languages and Systems (MoDELS'06) (formerly the UML series of conferences), volume 4199 of *Lecture Notes in Computer Science,* pages 574-588. Springer Verlag, 2006.

IV. Heidi E. I. Dahl, Ida Hogganvik and Ketil Stølen. ***Structured Semantics for the CORAS Security Risk Modelling Language***. In proceedings of 2nd International Workshop on Interoperability Solutions on Trust, Security, Policies and QoS for Enhanced Systems (IS-TSPQ'07), pages 79-92. Helsinki University Printing House, 2007.

V. Folker den Braber, Ida Hogganvik, Mass Soldal Lund, Ketil Stølen and Fredrik Vraalsen. ***Model-based security analysis in seven steps – a guided tour to the CORAS method***. Vol. 25 (1) of BT Technology Journal, pages 101-17. Springer Verlag, 2007.

VI. Fredrik Vraalsen, Tobias Mahler, Mass Soldal Lund, Ida Hogganvik, Folker den Braber, Ketil Stølen. ***Assessing Enterprise Risk Level: The CORAS Approach***. In *Advances in Enterprise Information Technology Security,* pages 311-333. Djamel Khadraoui and Francine Herrmann (Eds.), Information Science Reference, 2007.

VII. Ida Hogganvik and Ketil Stølen. ***Investigating preferences in graphical risk modeling***. Technical report, SINTEF A57, 2007.

The publications I-VII are available as Appendices A-G. In the case of publication I, II and IV we have included the full technical reports which are extended, slightly revised versions of the published papers.

Table of Contents

1 Introduction

Over time, we have gradually become more and more dependent on computerized systems of various kinds. Infrastructure like mobile phones, Internet, e-mail, online bank services etc. are important elements in our everyday life. In many cases we cannot imagine how to manage without them. As the systems increase in complexity, the number of security risks is most likely to rise. Security incidents occur on a daily basis within most companies, and the systems we surround us with are vulnerable to attacks [95]. In CSI/FBI Computer Crime and Security Survey for 2005 [35] 74% of the companies reported security incidents. The worst security breaches within UK companies are estimated to cost about 90.000£ (≈133.000€), and the companies experience about 19 incidents of that kind per year [128]. Virus attacks are the major source of the largest financial losses, with unauthorized access on second place [35]. Security incidents may have critical effects also for systems traditionally not associated with security breaches. In 2003 the Davis-Besse nuclear plant (Ohio, US) was hit by the SQL Slammer worm. The worm disabled crucial control systems for about five hours. Fortunately, the systems had analogue backups that remained unaffected. The so-called "northeastern blackout" in 2003 was a failure in the electrical power grid that left 50 million people in North America without power [127]. The incident affected transportation, communication, industry, water supply etc., and it took several days to restore everything back to normal operation. The failure was not caused by any form of malicious attack, but a contributing factor to the incident was a software bug.

The request for safe, secure and reliable systems have lead to a demand for good security risk analysis methods. The methods must handle detailed analyses of technical aspects, as well as more high-level enterprise level analyses. This means that the methods must be general, but not too general in order to provide sufficiently support to the analysis process. Many security risk analysis techniques include people familiar with the target of analysis as participants, under the guidance of an analysis leader. A well designed security risk analysis method should support the analysis process by facilitating communication, interaction and understanding between these participants. Security risk analysis is often considered to be complicated and time-consuming. The participants in a security risk analysis often have to deal with complex systems and advanced technology. In many cases the only visible part of the system analyzed is its interfaces and effects, while the internal structure is hidden. Consequently, the participants may experience difficulties in understanding and analyzing the system with respect to security risks. This makes room for errors and misunderstandings which may hamper the analysis process. Another important challenge of security risk analysis methods is how to facilitate the involvement of people with different backgrounds and competences. People who uses or maintain the system on a daily basis posses vital knowledge about how the system is actually used, not only how it is intended to be used. Combining this type of information with the knowledge of the system designers, developers and owners is believed to benefit the security risk analysis results. The challenge is then to facilitate communication and collaboration between these different groups of people when they meet in the security risk analysis setting. In this thesis both "security risk analysis" and its abbreviation "security analysis" will be used.

1.1 Our Work

Our work has been to develop a graphical approach to threat and risk modeling that supports the security analysis process. There exists no such thing as *the* correct representation of threats, vulnerabilities, risks etc. since this highly depends on the situation at hand, but it makes sense to look for the *preferable* representation with respect to a specific security analysis approach. In this work we target security analyses that use brainstorming sessions to identify and analyze security risks. These analyses are characterized by their involvement of people with thorough knowledge of the target of the analysis, like users, designers, developers and decision makers.

Our graphical approach to security risk modeling contributes to solving three issues related to security analysis. The first is *how to facilitate communication in a group consisting of people with different backgrounds and competences?* Our aim has been to provide the participants with a mean for communication that covers both technical and more high-level information, without being too complicated to understand. Offering a common communication basis will hopefully reduce misunderstandings and thereby give a more correct risk picture.

Second, *how to estimate the likelihood and consequences of the risks*? The existence of reliable data on which this can be based on is unlikely. The participants must use their expert knowledge, experience and familiarity with the domain to estimate both the likelihood and the consequences of incidents that might never happened to this day. The estimation process may easily become complex and difficult to follow, particularly when the number of risks is high. Our aim has been to offer a structured, graphical risk picture to make the complexity more manageable. A graphical representation may illustrate who or what caused the incidents and the weaknesses in the system that made them possible.

Third, *how to document the security analysis in a comprehensible manner?* The findings of a security analysis constitute vital information not only to the participants in the analysis, but to the organization as a whole. The results need to be documented in a way that is understandable even for those not involved directly in the analysis. Our aim has been to define a documentation method that should be more or less "self-explaining" and not rely on extensive training to be understood.

We believe our approach to security risk modeling will contribute towards making the above challenges more manageable. We have aimed to make a language that is easily understandable, even for people without training in modeling or security analysis.

1.2 The Contribution of the Thesis

The main contributions of this thesis are a set of new artifacts and the results from evaluating the new artifacts. First we summarize the artifacts, then the evaluation results:

1.2.1 New Artifacts

A) A conceptual foundation for security risk analysis:
We have specified main security risk analysis concepts and their relationships in a conceptual model. The conceptual foundation originates from the CORAS method and is based on international standards within security and risk analysis. A conceptual foundation may help to reduce misunderstandings with respect to interpretation of terminology. It may also contribute to a consistent use of terms both in the security analysis process and in security risk analysis modeling.

Described in: *Chapter 6 – Language Requirements Defined within a Quality Framework.*

B) A framework for evaluating the quality of security risk modeling languages:
To assess the quality of security risk modeling languages we have constructed a quality evaluation framework. The framework covers both practical modeling tasks as well as the more theoretically parts of a language like domain appropriateness and comprehensibility appropriateness. The framework consists of two main parts: (1) a detailed description of core security risk scenarios that a security risk modeling language should be able to express used as a "benchmarking test", and (2) a set of quality requirements based on an extended version of SEQUAL (quality framework for modeling languages). The framework has been designed with two goals; first to capture the rationale for our analytical evaluation. Second the framework makes it possible to compare the quality of different security risk modeling languages.

Described in: *Chapter 5 – Initial Investigations.*

C) A language for security risk modeling:
We present in our work a security risk modeling language that through its customized diagram types supports the entire security analysis process. Since it accompanies the CORAS method for security analysis it will in this thesis be referred to as the CORAS security risk modeling language (abbreviated to the CORAS language). The CORAS language has five diagram types: (1) asset diagram, (2) threat diagram, (3) risk diagram, (4) treatment diagram, and (5) treatment overview diagram. The asset diagram is used initially in the analysis to support asset identification, valuation and specification of risk acceptance levels. The threat diagram is the most central diagram type in the language. First, it is used in risk identification to support specification of threats, vulnerabilities, threat scenarios and unwanted incidents. Then, during risk estimation the threat diagram forms the basis for estimating likelihood and consequences of the different unwanted incidents. Finally, the threat diagram is the starting point for the treatment identification where treatments are added to the diagram. The risk diagram supports risk evaluation by specifying the risks and including which are acceptable and which are not. The treatment overview diagram is based on the treatment diagram and gives an overview of non-acceptable risks and the suggested treatments. This can be seen as a plan for mitigating the risks, and is suitable for presentation purposes. The CORAS language is not restricted to security analyses of information systems. We show this in *Appendix F* by using an early version of the language in an analysis of security-, legal- and trust issues related to collaborative engineering in virtual organizations within the aerospace industry.

Described in: *Chapter 8 – The CORAS Security Risk Modeling Language and Guideline, Examples of use in: Appendix H – Quality Evaluation of the CORAS Language, Appendix C – A Graphical Approach to Risk Modeling, Motivated by Empirical Investigations, Appendix E – Model-based Security Analysis in Seven Steps – a Guided Tour to the CORAS Method and Appendix F – Assessing Enterprise Risk Level: The CORAS Approach.*

D) A structured semantics for the security risk modeling language:
To ensure that the semantics of the CORAS security risk modeling language is understood in the same manner by everyone reading them, we have developed a textual syntax that maps from the graphical syntax to precise English. The mapping is based on the Backus–Naur form notation (EBNF) [81], and may also be seen as the first step towards a formal semantics.

Described in: *Chapter 8 – The CORAS Security Risk Modeling Language and Guideline (summary), Appendix D – Structured Semantics for the CORAS Security Risk Modeling Language*.

E) A guideline for the use of the security risk modeling language:
The full potential of a graphical security risk modeling language cannot be reached without providing a comprehensive guideline for how to use it. The guideline describes in detail how to create the different diagrams for each step during the security analysis process. In addition to realistic examples, it also provides guidance on how to conduct the analysis and what target descriptions that may be used. In *Appendix E* we provide a thorough, example driven introduction to a complete security analysis using the CORAS security modeling language, especially targeting practitioners. *Appendix C* contains a shorter example.

Described in: *Chapter 8 – The CORAS Security Risk Modeling Language and Guideline, Appendix C – A Graphical Approach to Risk Modeling, Motivated by Empirical Investigations (short version) and Appendix E – Model-based Security Analysis in Seven Steps – a Guided Tour to the CORAS Method (targeting practitioners).*

1.2.2 Results in the Form of Evaluations

We present two types of evaluations: investigations made during development to improve the artifacts, and evaluations of artifacts in their final form.

F) Empirical support for the conceptual foundation:
The conceptual foundation was revised on basis of the results from two empirical investigations with focus on ease of understanding. We first explored how risk analysis and security analysis concepts are understood based on their existence in every day language. The findings made us aware of particular difficult concepts that were considered when redesigning the conceptual foundation. In the second investigation we decided to use subjects with highly relevant backgrounds and competences, similar to potential users of a security risk modeling language. One of the most interesting findings was the discovery of the same pattern of difficult and easy concepts/relations as in the first study. This means that security risk analysis employs a conceptual foundation that may cause misunderstandings and confusion, even for highly skilled people.

Described in: *Chapter 5 – Initial Investigations (summary), Appendix A – Empirical Investigations of the CORAS Language for Structured Brainstorming and Appendix B – Risk Analysis Terminology for IT systems: Does it match Intuition?.*

G) Results from applying the quality framework:
We have applied the quality framework twice, first to evaluate the original UML profile [111, 112, 124], then to evaluate the new CORAS security risk modeling language. The results include complete modeling efforts of the core security risk scenarios and analytical evaluations according to the SEQUAL-based quality requirement framework. The results make it possible to compare the quality of the two security risk modeling languages.

Described in: *Chapter 7 – Evaluation of the UML Profile (summary, for full details we refer to [57]), Chapter 9 – Evaluation of the CORAS Language and Guideline, Appendix H – Quality Evaluation of the CORAS Language.*

H) Results from applying the language:
The CORAS security risk modeling language has from 2004 to 2006 been tested, evaluated and refined in six major industrial field trials. The security analyses have been conducted as part of the research project SECURIS (www.sintef.no/securis) which aims to investigate and improve the CORAS method by applying it in real security analyses. Each analysis required about 250 person hours from the analysis team and 50-100 hours from the client. The representatives of the client argued that the graphical language made it easier to actively involve the participants in the analysis and helped ensuring an effective communication between the analysis team and the participants. They also found the notation itself easy to understand and remember. It was considered to be a good way of visualizing threat scenarios and very suitable for presentations. According to one of the participants this type of visualization emphasizes the "message" or the purpose of the analysis.

Described in: *Chapter 9 – Evaluation of the CORAS language (field trials)*.

I) Empirical support for the use of visualization mechanisms:
It is well known that using visualization techniques correctly can improve a graphical representation. Information visualization techniques include among other the use of size, shape and color to make information more accessible to the reader. The study reported in *Appendix A* convinced us to use special risk-related symbols in our language since this improved the subjects' ability to complete tasks. The background for the investigation was related studies which claimed that graphical symbols in modeling languages make the diagrams more understandable. We compared basic UML icons with special risk related symbols. The results showed that one may further improve a UML based language by replacing the standard UML symbols with more specialized symbols that reflects the domain it is used for. An interesting finding in the second study (*Appendix G*) of graphical mechanisms was that people prefer textual information labels rather than just visual means like size, color and shape. We let a group of professionals within system and software engineering make the final design decision with respect to how the diagrams should look like. The results indicate that if using visual means like shape, size or color to convey meaning in diagrams they must be accompanied with an explanatory textual information label.

Described in: *Chapter 9 – Evaluation of the CORAS Language and Guideline (language design decisions), Appendix A – Empirical Investigations of the CORAS Language for Structured Brainstorming, Appendix G – Investigating Preferences in Graphical Risk Modeling*.

J) Results from applying the guideline:
The most complete example of using the guidelines is in the quality evaluation of the CORAS language, where it is used when modeling the core security risk scenarios (*Appendix H*). The description of the scenarios is provided as text only with no guiding or restrictions on to how it should be modeled. A smaller example is given in *Appendix C*. Since the guideline has been developed iteratively and improved for each of the SECURIS field trials, every field trial analysis report contains the results from a full application of the guideline, however these reports are confidential.

Described in: *Appendix H – Quality Evaluation of the CORAS Language, Appendix C – A Graphical Approach to Risk Modeling, Motivated by Empirical Investigations (smaller example)*.

1.3 The Structure of the Thesis

This thesis consists of the following chapters and appendices:

Chapter 1 – Introduction: this chapter provides an introduction to the research domain, the thesis contribution and the structure of the thesis.

Chapter 2 – Problem Area: this chapter introduces the main problems within our area of research, focusing on specification, documentation and communication of security risks. This is also where our success criteria are specified.

Chapter 3 – State of the Art: this chapter introduces the state of the art within security analysis methods and standards, graphical modeling techniques and information visualization mechanisms.

Chapter 4 – Research Method: this chapter describes the research methods we have used.

Chapter 5 – Initial Investigations: this chapter contains the finding from the initial empirical investigations made of security risk analysis terminology and the symbols in the previous version of the CORAS language. This is based on III and IV in page iii, and a full version can be found in *Appendices B* and *C*.

Chapter 6 – Language Requirements defined within a Quality Framework: this chapter gives the detailed requirements to an ideal security risk modeling language inspired by quality requirements for modeling languages within system development and also based on experiences from real security analyses in field trials.

Chapter 7 – Evaluation of the UML profile: this chapter presents a summary of the quality evaluation of the UML profile which was the starting point for our work on a graphical security risk modeling language. A full version can be found in [57].

Chapter 8 – The CORAS Security Risk Modeling Language and Guideline: this chapter presents the syntax and semantics of the language, including a detailed example driven modeling guideline. This is particularly based on II, IV and V, and the full versions can be found in *Appendices C, E,* and *F*.

Chapter 9 – Evaluation of the CORAS Language and the Guideline: this chapter summarizes the evaluations of the CORAS language using questionnaires, in field trials and applying the quality framework.

Chapter 10 – Discussion: the final chapter discusses our contribution with respect to the success criteria from *Chapter 2*.

References, Table of Figures and *List of Tables*.

Appendix A – Empirical Investigations of the CORAS Language for Structured Brainstorming. This is an initial exploration of the understanding of risk analysis concepts. It also includes the experiment with special graphical symbols versus traditional UML symbols. Corresponding to I in page iii.

Appendix B – Risk Analysis Terminology for IT systems: Does it match intuition? This technical report presents the investigation of the understanding of the revised conceptual foundation. Paper II in page iii corresponds to this report.

Appendix C – A Graphical Approach to Risk Modeling, Motivated by Empirical Investigations. This paper provides an introduction to the complete CORAS language, including the conceptual foundation and the guideline. Corresponds to III in page iii.

Appendix D – Structured Semantics for the CORAS Security Risk Modelling Language. This technical report defines the structured semantics for the CORAS language. Corresponds to IV in page iii.

Appendix E – Model-based Security Analysis in Seven Steps – a Guided Tour to The CORAS Method. An example of a complete security analysis, including modeling guidelines for the CORAS language. Corresponds to V in page iii.

Appendix F – Assessing Enterprise Risk Level: The CORAS Approach. An example of a full security analysis, using the previous version of the CORAS symbols. Corresponds to VI in page iii.

Appendix G – Investigating Preferences in Graphical Risk Modeling. Reports the design decisions based on the findings from the experiment. Corresponds to VII in page iii.

Appendix H – Quality Evaluation of the CORAS language: contains the evaluation of the CORAS language with respect to its ability to model a set of core security risk scenarios and fulfillment of language quality requirements.

1.4 How to Read the Thesis

According to which artifact you are interested in, we recommend four alternative strategies for reading the thesis:

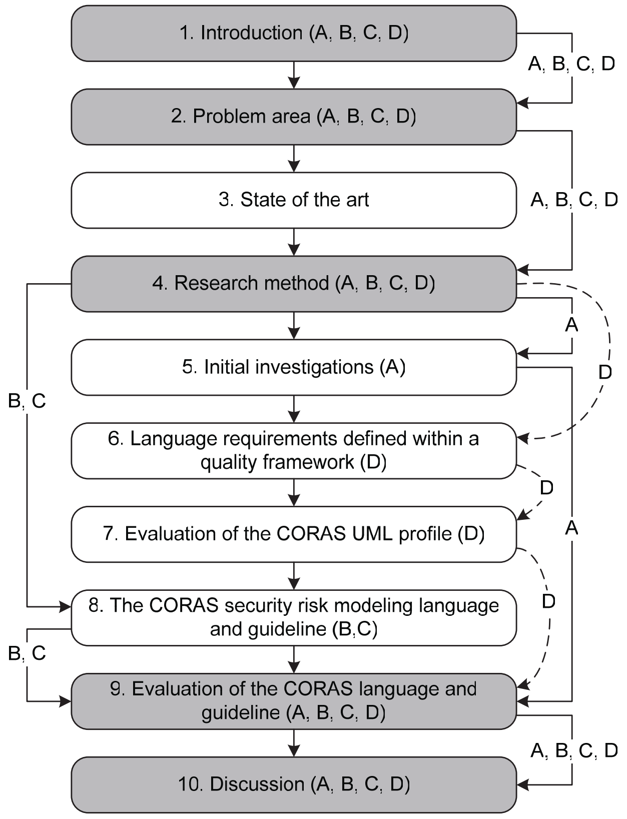
A) The development of the conceptual foundation is described in *Chapter 1, 2, 4, 5, 9* and *10*.

B) If you are mainly interested in the CORAS language itself, you should read *Chapter 1, 2, 4, 8, 9* and *10* since they take you through the most essential chapters.

C) For a description of the development of the modeling guideline, including the guideline itself you should read the same chapter as for strategy B.

D) If you like to explore the quality evaluation framework for security risk modeling languages, we recommend *Chapter 6*. The development strategy is described in *Chapter 4* and examples of how we have applied it are found in *Chapter 7* (a summary, for full details we refer to [57]) and in *Chapter 9* (full details in *Appendix H*).

In the figure below, we have illustrated the different strategies for reading the thesis using the letters A, B, C and D. The gray color indicates chapters that are recommended for all strategies.



1. Introduction (A, B, C, D)
A, B, C, D
2. Problem area (A, B, C, D)
3. State of the art
A, B, C, D
4. Research method (A, B, C, D)
A
D
5. Initial investigations (A)
B, C
6. Language requirements defined within a quality framework (D)
D
7. Evaluation of the CORAS UML profile (D)
A
D
8. The CORAS security risk modeling language and guideline (B,C)
B, C
9. Evaluation of the CORAS language and guideline (A, B, C, D)
A, B, C, D
10. Discussion (A, B, C, D)

Figure 1 – Reading guide

To avoid confusion while reading, it is important to remember the difference between the UML profile and the CORAS language:

The UML profile	Refers to the *UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms* [124] that was developed as part of the CORAS project (2000-2003). The language is based on the use case notation from the *Unified Modeling Language (UML)*.
The CORAS language	Refers to the language presented in *Chapter 8* developed as part of this thesis. The language does not conform to UML, but is inspired of the UML profile and therefore have similarities with the UML use case notation.

2 Problem Area

In this chapter, we first introduce the problem area, and then we formulate and motivate a list of success criteria for the work presented in this thesis.

2.1 Introduction

We all take actions to avoid security risks in our daily life. It may be as simple as locking the office door when leaving for the day, protecting the laptop with a password or avoiding insecure web browsers when accessing the net bank. For your home computer, it is quite manageable to enforce good security routines like updating the virus protection, activating the firewall and keeping updated on relevant security patches. The situation becomes more complex if the system is a major information system, providing business critical information through several interfaces to a large user base. To ensure a sufficient level of security for such a system, there is need for a thorough security risk analysis method.

The security field is rapidly changing and it can be challenging, if not impossible for a single person to keep updated on all relevant security issues for a complex system. This is why a full-scale security risk analysis is typically conducted by a group of people with competence on different areas, for example experts in security technology, system experts, users and decision makers. The idea is that a group of people with different competences will view the system from different perspectives, and therefore identify more and possibly other risks than a more heterogeneous group. When we in the following speak of security risk analysis, we refer to this kind of analysis. It is typically used in cases where security is crucial to an organization, either in their daily business or with respect to the product or service they deliver. The analysis may take hundreds of hours and is consequently an expensive operation. Any means that may contribute to reduced costs, without sacrificing the quality of the security risk analysis results, are more than welcome.

A security risk analysis (or just security analysis) may consist of several "structured brainstorming" sessions. Brainstorming is often associated with a creative process of coming up with new ideas to solve a problem. A structured brainstorming is a methodical and step-wise "walk-through" of the target of analysis, particularly useful for identifying potential security risks and estimating likelihoods and consequences of these risks. The brainstorming session may be structured according to special security aspects, the physical organization of the system, the work processes or similar. In brainstorming sessions within security analyses, the participants are asked to put forward as many threats, vulnerabilities and risks as they may think of within the context of the analysis. These are in turn assessed to find the best way to deal with them. Structured brainstorming may be used to assess systems of various size and complexity, including organizational procedures or routines in which the system is used. This does not mean that the same brainstorming approach is applied *directly* to all types of systems; some adaptation is needed to fit the particular domain.

2.2 The Need for a Special Purpose Language

A security analysis based on structured brainstorming is a team effort that involves a group of people that must communicate and cooperate with each other. The topics discussed may be complex and difficult. A common method for explaining complicated problems is to draw sketches or illustrations on a blackboard. For instance, in software engineering graphical models are often used to ease the explanation of complex systems (e.g. [105]).

The effectiveness of such drawings depends however on the artistic skills of the modeler, the group members' individual interpretations of the drawings and other factors. The potential sources of misunderstandings regarding a drawing are therefore numerous. To reduce this weakness and increase the comprehension and preciseness of security analysis documentation, we see the need for a well-defined and easily understandable graphical notation for documenting security relevant scenarios. Moreover, this notation (or language) must be supported by a carefully designed modeling guideline, which connects the language to the security analysis method. In the remaining of this chapter, we refer to this language and its guideline as the "security risk modeling approach".

Documentation from one step in a security analysis typically provides input to the next step. The security risk modeling approach should also follow this structure. This implies that the different steps in the analysis should be documented using customized diagram types. The diagrams should build on each other like the steps in the security analysis process, meaning that it should be possible to extend the diagrams from one step with more information in the next step. Since the brainstorming session is an interactive process, it must be possible to modify and update diagrams "on-the-fly". This may be very challenging for the modeler, since it is carried out while the participants are watching. Nevertheless, it gives the participants an opportunity to instantly validate and correct the diagrams, possibly providing additional information, which in turn gives a representation that is as complete, and correct as possible.

Threat scenario identification and risk estimation require particular support. Threat scenarios describe all the different ways a risk may occur, something that can be complex and difficult to follow (illustrated in Figure 2).

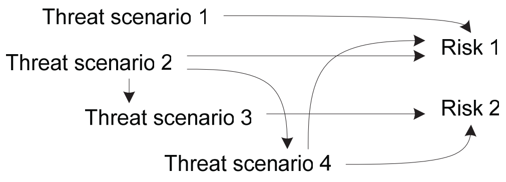


Figure 2 – Risks and threat scenarios

Risk estimation is concerned with determining the likelihoods and the consequences of risks. The likelihood of a risk is based upon the combined likelihood of all threat scenarios leading up to the risk. In theory, this may seem unproblematic, but practice has shown that reliable data to base estimates upon hardly exists. Sometimes statistical material can be used as a starting point or a suggestion, but someone familiar with the target of analysis and the domain must always judge its relevance. Quite often, the participants are unable to estimate the likelihood of a threat scenario, and consequently the likelihood of the risk, in terms of an exact numeric value. A solution may be to use intervals, where the participants only judge in which interval category a likelihood belongs to. This implies that a suitable modeling approach must support likelihood estimation on the basis of interval values, as well as exact, numeric values if they exist.

Security analysis documentation should serve multiple purposes, for example as highlighted in [6, 131]. As already explained, it should support specification of intermediate findings at the various methodological steps. The documentation should also show that the security analysis process has been conducted properly, and provide evidence of a systematic approach to risk identification and analysis. This may be required in the case of certification according to quality standards like ISO9001 [71] or BS7799-2 [26]. The documentation represents a record of security issues that can be shared with the rest of the organization and thereby extend the organization's knowledge base to become more robust against security threats. Furthermore, often the results from a security analysis should be presented to the management, board of directors or other parties that have not been involved directly in the analysis. The documentation should ideally provide them with a risk management plan, showing the risks, the main vulnerabilities and how they should be mitigated. This highlights of course the importance of having documentation in a form that requires only a minimum of explanation to understand.

Traditional risk analysis documentation has mainly focused on tables and text. This may require the reader to go through an extensive amount of documentation in order to comprehend the security analysis findings. Getting an overview of the findings may be especially challenging for people who only participate in parts of the analysis. Text and tables do not provide sufficient support to the participants during the analysis, since it requires careful attention to everything that is written. When a table grows it may become difficult to keep record of all relevant information and the participants may easily loose track of what has been documented. A graphical documentation approach may provide an overview of the complete risk picture in a more compact and space effective manner than text and tables, which is especially suitable for use in group work. Well established modeling techniques within risk analysis (e.g. fault trees [66] and event trees [64]) are more commonly used for calculating probabilities, than describing the overall risk picture. A new modeling approach must relate to these existing modeling techniques, by either using them as complementary modeling techniques or being able to express the same information. In fact, it may be said that the new modeling approach is an effort to better integrate existing notations. We do not argue for a complete replacement of text, tables and existing notations with something new, but rather build on existing techniques to bring security risk analysis documentation one step further. The intention is to combine traditional risk documentation techniques with state-of-the-art modeling techniques, to support the entire security analysis process.

2.3 Problem Characterization

Above we provided an overview of the overall problem area. In the following, we narrow this down to four main "work packages" addressed by this thesis. For each of them we formulate a set of criteria, providing a characterization of what it means to successfully complete the work package.

2.3.1 A Common Basis for Security Analysis Terminology

As mentioned previously, the participants in a structured brainstorming may have very different backgrounds and competences. The authors of [131] recommend these roles to be represented when assessing IT systems: (1) user or intended user of the system assessed, (2) experts on relevant aspects of the system and (3) system designer. There must also be a person responsible for facilitating the process (analysis leader), and a secretary whose responsibility is to document the findings. To ensure an effective analysis process, the participants must quickly become familiar with the security analysis terminology. This means that the terminology should be as easy to understand as possible. Moreover, this terminology must be reflected in the security risk modeling approach. There are several international standards relevant to risk- and security analysis (see *Chapter 3*). However, these standards have often been made for specific purposes by different standardization organizations, and may not focus on the same aspects. To help achieving a consistent and clear terminology for use in security analysis, there is a need for a conceptual foundation of security risk related terms, building on the most relevant of these standards. An important prerequisite for the usefulness of such a conceptual foundation is that the definitions are as close as possible to how the concepts are interpreted in everyday language. At the same time, the definitions should not directly conflict with relevant standards. This will facilitate integration with other security analysis methods and avoid misunderstandings among the users. A conceptual foundation defined as a list of definitions is not sufficient. There should also be a description of how the various concepts relate to each other. A concept may have relations to several other concepts, and this must be expressed in a simple manner, preferably illustrated graphically. This motivates the following success criterion:

1. The work of this thesis should include a conceptual foundation for security analysis, that:
    a. specifies the main security analysis concepts and their relationships
    b. uses a terminology that is easy to understand
    c. uses a terminology that is in accordance with international standards
    d. is described in a simple and understandable manner
    e. provides a solid, underlying basis for a security risk modeling approach

2.3.2 A Language for Specifying and Documenting Security Analysis Findings

The need for a customized security risk modeling approach that includes both a language/notation and a set of modeling instructions has already been highlighted. The language should have a well-defined syntax that specifies its components and rules for how they may be combined in diagrams. This will ease the learning, since the models will build on a known set of rules that is recognizable for the reader, even if the actual information modeled in the diagram is unknown. The syntax should make use of best practice within information visualization to help illustrating particular important parts of the security analysis findings. This means that mechanisms like size or color may be used to convey specific meanings in the models. The language should also be designed in interaction with real users to include their preferences, and to ensure that it fulfils their needs. Well-known risk modeling notations should be considered for inclusion in the language. If this is impractical, one should try to include their underlying ideas even though they will be expressed with a different syntax. The language also needs a precise semantics that defines how the diagrams should be understood. The semantics should be structured in the sense that the meaning of an arbitrary, syntactically correct diagram can be generated schematically. Moreover, to make the semantics accessible for regular users of the language we would like the generated meaning of a diagram to be expressed in English. The structured semantics will help ensuring that all readers understand the models in the same manner. This motivates the following success criterion:

2. The work of this thesis should specify a language for describing security risks, that:
    a. is suitable for use in structured brainstorming sessions
    b. is easily understandable for the participants in the brainstorming, including those who receive the analysis results afterwards
    c. has a precise syntax, meaning its design should be based on:
        i. best practice within information visualization
        ii. experiences with realistic security risk scenarios
        iii. users' preferences
        iv. existing risk modeling techniques
    d. has a structured semantics that translates arbitrary diagrams into English

e. supports and documents the different steps in the security analysis process

2.3.3 Modeling Instructions for the User

To utilize a language to its full potential, the person who uses it (typically the one who is responsible for documenting the analysis) must be provided with instructions for how it should be used. This is not only valuable to the modeler, but also important for the quality of the models. A set of instructions for how to model, may help achieving models that are more complete and consequently the analysis will be better documented. The instructions should be in the form of a guideline, which in addition to rich and detailed modeling instructions also provides realistic modeling examples from real security analysis situations. The guideline should first of all address inexperienced modelers and those unfamiliar with security analysis. To give this group as much support as possible, the guideline should if possible include recommendations from practical use of the modeling approach. Since the modeling tasks are tightly coupled to the analysis process, the guideline should follow the same structure, with relevant guidance for each step. This motivates the following success criterion:

3. This thesis work should provide a modeling guideline for the modeler, that:
    a. is rich and detailed with realistic examples
    b. addresses non-experienced modelers
    c. has recommendations based on user experiences
    d. follows the security analysis process step by step

2.3.4 A Way of Evaluating the Quality of Security Risk Modeling Approaches

When developing this security risk modeling approach, the requirements should be properly defined, similarly to a requirement specification for e.g. a software product. The degree of fulfillment of this specification can then be used as a measure of the quality of the modeling approach. A considerably portion of these requirements will however be of a general character and can therefore be seen as a quality assessment framework for any security risk modeling approach. People who need to judge the quality of, or compare, different security risk modeling approaches, may use this framework. There exist frameworks for comparing *security analysis methods* [158] and for evaluating the quality of *modeling languages in general* (Sect. 6.2), but to our knowledge there are none for evaluating the quality of security risk modeling approaches. Such a quality evaluation framework should cover model quality in general, and aspects that are specific to security risk analysis. Examples of the latter are the strong relationship between the different steps in the analysis process and the modeling approach, the special terminology, and the focus on structured brainstorming. The framework should also have a modeling test case that may be used as a "benchmarking test" to evaluate the approach's ability to model typical and realistic security risk scenarios. This motivates the following success criterion:

4. This thesis work should provide a framework for evaluating the quality of security risk modeling approaches, that:
    a. covers quality requirements related to modeling languages in general
    b. includes quality requirements special to security risk analysis
    c. includes a modeling test case representing core security risk scenarios

3 State of the Art

This state of the art is divided into 6 sub sections: (1) risk management, (2) security risk analysis methods, (3) standards relevant to security- and risk analysis, (4) risk- and system analysis techniques, (5) risk documentation techniques and (6) information visualization techniques.

3.1 Risk Management

Domains like finance, safety/reliability, and security all use the term risk. Even though our work addresses security risk analysis, it will cover aspects of the safety/reliability domain since we use or build on their methods, terminology and more. When speaking of approaches, methods or techniques that do not belong to a particular domain we will use the terms risk analysis, risk analysis technique etc. If the focus is particularly on security, we will use "security risk analysis" or just "security analysis". Financial risk analysis will not be covered by this thesis.

It may be difficult for the reader to clearly understand the differences between risk management, risk analysis, risk analysis techniques etc. The problem lies in the lack of precise definitions. Something presented as a method in one context may be regarded a framework, methodology or technique in another. To make this easier we will use the following distinctions and classifications in this state of the art (framework and methodology fall within method):

- **Risk management** is the culture, processes and structures that are directed towards realizing potential opportunities whilst managing adverse effects [6]. In practice, this means the use of one or more methods to identify and evaluate new risks, and also monitor, review, and communicate information about existing risks.
- **Risk analysis method/security risk analysis method** refers to a method used in the systematic process of understanding the nature of and deducing the level of risk [6]. In practice, this refers to the overall assessment of a system, or part of a system, that includes risk identification and evaluation, and also suggestions for risk mitigation. This kind of method often makes use of one or more risk analysis techniques.
- **Risk analysis technique/security analysis technique** refers in this thesis to analysis techniques that may be used in various parts of a risk analysis method. These techniques are typically used to e.g. estimate the likelihood of risks, analyze potential outcomes of a risk, look for particular vulnerabilities, and more.

The domain, objective and the level of detail of an analysis influences which approach for managing and controlling risks that is appropriate to use. Often it is necessary to tailor a risk management approach or strategy to meet specific usage requirements. The two main strategies within risk management are *asset-based* and *threat- and vulnerability-based* risk management [133]. The asset-based risk management focuses on identifying the valuable aspects of the system (assets), and then assesses how they may be protected from threats and risks. Asset-based risk management typically involves so-called bottom-up risk analysis techniques like FMEA/FMECA (Failure Mode Effect Analysis / Failure Mode Effect and Criticality Analysis) (Sect. 3.4.2) and HazOp (Hazard and Operability) analysis (Sect. 3.4.3). Threat- and vulnerability-based risk management aims to identify the threats and vulnerabilities of a system first, and then look at the risk they pose in a top-down manner. An example of a threat and vulnerability-based risk analysis technique is SWOT (Strengths, Weaknesses, Opportunities and Threats) which helps identifying the general areas of concern regarding a system (Sect. 3.4.1).The two categories of risk management have their strengths and weaknesses and the optimal solution is probably to use a combination of them. Often a top-down threat and vulnerability assessment is conducted initially to establish the scope of the analysis and characterize the overall risk picture. Then a more detailed bottom-up approach can be used to identify assets and their risks, using the findings from the top-down approach.

3.2 Security Risk Analysis Methods

In this section we first present the starting point of our work, the CORAS security analysis method (3.2.1), and then other relevant security analysis methods. We survey the main differences and similarities between these methods and CORAS, and summarize this in the end of the section. One important common factor for these methods is that they employ some form of structured brainstorming in the identification, estimation, evaluation or treatment of risks.

3.2.1 The CORAS Security Analysis Method

The security analysis method CORAS (http://coras.sourceforge.net) was developed in the EU-funded CORAS project (IST-2000-25031) from 1999 to 2003. The project had eleven partners representing the United Kingdom, Greece, Germany and Norway. One of the aims of the project was to gather well-known risk analysis techniques into an integrated security risk analysis method called the CORAS security analysis method. The method should make it easier to apply several risk analysis techniques in an integrated manner with particular focus on security risks analysis.

The CORAS method employs several international standards [7, 65, 66, 76, 82, 86] and the process follows the Australian / New Zealand Standard for Risk Management [7] (Sect. 3.3.1). This CORAS method has five main phases: *(1) establish context, (2) identify risks, (3) analyze risks, (4) evaluate risks and (5) treat risks.* Each phase has different purposes and produces different types of documentation. Structured brainstorming is used in risk identification, risk estimation and in the risk treatment phases.

Another major result from the CORAS project was the graphical security risk modeling language based on UML, the UML profile (Sect. 3.5.1). In addition to the methodological approach and the graphical language, the CORAS project resulted in a library of reusable experience packages, a computerized integration tool, an XML mark-up for exchange of risk assessment data and a vulnerability assessment report format (Figure 3), but these are not relevant to this thesis work. The work with the CORAS method has been continued within the research project SECURIS (NFR 152839/220) which runs from 2003 to 2007. One of the main objectives of SECURIS is to improve the security analysis method by applying it in industrial field trials.

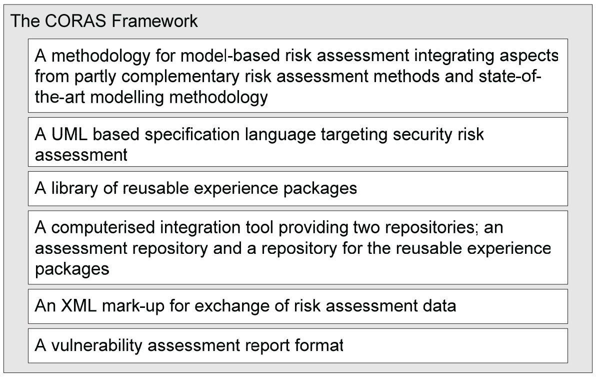

The CORAS Framework
A methodology for model-based risk assessment integrating aspects from partly complementary risk assessment methods and state-of-the-art modelling methodology
A UML based specification language targeting security risk assessment
A library of reusable experience packages
A computerised integration tool providing two repositories; an assessment repository and a repository for the reusable experience packages
An XML mark-up for exchange of risk assessment data
A vulnerability assessment report format

Figure 3 – Results from the CORAS project

3.2.2 OCTAVE

OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation) [3] is a risk-based strategic assessment and planning technique for security. OCTAVE is conducted in three phases: (1) identify critical assets and the threats to those assets, (2) identify the vulnerabilities that expose the assets to threats, and (3) develop an appropriate treatment strategy (Figure 4).

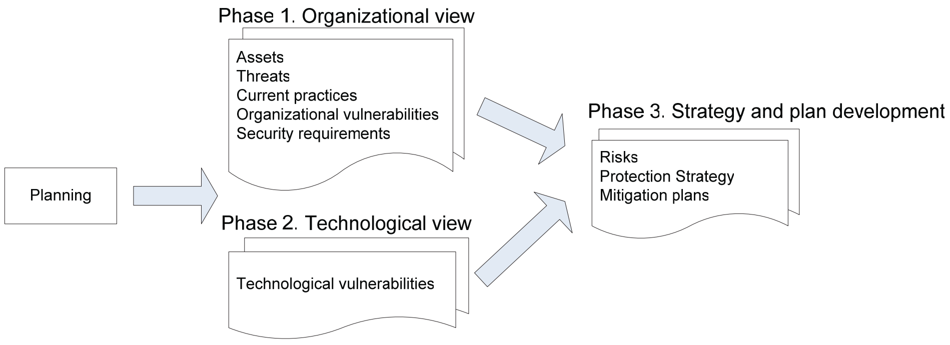

Phase 1. Organizational view
Assets
Threats
Current practices
Organizational vulnerabilities
Security requirements
Planning
Phase 2. Technological view
Technological vulnerabilities
Phase 3. Strategy and plan development
Risks
Protection Strategy
Mitigation plans

Figure 4 – The OCTAVE process

Phase 1 normally involves two workshops, the first with senior management to define the scope of the analysis, and the second with staff that have a more technical expertise on the target of analysis. The workshops may have a form of a structured brainstorming where people with different competences and backgrounds participate. The intermediate findings are documented in tables and form the basis for developing asset-based threat profiles using a simple graphical tree-structure (described in Sect. 3.4.8). The approach in OCTAVE is quite similar to the one used in CORAS.

3.2.3 CRAMM

CRAMM (CCTA Risk analysis and Management Method), is the UK Government's Risk Analysis and Management Method [8]. It is owned by the UK government's Security Service, but managed by Siemens/Insight (http://www.cramm.com). In CRAMM, *risk analysis* is identification and assessment of security risks while *risk management* is concerned with identifying appropriate countermeasures, or treatments for those risks. In our work the term security analysis covers the complete process from risk identification to treatment identification. Risk management according to CRAMM includes three phases: (1) asset identification and valuation (including dependencies between assets), (2) threat and vulnerability identification and (3) treatment (countermeasure) identification (Figure 5). The information is gathered through interviewing the owners of the assets, the users of the system, the technical support staff, and the security manager. In this manner, CRAMM is more like a review of the security of a product, conducted during system development or for an already running system.

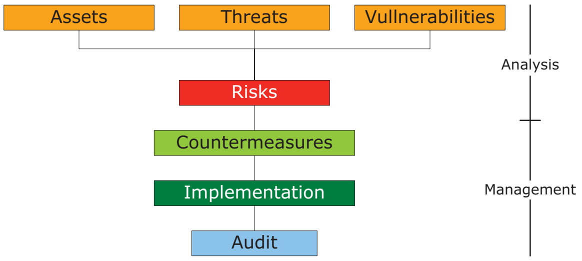


Figure 5 – The CRAMM process

The documentation produced during a CRAMM review uses a standardized CRAMM format, mostly in the form of specialized tables. CRAMM may help an organization to achieve compliance with ISO17799 [82], and the outcome is compliant with the mandatory documentation needed to achieve ISO27001 certification (BS7799-2) [26, 75]. The concepts and activities in CRAMM were a source of inspiration in the CORAS project.

3.2.4 Facilitated Risk Assessment Process

Peltier Associates (http://www.peltierassociates.com) develops the Facilitated Risk Assessment Process (FRAP). A risk assessment according to FRAP focuses on security aspects of systems or business processes. The assessment team consists of representatives with competence on technical aspects, as well as business- and management aspects. The FRAP has focus on threats, vulnerabilities and consequences towards data integrity, confidentiality and availability. A FRAP consists of the following three phases:

- Pre-FRAP meeting: the objective of this meeting is to decide on the system description and scope of the assessment, as well as assembling an assessment team.
- The FRAP session: this phase consists of three activities; first one decides the roles each participant will have in the brainstorming session, reviews and agrees on the definitions and scope of the risk assessment. Second, one conducts the actual brainstorming to identify potential risks within the scope of the assessment, and third one prioritizes the identified risks according to how vulnerable the system is, and what impact the risks may have. When the risks are sufficiently specified, the participants may also suggest possible controls or treatments for the risks.
- The post-FRAP meeting(s): these meetings aim to further analyze the information gathered at the FRAP session. The outcome is an overview of risks and how they should be mitigated by existing or new controls (treatments). The final report contains a complete documentation of the process, including an action plan for the recommended treatments.

3.2.5 Microsoft's Security Risk Management

As the name indicates, the Microsoft security risk management process [120] includes more than just a risk analysis method. The process consists of four phases (Figure 6, from [120]), where the first and the second correspond to our interpretation of a risk analysis method.

- 1st phase - assessing risk: during this phase one gathers data about assets, threats, vulnerabilities, existing security controls and suggested treatments. This information is then analyzed in facilitated discussions (what we call structured brainstorming sessions) and the outcome should be a list of risks.
- 2nd phase - conducting decision support: the list of risks from the previous phase function as input to an assessment of the various control or treatment solutions that are proposed. The outcome of this phase is a set of treatment options that are considered to be appropriate for mitigating the risks.
- 3rd phase - implementing controls: the decided risk treatments are implemented.
- 4th phase - measuring program effectiveness: in this phase the implemented treatments are monitored to verify their effectiveness. This phase also covers the ongoing process of watching out for new, potential risks.

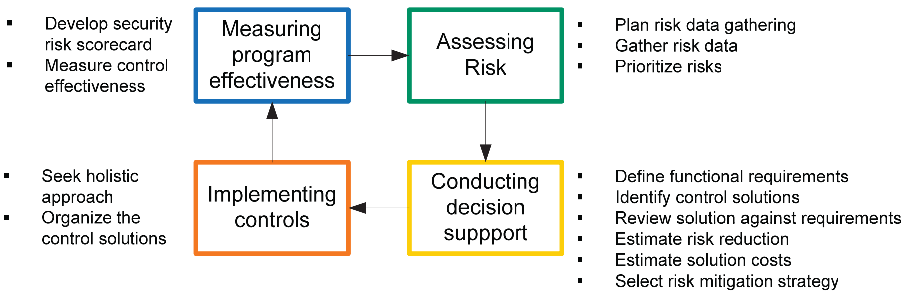

Develop security risk scorecard
Measure control effectiveness
Measuring program effectiveness
Assessing Risk
Plan risk data gathering
Gather risk data
Prioritize risks
Seek holistic approach
Organize the control solutions
Implementing controls
Conducting decision suppport
Define functional requirements
Identify control solutions
Review solution against requirements
Estimate risk reduction
Estimate solution costs
Select risk mitigation strategy

Figure 6 – Microsoft's security risk management process

3.2.6 Other Methods

We also include some security related methods and framework that are worth mentioning.

3.2.6.1 COBIT

ISACA (http://www.isaca.org) develops *COBIT* (Control Objectives in IT) that started as a guide on best practice IT management controls, intended for computer auditors. COBIT is a framework that focuses on assisting managers in the implementation and quality control of IT processes and systems in their organization, and not so much a risk analysis method according to our definition. To obtain the latest version of the COBIT (v4) documentation you must be a member of ISACA.

3.2.6.2 FIRM

The Information Security Forum (ISF) is an international association of more than 260 leading companies and public sector organizations (http://www.securityforum.org). ISF provides a set of methods and tools related to risk analysis and risk management. Among others they have a methodology for the monitoring and control of information risk at the enterprise level called *FIRM*. *SARA* is their risk analysis method for critical information systems, and *SPRINT* is a method for less critical systems and assessment of business risks. The methods are however only available through membership in ISF.

3.2.6.3 MEHARI

MEHARI (Méthode Harmonisée d'Analyse de Risques Informatiques) [119] is a French risk analysis method, designed by CLUSIF (https://www.clusif.asso.fr/en/clusif/present/), an organization consisting of security experts. MEHARI provides a method, tools and knowledge bases for use in controlling the risks of an organization. The method is compliant with ISO/IEC13335 (see Sect. 3.3.2) and its results can be used as input to ISO27001/BS7799-2 certification (see Sect. 3.3.3). MEHARI is not well known outside France and can only be purchased from CLUSIF.

3.2.7 Summary

This section summarizes the relations between CORAS and other security analysis methods.

Method	Compared to CORAS
OCTAVE	Similar process, similar vocabulary. Comes with more specific guidelines and predefined templates for documentation than CORAS. Limited use of graphical risk modeling.
CRAMM	Similar process, similar vocabulary. More focused on certification and the requirements needed for this purpose than CORAS. No graphical risk modeling.
FRAP	Similar process, similar vocabulary. Makes no particular use of graphical modeling.
Microsoft's Security Risk Management	Covers the entire risk management process, where the two first phases constitute a method according to our definition. The analysis process is similar to CORAS, but it does not use graphical modeling.
Other methods	COBIT is internationally well known, but it is more an audit framework than a security analysis method. FIRM, SARA, SPRINT and MEHARI are not widely used outside their organizations countries, and therefore of less interest. We have however no indications that any of these methods are fundamentally different in their risk analysis process than CORAS.

3.3 Standards Relevant to Security- and Risk Analysis

There exist several international standards relevant to security and risk analysis. In this section, we give a presentation of the ones relevant to our work, and how they relate to the conceptual foundation in CORAS. The section concludes with a short summary of each of the standards' relations to CORAS.

3.3.1 AS/NZS4360

The Australian New Zealand Standard for Risk Management (AS/NZS4360) [6, 7] defines a complete risk management process. It defines risk management as *the total process of identifying, controlling, and eliminating or minimizing uncertain events that may affect IT system resources*. The standard is accompanied with a handbook that gives a "generic guide for the establishment and implementation of a risk management process for information security risks" [54, 55]. The process has five main phases (Figure 7): *(1) establish context, (2) identify risks, (3) analyze risks, (4) evaluate risks and (5) treat risks*:

1. When establishing the context, the purpose is to decide what parts of the system that will receive attention, and why. This phase also aims to specify the stakeholders and vulnerabilities of the system.

Risk assessment is the collective term for identification, analysis and evaluation of risks. These three phases have the following purpose:

2. During risk identification, one should identify potential threats and threat scenarios that can constitute risks towards the system. A commonly used risk identification technique is some form for structured brainstorming.
3. The purpose of risk estimation is to estimate the risks' likelihoods and potential consequences. This activity is mostly centred on applying statistical methods or more qualitative judgments to establish the likelihood and consequences of the risks, and modeling notations like fault trees [66] and event trees [64] may be used.
4. Risk evaluation includes prioritization of the risks according to their severity and selecting the ones that cannot be tolerated which will be subject to treatment identification.
5. In the risk treatment phase the purpose is to identify the risks that need treatments, i.e. risks that are too serious to leave unattended.

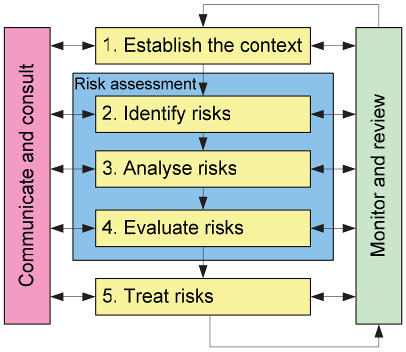


Figure 7 – Risk management process overview

The standard also defines two parallel activities *communicate and consult*, and *monitor and review*, which are related to the continuous risk management effort within an organization. CORAS takes much of its terminology from AS/NZS4360 and consequently the conceptual foundation is to a high degree in accordance with this standard.

3.3.2 ISO/IEC13335

The standard *ISO/IEC13335 Information technology* [76-79] provides help and guidance on how to manage security of information systems. The standard consists of five parts. *Part 1- Concepts and models for information and communications technology security management* [76] addresses security and security management in and organizational view, providing guidance on planning, implementation and operation of security measures. *Part 2- Information security risk management* has been revised by Part 1. *Part 3- Techniques for the management of IT Security* [77], *Part 4- Selection of safeguards* [78] and *Part 5- Management guidance on network security* [79] are more technical security standards. ISO/IEC13335 has in its latest revision been adapted to align with AS/NZS4360 and it is expected that several of its parts will be released under the new standard: Information Security Management (ISO/IEC 27000-family). Since ISO/IEC13335 has become aligned with AS/NZS4360, there is a substantial agreement between the conceptual foundation in CORAS and the terminology used in the ISO standard.

3.3.3 BS7799/ISO17799

BS7799 [25] was published in 1995 by the British Standards Institution, as a standard to guide the development and implementation of information security management system. It was a so-called *code of practice*, which means that it provides help and guidance on the selection and implementation of controls, but cannot be used in third party verification or certification of products. The demand for such a certification option lead to the development of *BS7799-2 (ISO27001) Information technology – Security techniques, Information security management systems – Requirements* [26, 75] which has requirements specified in the form "shall". In addition it integrates the process-based approach of *ISO9001 Quality management systems – Requirements* [71] and *ISO14001 Environmental management systems – Requirements with guidance for use* [72]. The original code of practice is now known as *BS7799-1 (ISO17799)* [25, 82]. Even though the standard has a broad international accept, it does not define concepts in the level of details like the CORAS conceptual model. It is more like a high-level guideline for security risk management.

3.3.4 ISO15408 (The Common Criteria)

The Common Criteria (ISO15408) [33, 73] gives a set of security requirements a product or system must conform to in order to match a specific level of security. This makes it possible for a vendor to explicitly state the level of security he/she require from a system or product. The requirements are divided into functional requirements and assurance requirements. The functional requirements are security aspects that a product must conform to. Assurance requirements are used to evaluate whether the security aspects of a product or system work and whether these are implemented correctly. There are seven assurance levels, each with a stricter requirement to how the product or system should be evaluated:

1. EAL1: the product must be functionally tested
2. EAL2: the product must be structurally tested
3. EAL3: the product must be methodically tested and checked
4. EAL4: the product must be methodically designed, tested and reviewed
5. EAL5: the product must be semiformally designed and tested
6. EAL6: the product must have semiformally verified design and be semiformally tested
7. EAL7: the product must have formally verified design and be formally tested

Although the Common Criteria does not come with an implementation method, various attempts have been made to specify a method that supports integration of its security aspects into the software engineering process [157]. The common criteria evaluation is conducted by authorized entities as an assessment of the product or system against the defined criteria. This is more like a security audit than a complete security analysis process, and focuses on specification and testing of the product or system. Except for "asset" the standard does not use any of the risk related concepts in CORAS.

3.3.5 IEC61508

IEC61508 *Functional safety of electrical/electronic/programmable electronic safety-related systems* [69] is the international standard for all safety lifecycle activities for systems comprised of electrical and/or electronic and/or programmable electronic components (electrical/electronic/programmable electronic systems (E/E/PESs). The safety lifecycle of a system covers all safety related activities made in phases from concept, through design, implementation, operation, maintenance until decommissioning. Typical examples of E/E/PESs are control and supervisory systems used in domains like industry, health and transportation. The key features of IEC61508 can be summarized like this [15]:

1. It facilitates development of sector specific standards for safety related E/E/PESs that builds on the same underlying principles, terminology etc.
2. It helps specifying safety requirements for safety related E/E/PESs.
3. It introduces safety integrity levels (SIL) that are used to specify the level of safety required by a function in a safety related E/E/PESs.
4. It has a risk-based approach to determine SIL requirements.
5. It sets numerical target failure measures for safety related E/E/PESs that are linked to the SIL's.
6. It sets a lower limit on the target failure measures that can be claimed for a single safety related E/E/PES.

Compared to the concepts in CORAS, IEC61508 concepts are much more system oriented with terms like safety function, module, safe state etc. The standard does not focus on traditional security oriented aspects like risks posed by someone/something with malicious intents like hackers, intruders or malicious code. The lack of such concepts makes it less comparable to a conceptual model for security risk terminology like in CORAS.

3.3.6 ISO21827

The standard *Information technology - Systems Security Engineering - Capability Maturity Model (ISO21827)* [74] defines a complete development process for security engineering. Its concepts and definition are mainly taken from ISOIEC13335, except for "threat" which is defined as *Capabilities, intentions and attack methods of adversaries, or any circumstance or event, whether originating externally or internally, that has the potential to cause harm to information or a program or system or cause those to harm others*. Nevertheless, this interpretation of threat does not conflict with the more general definition used in the CORAS method: *a potential cause of an unwanted incident*. Due to its near coupling with ISO/IEC13335, which is already used in CORAS' method and underlying foundation, the terminology used in ISO21827 is considered to be sufficiently covered.

3.3.7 Other Standards and Guidelines

The standards *Information technology – Security techniques – Guidelines for the use and management of Trusted Third Party services (ISO/IEC14516)* [80] and *Information technology – Security techniques – IT Network security (ISO/IEC18028)* [83] are standards targeting technical security solutions in a detailed level with no focus on security analysis as a process. Their objectives are to extend the security management guidelines provided in ISO/IEC13335 and ISO/IEC17799.

Information technology – Security techniques – Information security incident management (*ISO/IEC TR 18044)* [85] provides advice and guidance on information security incident management for information security managers and for information system managers. The standard can be seen as a supplement to the security risk management process and not a stand-alone security analysis standard.

Information technology – Software life cycle processes – Risk management (ISO/IEC16085) [63] defines a process for the management of risk during a complete software lifecycle process with no particular focus on security. This includes risks related to both buying and supplying software, as well as developing, operating and maintaining software. It may be used to guide standards like *Software life cycle processes (ISO12207*) (a high-level standard addressing all processes of the software life cycle) or *Systems Engineering - System Life Cycle Processes* (*ISO15288)* (a framework for describing the life cycle of systems created by humans).

There are also quality management standards that organizations may be audited against to prove their compliance with specific quality requirements. These includes among others: the *ISO9000 quality management family* [71], *IC9700* and *IC9200 business certifications* (www.icharter.org), and *ISO20000* for IT service management [84]. None of these is targeting risk analysis or security analysis.

The U.S. National Institute of Standards and Technology publishes standards and best-practice guidelines for a wide range of IT security related topics. The *NIST SP800-30 Risk Management Guide for Information Technology Systems* [123] provides *a foundation for the development of an effective risk management program, containing both the definitions and the practical guidance necessary for assessing and mitigating risks identified within IT systems*. The publication is therefore more like a guideline than a standard and in a comparison with OCTAVE the authors claim [162] "following the OCTAVE guidance will meet the spirit and intent of the NIST guidance for conducting the risk assessment as part of a total risk management program described in NIST SP 800-30"

3.3.8 Summary

This section summarizes the state-of-the-art with respect to standards for security and risk analysis that are relevant to the conceptual model for security risk analysis defined in CORAS.

Standard	Relation to CORAS
AS/NZS4360	CORAS employs many of the concepts in this standard in its underlying foundation.
ISO/IEC13335	CORAS employs many of the concepts in this standard in its underlying foundation.
BS7799 and related standards	Has a stronger relation to the CORAS method than the conceptual foundation, since it deals with risk management without detailed risk analysis concept definitions.
Common Criteria	The standard does not provide as detailed concept definitions like the one used in CORAS' conceptual model.
IEC61508	Employs much more system oriented concepts than the security focused conceptual model in CORAS.
ISO21827	Sufficiently covered by CORAS, since it is based on ISO/IEC13335.
Other standards	The other standards and guidelines are either specialized towards particular security technologies, too high-level or lacks focus on security analysis.

3.4 Risk- and System Analysis Techniques

Traditionally risk analysis techniques document their findings in tables only, but our claim is that many of them may benefit from using graphical documentation techniques as well. In this section, we present risk analysis techniques that have in common that they may be used in structured brainstorming, and we evaluate whether they are appropriate for graphical risk documentation techniques. The evaluation is summarized in Sect. 3.4.13.

3.4.1 SWOT

A *SWOT* analysis [62] is used to evaluate the Strengths, Weaknesses, Opportunities, and Threats associated with a project or business activity in a top-down manner. Used in project or business planning the objective is to define a goal, strategy or actions, and find means to achieve it (e.g. increase income, reduce development time). In security analysis it is used to get an initial overview of the risk picture and helps scoping the analysis to focus where it is most needed. In a security analysis the objective of a SWOT will be to protect the assets within the target of analysis and then find:

1. Strengths: attributes of the target that are helpful in protecting assets.
2. Weaknesses: attributes of the target that are harmful to achieve sufficient protection of the assets.
3. Opportunities: external conditions that are helpful in protecting assets.
4. Threats: external conditions that are harmful to achieve sufficient protection of the assets.

A SWOT analysis takes form as a brainstorming session involving a cross-functional team consisting of people with different background and view of the target system. The information they come up with may be used as input to a more detailed security analysis. The findings are typically documented in various forms of table formats, adjusted to the particular need of the analysis, but often structured according to the four SWOT aspects. A security risk focused SWOT analysis may benefit considerably from using an easily understandable graphical approach to model its findings. This will for instance make it possible to illustrate how a threat in one area may exploit a weakness in another, or how an opportunity may become a weakness if seen from a different point of view.

3.4.2 FMEA/FMECA

FMEA/ FMECA (Failure Mode Effect Analysis / Failure Mode Effect and Criticality Analysis) [17] is a method that assess potential failures of individual components within a system. The method is usually conducted in two steps, first the failure modes and their effects are identified (FMEA). Then the failure modes are ranked according to their criticality and their probability (FMECA). The basis of the FMEA/FMECA is functional description of the system, where each component is analyzed to identify all possible or failure modes and classify them according to their criticality. The FMEA/FMECA is a "bottom-up" approach, especially suitable for detecting a system's possible failure modes, and determining their consequences. The failure identification is normally organized as a brainstorming, structured by the system's functional descriptions. The findings are documented in a table where each separate module's potential failure modes are investigated with respect to failure detection method, failure effect and how critical it may be. It is not obvious how to document relations between failure modes in different modules, neither how the effects may be common for several modules. This is a common problem of tables, where relations between different rows are difficult to show. In this regard, a graphical language may be useful to document relations between the findings instead of, or in addition to the conventional FMEA/FMECA tables.

3.4.3 HazOp

HazOp (Hazard and Operability) analysis [70] is a well known risk identification technique used in all forms for risk analyses. A HazOp is a structured brainstorming with the aim of finding ways system behavior may deviate from design intention, and whether threes deviation can lead to unwanted incidents (hazards). The participant all must have through knowledge of one or more aspects of the system analyzed. The input to the analysis is system documentation of any kind, and in addition the analysis leader uses specialized guidewords to ensure that all aspects are covered. The guidewords are used in questions like "what if the service delivers too much data?", "what if the service delivers too little data?", "too slow response or too early?" and so on. This is meant to mitigate the weakness that the information gathered during a HazOp is restricted to the already existing knowledge within the group. The idea is that the guidewords can make people think of aspects they have not been thinking of before. A similar technique that is commonly used within safety analysis is called *HazId* (Hazard Identification). This is basically a simplified HazOp that uses checklists rather than guidewords, and it is often used early in the analysis process or for smaller risk analyses. HazOp can be tailored to fit any domain and system, in for instance [131] the method is especially targeting software HazOp. These kinds of methods represent particular suitable situations for using graphical security risk modeling languages since a common understanding and communication between the participants are crucial to the quality of the findings. Similar to FMEA/FMECA tables, also HazOp tables are unsatisfactory for showing relationships between the findings (the different rows in the table). There is clearly a need for both documentation methods, where one keeps detailed information about each risk in tables while the relationships between the risks are documented graphically.

3.4.4 Fault Tree

The *fault tree* notation is used in fault tree analysis (FTA) [66] to describe the causes of an event. Fault trees are well known and widely used within risk analysis, and become more commonly in security analysis, typically of systems that may have consequences for safety. The notation provides an excellent way of structuring the order of events, and is particular useful if there exist numerical statistical data to use in calculations. Fault trees may among others be used to model the findings of HazOp analyses [152]. The top node represents an unwanted incident, or failure, and the different events that can lead to the top event are modeled as intermediate nodes or leaf nodes (Figure 8, taken from [133]). The probability of the top node is calculated based on the probability of the leaf nodes and the logical gates "and" and "or".

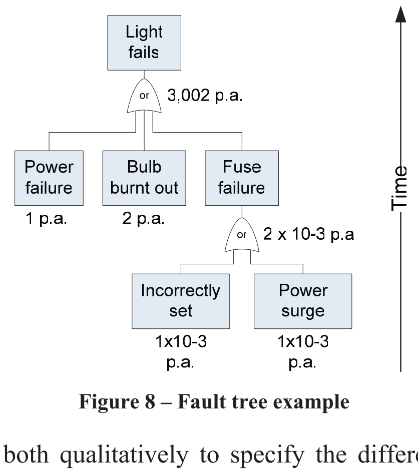


Figure 8 – Fault tree example

Fault trees can be used both qualitatively to specify the different paths that lead to the unwanted incident, as well as quantitatively to estimate the likelihood of the top node incident [2]. The leaf nodes in a fault tree must be independent of each others, otherwise one has to apply special methods for computing likelihood values [107]. An incident model that takes the fault tree notation a step further into a more complex structure is the MORT

(Management Oversight and Risk Tree) [87] which is more common within safety risk modeling. There exist specialized methods for quantitative analysis of fault trees (e.g. [43]) and also methods that takes into account uncertainty regarding the likelihood estimates (e.g. [92]), but these topics are beyond the scope of this thesis.

The modeling notation used in FTA is quite easy to understand and particularly useful for systems consisting of hardware/software modules. Whenever the system also includes people's behavior, the notation becomes too "rigid". It is not feasible to set numerical fault rates for humans in the same manner as for e.g. hardware components. FTA does cover the outcome of the unwanted incident and provides therefore only one side of the risk picture. The underlying idea of illustrating the chain of events that may lead to a fault or unwanted incident is however very useful, and can be used as a basis for a modeling language that is less strict than the fault tree. The

3.4.5 Event Tree

Event trees [64] use a tree notation to represent the outcome (or consequences) from an event and the probability of the various consequences (Figure 9, example from [133]). In the same manner as a fault tree, also the event tree is both qualitative (shows the outcomes from and event) and quantitative (estimates the likelihood of each outcome). When constructing an event tree it is normal to use a binary split from the initial event, towards the final consequences (success/failure). Event trees can be used as an extension of fault trees to create a "cause consequence" diagram (Figure 10). The event tree lets the modeler specify the "barriers", or the mechanisms that shall prevent the consequences of an unwanted incident from escalating. It also describes what the outcome will be if the barriers fail to work.

The event tree lets you specify every detail about the expected outcome from an unwanted incident. It also includes the barriers that should be in place to minimize the consequences. Similar to the fault tree, also event trees provides half the risk picture, excluding the chain of events that may lead to the incident. As one may see from the small example in Figure 9, the tree will grow rapidly when the number of barriers is high, and it does not allow for showing how a failure in a barrier may initiate a new unwanted incident. Nevertheless, the underlying idea of event trees is very valuable, but there is room for an improved and possible more flexible notation.

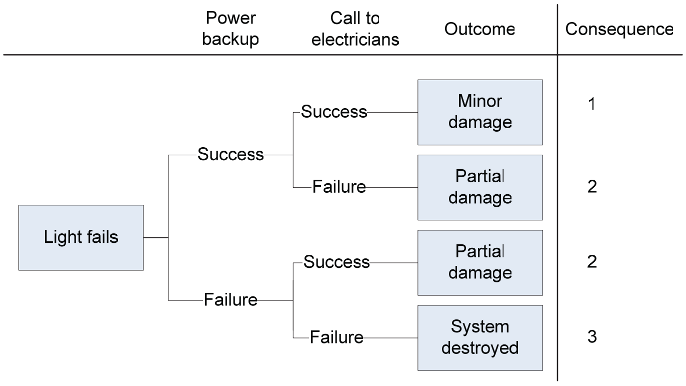


Figure 9 – Event tree example

3.4.6 Cause-consequence Diagram

The *cause-consequence diagram* [122] combines the features of both fault tree and event tree. When constructing a cause-consequence diagram one starts with an unwanted incident and develops backwards to find its causes (fault tree) and forwards to find its consequences (event tree). Figure 10 shows and example of a cause-consequence diagram which is a combination of the two previous diagrams (Figure 8 and Figure 9).

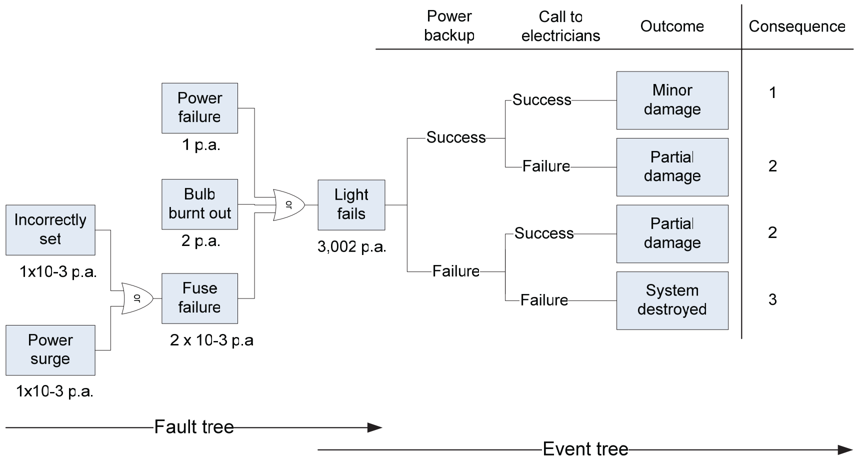


Figure 10 – "Cause consequence" diagram

The cause-consequence diagram provides the complete risk picture, which both FTA and ETA lacks. It illustrates the chain of events from the very beginning with the initiators of unwanted incidents, to their final consequences towards assets. Since it builds on FTA and ETA it also inherits their weaknesses (as discussed in the two previous sections), but it captures much of the main idea behind what we consider as the ideal way of presenting a risk picture. The notation should be optimized with respect to presentation, but it should also be able to model the risk picture in a high level manner, without demanding all details about the chain of events (order of events, probabilities, logical gates etc.). For instance, detailing each path through the diagram may be left for subsequent analyses.

3.4.7 Attack Tree

Attack trees [138, 139] is a modeling notation that aims to provide a formal and methodical way of describing the security of a system based on the attacks it may be exposed to. The notation uses a tree structure similar to FTA, with the attack goal as the top node and different ways of achieving that goal as leaf nodes (Figure 11, example from [138]).

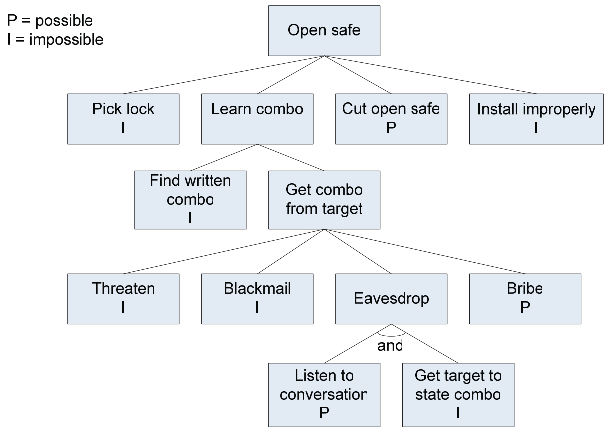


Figure 11 – Attack tree example

There exist an extension of attack trees called *defense trees* [16] which in addition to representing attack strategies performed by an attacker, also represents the possibly countermeasures that may be implemented in the target system to mitigate the attacks.

Since the notation is based on fault trees we get the same difficulties with respect to expressing logical gates, assigning likelihood/probability values to threat scenarios and computing precise likelihoods. Attack trees do however allow for less precise likelihood values (e.g. possible/impossible like in Figure 11) which makes it easier to use for high level analysis. Its focus is more on human behavior than system behavior since it represents different ways of attacking a system. In many cases it may for instance be valuable to represent both how the system reacts to a security breach caused by a human, and a non-human source.

3.4.8 OCTAVE Threat Tree

The OCTAVE method (Sect. 3.2.2) for security analysis has its own tree notation which has much in common with event trees, but also fault trees. The figure below is an *OCTAVE threat tree* from [3] showing the threat posed by a human actors using network access to harm "PIDS" (Patient Information Data System). The diagram illustrates the source of an incident, the method and the motive behind the incident which can be compared to fault trees. At the same time, it illustrates the outcome of the incident that is more like an event tree. The OCTAVE threat tree can be seen as taking the tree notation one step closer to security risk analysis, in particular information security. The use of deliberate and accidental threats (in OCTAVE called motive) is for instance in accordance with ISO/IEC13335 *Information security* (Sect. 3.3.2). Our concern is whether it is feasible to mix two well established techniques like FTA and ETA in this way. This makes it more difficult to exploit the benefits from computerized FTA or ETA tools, and also conflicts with many analysis methods that recommend using these analysis techniques. The intention of adapting the tree notations to more security focused analysis by integrating security related concepts is however good.

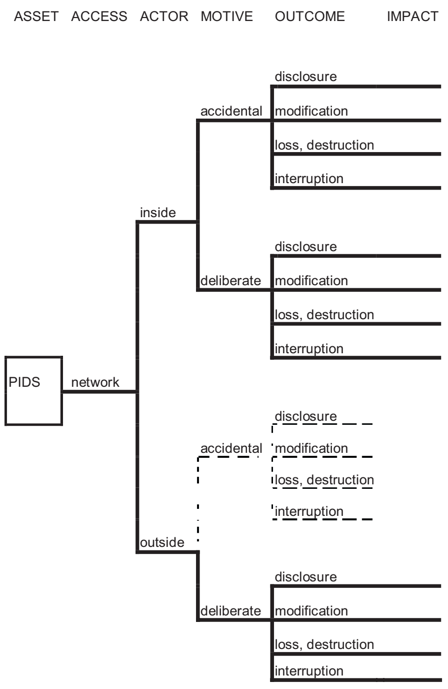


Figure 12 – OCTAVE threat tree

3.4.9 Bayesian Network

A *Bayesian network* [98, 107, 108, 129, 136, 150] is like a directed, acyclic graph. The intermediate nodes represent causes or contributing factors to the top node, which in Figure 13 is a "system failure" (taken from [129]). A Bayesian network is both a graphical and a probabilistic model that may be used to for instance predict the number of faults in a software component [41]. In Figure 13 the causes that contribute strongest to the event (A1-A3) are placed directly before the event. The causes are grouped into three categories: organizational factors, human factors and technical factors. When a Bayesian network is analyzed quantitatively, each node holds a table with a probability distribution reflecting its parent nodes. For any manipulation of the probabilities of the nodes, the effects both forwards (towards child nodes and the top node) and backwards (towards parent nodes) can be computed [42]. A Bayesian network can be utilized both quantitatively and qualitatively. If the Bayesian network is analyzed qualitatively, it provides relations between causes and effects. When analyzed quantitatively, one uses its powerful mathematical model for computing probabilities, which is not only based on the probabilities for the leaf nodes like in fault trees, but also on intermediate nodes.

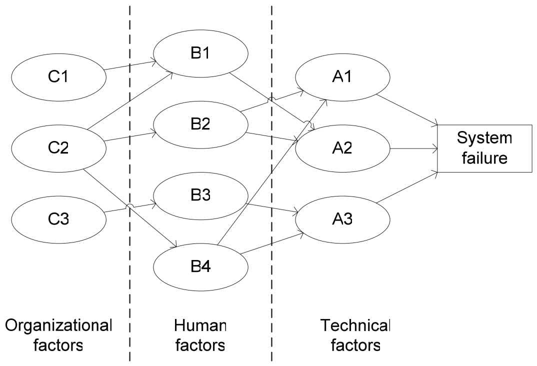


Figure 13 – Example of a BN from [129]

3.4.10 Markov Analysis

Markov analysis [61, 68, 94, 107] is a stochastic mathematical analysis method that looks at sequences of events and analyzes the tendency of which event that will be followed by another. Markov analysis may be used to analyze the reliability of systems that have a large degree of component dependencies. In contrast to FTA, Markov analysis does not assume complete component independence. It is also well suited to analyze systems that may partially fail or experience degraded states. A Markov analysis considers the system as a number of states, and transmissions between these states. The states are modeled graphically and statistical calculations are performed to determine the probability of each state transmission. Markov analysis is among others promoted by ISO/IEC61508. Markov models are more suitable for showing the operation modes of a system where one may transit forth and back between states, than a chain of events of a security attack which is more likely to be a one way chain. Nevertheless, describing the operation modes of a system also includes describing the different barriers that should prevent an attack or reduce the consequences of an attack and for this purpose Markov analysis may be a useful tool. Using Markov analysis requires a well-specified system and may not be as suitable for high-level analyses.

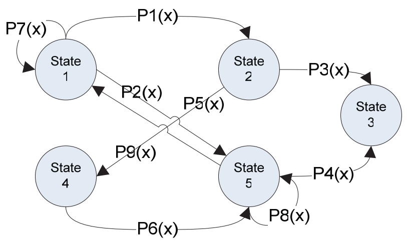


Figure 14 – Markov model

3.4.11 Block Diagram

Block diagrams [67, 129] are often used in reliability assessments and shows how the components of a system are parallel or serial and thereby identifies possible weak points. In the figure below, components 1-3 are parallel, while component 4 is not and therefore may be the weakest point in this system. While this notation is a suitable way of representing the components of the target system, it is not directly applicable to model security risks. The idea of showing alternative paths in a system can however be used in security risk modeling to show where a threat may work its way through the target of analysis. The UML profile is already using this modeling principle, and it will be an important feature in a new security risk modeling language.

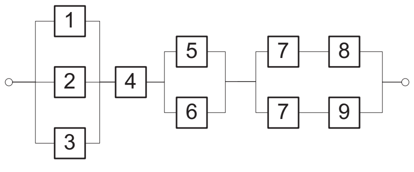


Figure 15 – Block diagram example

3.4.12 Other Techniques

There exist also risk analysis methods that assess UML models on a very detailed level. In [53] the authors propose a severity assessment methodology which integrates Functional Failure Analysis (FFA) [1], FMEA and FTA to analyze the severity of failures of the architectural elements. In [34] the authors describe a risk analysis method of performance failures based on annotated UML models and [50] describes a method for analyzing architecture level UML models to identify the parts of the system which may give rise to risks. The method applies complexity analysis of components and connectors, expert judgments for severity analysis and Markov models to estimate the risk factor of the different system scenarios. A method for estimating risk in the early phases of design is proposed in [5], where the analysis is conducted at the requirements-level based on UML design specifications. In [149] the authors propose a framework that models cluster computing systems' availability based on UML design notations, and evaluates system availability by transforming the UML availability modeling into corresponding analytical models (e.g. Markov models). Common for these techniques is that they deal with risk analysis on a very technical and detailed system level.

Preliminary Hazard Analysis (PHA) [93] is typically used early in the design phase of a plant where the process is new and one has little knowledge of its risks. The outcome is a set of hazards (what we refer to as unwanted incidents), their causes and major effects, in addition to a list of corrective/preventive measures. As the name indicates, the analysis is similar to a HazOp, except for being used in an earlier phase.

MORT (Management Oversight Risk Tree) [87] is a technique that can be used both to investigate the causes of incidents and to audit risk management systems. The technique uses a tree structure similar to fault trees (Sect. 3.4.4). A related approach is SMORT (Safety Management and Organization Review Technique) [96] which builds on MORT and targets internal investigations of incidents and near incidents.

"What if?-analysis" is one of the oldest risk identification methods, based on asking a series of questions often starting with "What if…?" or similar. The method may use checklists as a basis and is used in assessing system design. The analysis method is a typical brainstorming method, where consequences and recommendations are put forward by the people participating. This is an example of a "what if"-analysis result from an analysis of an reactor [93]:

What if?	**Consequence / hazard**	**Recommendations**
Cooling water pump breaks down	Runaway reaction explosion/fire	Stand-by pump / alarm systems

3.4.13 Summary

This section summarizes our judgments of existing risk analysis techniques and how they are relevant when developing a new, graphical risk modeling language inspired by the UML profile.

Technique	**Relevant to a new risk modeling language?**
SWOT	Yes, since it relies on a brainstorming technique which involves several participants it is a suitable situation to use graphical modeling language to facilitate communication and understanding.
FMEA/FMECA	Yes
HazOp	Yes, since it relies on a brainstorming technique which involves several participants it is a suitable situation to use graphical modeling language to facilitate communication and understanding.
Fault tree	Yes, there is no method for modeling logical and/or-gates in the UML profile which means that the reader cannot know which one to use when reading UML profile threat diagrams. The CORAS UML diagrams can probably express the chain of events that a fault tree represents, but not the logical order in terms of and/or dependencies, and not facilitate precise calculations
Event tree	Yes, there is no method for modeling or-gates in the UML profile, but some of the intention of an event tree can be modeled, although it will not look like a conventional event tree. Since event tree is a commonly used notation, the new language should include its idea and objective.
Cause-consequence diagram	Yes, since this is based on both fault tree and event tree, it is natural to include the intention behind cause-consequence diagrams into a new language.
Attack tree	Yes, since the notation is quite similar to fault trees it should be considered and sufficiently covered when developing the new language.
OCTAVE threat tree	Yes, both because it has strong similarities with event trees and because OCTAVE is an international well known security analysis method that a new language should aim to comply with if possible.
Bayesian network	Yes, the event modeling in Bayesian networks is similar to the threat scenario modeling in the UML profile, but the statistical model is unique to the Bayesian network. While it is not within our ambitions to develop this kind of a probability model, it would undoubtedly be very interesting and possibly future work.
Markov analysis	Partly, but only to simulate the different states since the analysis relies on it own, special modeling notation.
Block diagram	Yes, even though block diagrams are more suitable to model components of the target system and not the security risks itself, the idea of showing different paths through a system can be used in a new security risk modeling language.
UML based analysis techniques	These techniques already use UML modeling in analysis, and they also target a much more detailed level of system specification than our approach will address.
Preliminary Hazard Analysis (PHA)	Yes, with its similarities to HazOp we can conclude that also for this technique graphical modeling can be useful
MORT/SMORT	Partly, because MORT is based on using fault trees, which means that it already has a customized modeling notation. However, if the graphical approach is able to express all aspects of the MORT/SMORT analysis and shows superior with respect to understandability or usability, there is no reason for not substituting the fault tree with our approach.
What if? Analysis	Yes, with its basis in a kind of structured brainstorming we are certain that a graphical modeling approach may be useful.

3.5 Risk Documentation Techniques

There are many techniques that can be used to support risk identification, estimation and documentation, but the fact that CORAS already had a graphical modeling language made it a suitable place to start. We therefore give a rather detailed presentation of the UML profile, and then we describe other graphical notations that may be used in risk- and security analysis, and system development.

3.5.1 The UML Profile

The full name for the UML profile is the *UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms* [124], and it is standardized by the OMG (Object Management Group). The language is based on the use case notation from the *Unified Modeling Language (UML)* [134]. It uses the official UML meta model, but with added richness in terms of domain specific symbols and concepts related to security risk modeling. In the following we describe the UML profile by means of examples of modeling taken from the profile [124]. The presentation will guide you through a complete security risk modeling process. The diagrams have not been given special names in the standard, but for simplicity we provide each diagram with a name in the figure caption. Stereotyping is a technique used in UML to add information to a model element by giving it special names or stereotyping labels.

The values and scales that will be used during the security analysis are defined in the value definition diagram (Figure 16). The stereotype <<ValueDefinition>> is used for defining each value type that is used. In this example all values are enumerations, i.e., values on an ordinal scale, except for "RiskReductionRef" which defines a mapping. Alternatively, assets could have been defined in terms of monetary values, frequency as probabilities and so on.

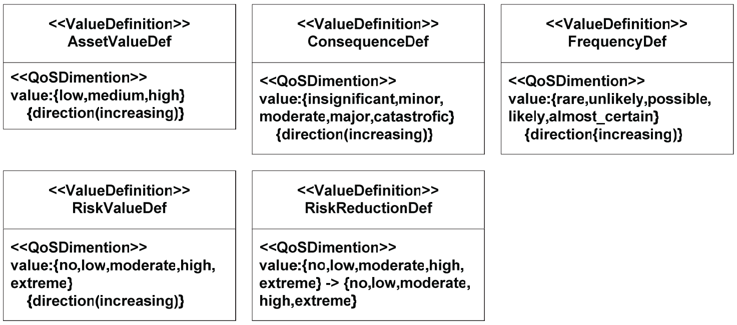

<<ValueDefinition>> AssetValueDef
<<QoSDimention>> value:{low,medium,high} {direction(increasing)}
<<ValueDefinition>> ConsequenceDef
<<QoSDimention>> value:{insignificant,minor, moderate,major,catastrofic} {direction(increasing)}
<<ValueDefinition>> FrequencyDef
<<QoSDimention>> value:{rare,unlikely,possible, likely,almost_certain} {direction{increasing)}
<<ValueDefinition>> RiskValueDef
<<QoSDimention>> value:{no,low,moderate,high, extreme} {direction(increasing)}
<<ValueDefinition>> RiskReductionDef
<<QoSDimention>> value:{no,low,moderate,high, extreme} -> {no,low,moderate, high,extreme}

Figure 16 – Value definitions diagram in the UML profile

Figure 17 shows the specification of an asset, which is an important part of the CORAS method. The service is an entity that is associated with some quality characterizations. The asset is defined as the quality level of the service, related to the offered service quality. The asset is owned by the stakeholder "Service provider", and its value is assigned by instantiating the value definition for asset values (Figure 17). The diagram also shows that the asset has one vulnerability, which is "extensive computation".

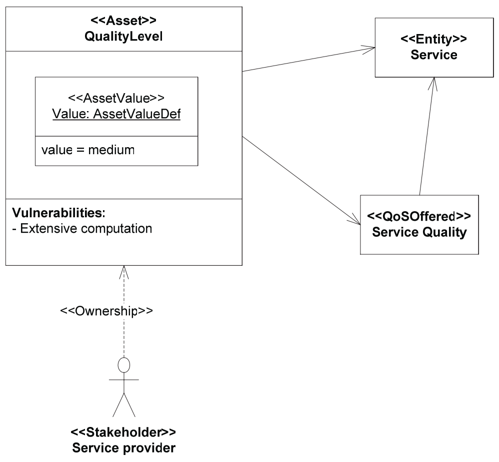


Figure 17 – Asset specification diagram in the UML profile

In Figure 18, the modeling of a threat is exemplified in a UML profile threat diagram. The threat "Malicious person" has the scenario (i.e. behavior) "Flooding". This threat scenario is related to the asset "QualityLevel". In this diagram, asset is shown using the UML actor stereotype only, while Figure 17 provided detailed information about the same asset.

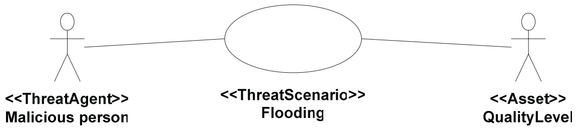


Figure 18 – Threat diagram in the UML profile

Figure 19 illustrates how unwanted incidents are modeled with the UML profile. The unwanted incident "Denial-of-Service" may harm the asset "QualityLevel", and includes the threat scenario from the threat diagram above. A scenario may lead to another scenario, and this is shown by use of the stereotype <<Initiate>>. In this case, "Denial-of-Service" initiates the unwanted incident "Loss of customer" which may affect the asset "Customers".

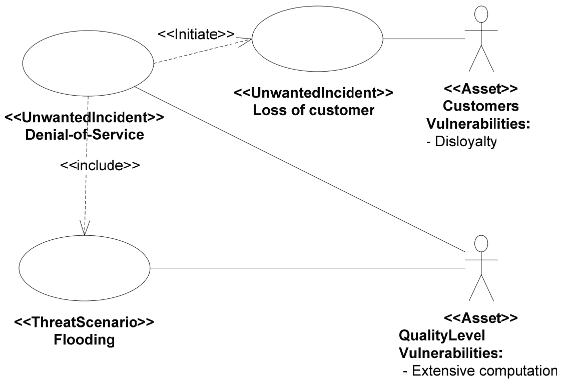


Figure 19 – Unwanted incident diagram in the UML profile

A risk is an assignment of consequence, frequency and risk value to an unwanted incident. Figure 20 illustrates how this is modeled. The values are instances of the corresponding value definitions (Figure 16). The risk of "Denial-of-service" is assigned to the unwanted incident "Denial-of-Service" using the stereotype <<RiskEvaluation>>. The diagram also shows that the risk is related to the asset "QualityLevel".

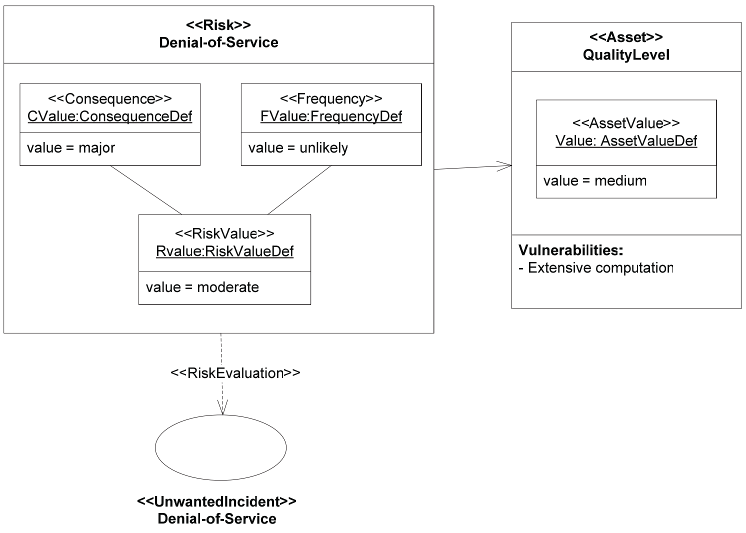


Figure 20 – Risk diagram in the UML profile

Similar risks may be grouped into risk themes. Figure 21 shows how the stereotype <<RiskTheme>> is used to define risk themes of instances of risks. This allows a risk to be a member of several risk themes. In this example, the risks "Denial-of-service" and "Loss of customer" are grouped to form the risk theme "DoSRelated". As seen in the example, a risk theme is also assigned an overall risk value.

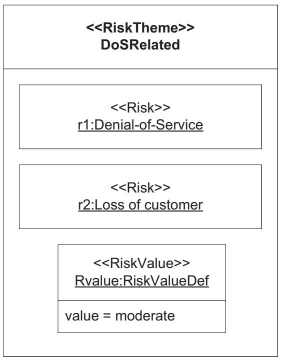


Figure 21 – Risk theme diagram in the UML profile

Figure 22 models "Authentication" as a treatment for the unwanted incident "Denial-of-Service". The stereotype <<Transfer>> (one of the predefined treatment options in AS/NZS4360) denotes that this treatment involves transferring the responsibility for the risk to the authentication mechanism solution.

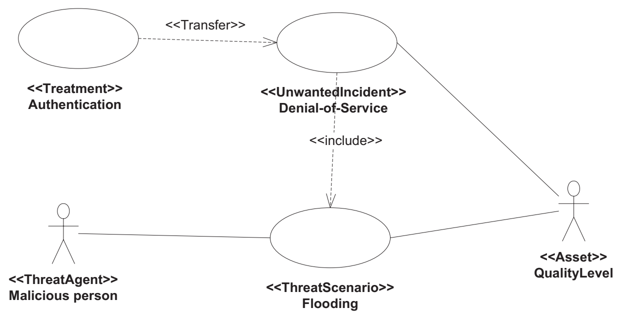


Figure 22 – Treatment diagram in the UML profile

Figure 23 shows an example of how a treatment effect is modeled. The treatment effect "DoSTransfer" is bound to the treatment "Authentication" by the use of the stereotype <<TreatmentEvaluation>>. The figure also shows that "DoSTransfer" relates to the risk "Denial-of-Service". The risk reduction, i.e. the value of the treatment effect, is a mapping from *moderate* to *low*, which means that implementing the treatment will reduce the risk value of "Denial-of-Service" from moderate to low.

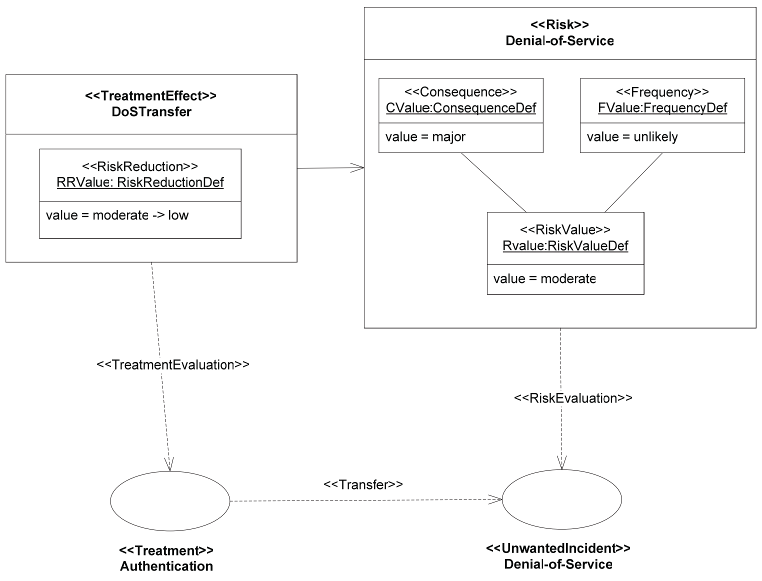


Figure 23 – Treatment effect diagram in the UML profile

The UML profile is as far as we know the only modeling notation that supports the entire security analysis process step-by-step. However, there exist related notations that can be used to model particular parts of the documentation in a security analysis, like reliability aspects of the system analyzed (e.g. "block diagram" in Sect. 3.4.11) or potential ways of attacking a system (e.g. "attack tree" in Sect. 3.4.7).

Comparing CORAS to the other techniques

The threat diagram in the UML profile may resemblance a fault tree, but may have more than one top node and lacks the notion of logical gates. The threat scenarios can be seen as leaf nodes/intermediate nodes are not given probabilities, something which makes fault tree computation impossible. The CORAS UML diagrams can express the chain of events that a fault tree represents, but not the logical order in terms of and/or dependencies. Neither can the CORAS UML diagrams provide a model for precise likelihood calculations.

The concept of barriers that one finds in event trees is underspecified in the CORAS UML diagrams, which only models the relation from an unwanted incident to each of the assets it potentially may harm. The success or failure of the barriers and the different types of consequences are not modeled explicitly. In addition, the consequence towards each asset must be modeled in separate CORAS UML diagrams

The UML profile provides insufficient support for the end-to-end view cause-consequence diagrams give. The complete chain of events are not modeled in one diagram, but must be specified using multiple diagram types.

The UML profile can be used in the same manner as an attack tree, with a threat assigned to the initial threat scenario as the leaf node, more threat scenarios as the intermediate nodes and the unwanted incident as the root node. The UML profile may on the other hand extend the attack tree notation by illustrating which assets that are affected by the unwanted incident (root node), and also the threats initiating the unwanted incidents.

The concepts modeled in OCTAVE threat trees are in many ways the similar to the concepts modeled in UML profile diagrams but they have different names. The concept "asset" is similar to asset in the CORAS method, while "access via network" would be considered a vulnerability. Both "inside actor" and "outside actor" would be called threats in the CORAS method, the "outcome" would be an unwanted incidents and their "impact" would be consequences in terms of damage to the asset. Also the information modelled is similar to the one gathered in the CORAS method, but the UML profile would normally not be used to model all possible outcomes of an event. This is however possible, but maybe not feasible when the events are numerous.

If the Bayesian network is analyzed qualitatively, it provides relations between causes and effects similar to the UML profile. The difference lies however in the Bayesian network's powerful mathematical model for computing probabilities, which is not only based on the probabilities for the leaf nodes like in fault trees, but also on intermediate nodes

The strength of the Markov model is the complex, underlying statistical model used for calculating probabilities for the different states of a system. The UML profile does not include a similar feature, and can at most be used to illustrate the different states of a system.

3.5.2 Misuse Case

The *misuse case* notation [145-147] is related to the UML use case notation (the example in Figure 24 is taken from [145]). As opposed to a use case which expresses allowed functionality in a system, a misuse case expresses the opposite, i.e. the functions that the system should **not** allow. A misuse case can be defined as "a completed sequence of actions which results in loss for the organization or some specific stakeholder" [145]. Among others, misuse cases have been used in design of secure systems architectures [126]. A similar notation to use cases is *abuse cases* [115, 117] which by Sindre and Opdahl is considered to be complementary to misuse cases. Abuse cases target security requirements with respect to design and testing, whereas misuse cases are used to elicit security requirements in relation to other system requirements.

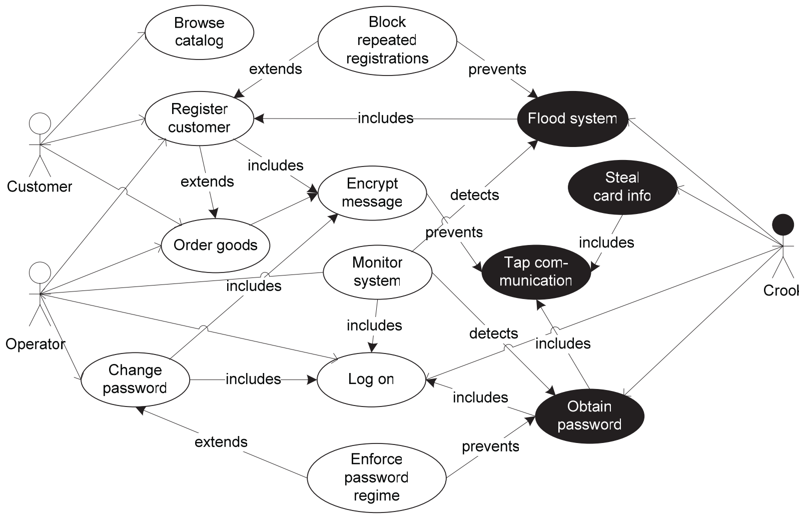


Figure 24 – Misuse case example

Misuse cases can be a useful tool in security analysis to direct focus towards functions in a system that may be exploited. The UML profile is inspired by the misuse case notation, but is richer and has a stronger relation to traditional risk analysis techniques. Misuse cases can therefore be expressed with the UML profile.

3.5.3 UML Variants for Security

There are a number of security oriented extensions and applications of UML, and in this section we introduce some of them. These and related approaches have however all been designed to capture and analyze security properties and security aspects at a more detailed level than what is required in the type of modeling aim for.

UMLsec [89-91] is an extension of UML (a UML profile) that provide means for specifying security requirements. The underlying basis is an "abstract state machine model" that formalizes UML elements (except for use cases) and extends stereotypes. The purpose is to be able to formally verify software specifications, which may reduce the number of security risks. A similar approach to UMLsec is described in [39], focusing on extending properties of essential UML elements (including use cases, actors, classes and methods) in order to apply security models directly (exemplified with their "mandatory access control" model).

Another language is *SecureUML* [11, 12, 110] which aims to extend UML with a meta model for role based access control (RBAC) [44] for use in model driven security engineering. Their "Model Driven Security" approach is based on first specifying systems models and their security requirements and then use tools to generate the system architecture from these specifications. The approach combines system modeling and system security in a detailed level with particular focus on RBAC. RBAC is also targeted in [130] where the authors model the concept as reusable UML templates, more specifically by proposing a class diagram template for RBAC and use object diagram templates to specify RBAC constraints.

authUML [4] is a smaller framework in the same category as SecureUML and UMLsec which lets the modeler formally model and analyze access control requirements in UML use cases. Also within the area of aspect-oriented design there exists methods for analyzing the consequences of security concerns to other functional concerns [48], and in [116] the authors propose a method for visual security protocol modeling to be used within model-driven architecture.

3.5.4 Simple Technique for Illustrating Risk

The simple technique for illustrating risk (STIR) [114] modeling technique uses the UML use case diagrams to illustrate assets, threats and safeguards (existing countermeasures against risks). STIR has similarities with the UML profile, but the technique has fewer concepts and a different way of modeling. Figure 25 from [114] is an example of an ATS diagram (assets-threat-safeguard) that is used to get a complete overview of the threats for each asset and their protection by safeguards. The ATS diagrams are then examined to see whether the assets are sufficiently protected and the findings can be used as input to a more detailed risk analysis.

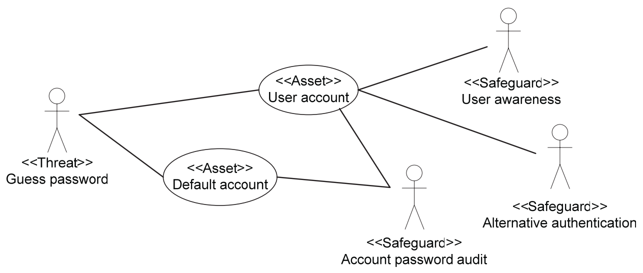


Figure 25 – Asset-threat-safeguard diagram

3.5.5 Riskit Graph

The *Riskit method* [97] includes a risk modeling technique based on a graph notation that makes it possible to specify factors that may influence a software development project (example in Figure 26 from [97]). The Riskit method deals with project risks and has main focus on supporting software development organizations in *developing* their products. The evaluation and management of risks that might occur during the operation of software has therefore been left out [97] (p. 12). Riskit uses its own definitions inspired by for instance organizational strategy research [45]. A *factor* may be compared to a threat scenario in UML profile, while the *event* is an unwanted incident. *Reaction* can be compared to consequence, and the *effect set* can be seen as a further detailing of the consequence. Riskit lacks "threat", possibly because it unusual to consider deliberate harmful actions towards a software development project when assessing the project risk. It also lacks the notion of "vulnerability".

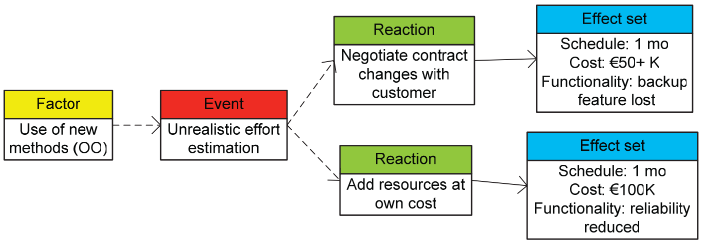

Factor
Use of new methods (OO)
Event
Unrealistic effort estimation
Reaction
Negotiate contract changes with customer
Reaction
Add resources at own cost
Effect set
Schedule: 1 mo
Cost: €50+ K
Functionality: backup feature lost
Effect set
Schedule: 1 mo
Cost: €100K
Functionality: reliability reduced

Figure 26 – Example of a Riskit graph

3.5.6 Microsoft's Threat Modeling and Related Approaches

In [59, 60, 153] Microsoft present what they call *threat modeling for software applications*. The process involves defining threats to a system, ranking them according to their risk level, and finally choosing between different techniques to mitigate them. By using their threat model STRIDE, the risk analysis will be focused towards particular threat scenarios (i.e. Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, and Elevation of privilege). To support the process they make use of data flow diagrams [38, 46] to describe the target, and a kind of tree notation to rank risks (quite similar to attack trees). The threat modeling takes place in the design phase to help reveal potential risks, but it is also claimed to be helpful in code review and testing.

In [121] another method called threat modeling is presented. The process resembles [153] but claims that *the sequence and description of steps is different and the execution of steps is extended to suit complex, networked systems*. The threat modeling is used as a basis for defining security requirements to a system and consists of three steps: (1) characterizing the system, (2) identifying assets and access points, and (3) identifying threats. Only step 1 seems to involve modeling, the other two assess the models from step 1 using check lists for common threats, vulnerabilities, attack goals etc. Attack trees [138] are mentioned during threat identification, but only as an additional mean that may be used to support the process. The outcome of the process is a threat profile for the system that is used for security requirement elicitation.

The process presented in [169] is claimed to be a lightweight formal complement to Microsoft's threat modeling approach. The process focuses on modeling functions, threats, and threat reducing efforts and then it checks the consistency between security threats and functions. Finally, it verifies the lack of threats in the refined model of indented functions and threats that have been mitigated. The process employs high level Petri nets (Predicate/Transition nets) [47], a formal method with both a graphical as well as a mathematical notation, often used to describe distributed systems.

3.5.7 Summary

This section summarizes the various modeling techniques that may be relevant to a new security risk modeling language.

Modeling technique	**Is the technique relevant to a new security risk modeling language?**
Misuse case	Yes, misuse cases were a source of inspiration for the UML profile and its ideas and notation will also be relevant to a new language.

UML variants for security	Partly, since these notations deal with risk analysis on much more detailed system specification level than our approach will do. They are however interesting as notations that can be used if there is a need for further specializations detailing of the threat scenarios.
STIR	Partly, because the UML profile already covers the modeling aspects of STIR.
Riskit graph	Partly, since the notation mainly addresses project risks and it becomes a bit awkward to "force" project related concepts like tools, techniques, resources and project schedules into a security analysis focused setting like the CORAS method.
Microsoft Threat Modeling	Yes, since the intention of dataflow diagrams and attack trees already may be illustrated in the UML profile as threat scenarios and unwanted incidents. It may also be interesting to investigate the possibility of complying to this approach since it represents a major, international threat modeling approach.

3.6 Information Visualization Techniques

The phrase "a picture is worth a thousand words" is claimed to bee a too simplistic view [164] since the usefulness of pictures highly depends on the situation at hand, and the skills of the reader. Nevertheless, software engineering studies have shown that applying graphical means to program- and component specifications increases the understanding of the system [18], and one has successfully applied graphical means to visualize the program interdependencies [109]. According to [106], the key activities for a reader of a diagram is searching and recognizing relevant information, and then using this to draw conclusions. Many graphical means and mechanisms may help the reader with these activities. We were inspired by research within the fields of information visualization in computer displays and statistical graphs/figures, related modeling languages, diagrammatic reasoning and model quality.

One of the key issues when documenting risks is the ability to show which methods a threat may use to initiate risks, and which of these are more likely than the other are. In many ways, this can be compared to showing different paths through a graph, and therefore we refer to it as graph navigation. It is also useful to be able to illustrate the weaknesses or deficiencies that make one vulnerable to threats, and finally one should be able to provide a summary of the most crucial risks. In the following we motivate and present the graphical means and mechanisms that may be used in security risk modeling, structured according to these three main issues: *representing graph navigation, vulnerabilities* and *risk*.

3.6.1 Representing Graph Navigation

We refer to the process of understanding and reading diagrams as "graph navigation". When describing the different paths via which a threat may choose to harm assets it may be useful to draw special attention to the paths that they are more likely to follow. It was also desired by the users of the original UML profile to have a way of describing the logical operators AND and OR when two paths are combined. An ambiguous threat scenario situation where logical operators could be useful is described in Figure 27.

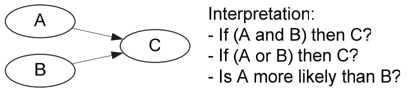


Figure 27 – Typical threat scenario situation

The original UML profile is based on a node-link diagram type, which in its simplest form consists of nodes connected via edges. A closed contour in a node-link diagram generally represents a concept of some kind, and a linking line between concepts represents some kind of relationship between them. Manipulating the appearance of these lines may give them different meanings, for instance the likelihood of the different paths (Figure 28).

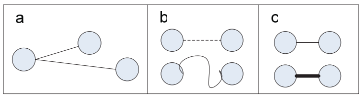


Figure 28 – Node-link configurations

Lines linking closed contours may have different colors, be straight or wavy, or have other graphical qualities that represent an attribute or type of relationship [160]. The thickness of a connecting line may be used to represent the magnitude of a relationship (a scalar attribute). Contrasts in the size or color of a line may make it "pop-out" perceptually and thereby call for the reader's attention [156]. This may be a useful feature to include in a security risk modeling language.

Network traffic maps that often use node-link diagrams to visualize network traffic, also inspired the use of line variations. Typically one illustrates the load or type of link data using different line colors or varying the thickness of the lines [14]. Variation of line appearance is also inspired by the gestalt principle *similarity*, meaning something that is different from its surroundings, will be easier identified by the reader (e.g. *italic style* vs. normal style in text). The gestalt principles [161] are some of the most recognized set of rules for good graphical displays. Implementing the gestalt principles may reduce the effort needed to understand illustrations, program interfaces, websites etc.

When looking for an AND-operator symbol, it was natural to turn to one of the most frequently used technique in risk analysis: Fault Tree Analysis (FTA) [66]. Fault trees let you specify dependencies between events in a tree structure using logical AND/OR operators called "gates" (Figure 29), similar to the ones used in electrical circuit design [140]. One of our goals is to make the security risk modeling language fault tree compliant. In the original UML profile, there was no standard interpretation of two threat scenarios pointing to the same threat scenario or unwanted incident (Figure 27). In order to express fault tree-logic we needed a way of saying both "If A and B occurs, C must occur" and "If either A or B occurs, C must occur". The challenge was to find the preferred way of illustrating this.

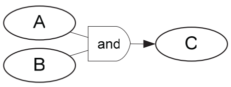


Figure 29 – The AND-gate symbol

An alternative to logical gates may be the UML inspired "information note". In UML one may attach notes to a diagram element to convey additional information about the specific element.

3.6.2 Representing Vulnerabilities

The original UML profile could only illustrate vulnerabilities as properties of their respective assets. There was no way of showing exactly where in the chain of events a vulnerability was exploited, and there was no special vulnerability symbol. Both these aspects were requested by the users of a security risk modeling language.

The way vulnerabilities was modeled in the original UML profile made it simple to see all the vulnerabilities of an asset at a glance, but involved the duplication of vulnerabilities that were shared between two or more assets. Vulnerability is a stand-alone concept, not a binary relation as the ones discussed in the previous section on graph navigation. The descriptive text accompanying the vulnerability is obligatory. The question was whether it should have a symbol in addition to the textual description.

In research on the quality of modeling languages, pictures are claimed to have a much higher perceptibility, and information conveyed in pictures will be emphasized at the cost of textual information [99]. To achieve what [49] terms "syntactic disjointness" , the symbol must be unique and it must neither be too small, nor too large to cause misinterpretations (large elements often attracts attention at the sacrifice of the smaller elements). Syntactic disjointness makes it easier to separate the different elements in a model.

One may argue that the way vulnerabilities was modeled in the original CORAS language was an example of the gestalt principle *proximity*, stating that things that logically belong together should be placed close to each other (commonly used in program interfaces and websites). This view is very suitable if the focus is on the assets and their vulnerabilities, but when it comes to addressing *which* threats exploiting *which* vulnerabilities and especially *where* in the chain of events the exploitation takes place, the situation is different. This kind of information is especially valuable in relation to treatment identification.

Winn [165, 166] has found that people, whose languages are written from right to left, also process graphics from right to left. A related finding is that the item to the left always was interpreted as the cause while the item to the right was the effect [167]. This is in accordance with one of the node-link rules [160] stating that "placing closed contours spatially in an ordered sequence can represent conceptual ordering of some kind" (Figure 28, alt. b), and vulnerabilities may very well be placed within this ordering. The direction of the line should also be indicated with arrows along the line, since this have been found helpful in understanding similar notations like flow charts (used in schematic representations of processes) [29]. If the number of lines increases one may easily loose track of the path and repeated arrows help distinguishing between crossovers and connection points.

3.6.3 Representing Risks

Representing risk is a great challenge to its abstract nature. In the threat diagrams of the original UML profile risks were not illustrated explicitly, but represented as relations between unwanted incidents and assets. Illustrating the severity or magnitude of risks and unwanted incidents helps keeping focus on the most critical risks, something that is especially useful during the final risk treatment phase.

To visualize this issue, one may use means like line thickness, textual information labels, shapes that varied in size or color and more. The color of an enclosed region may represent a concept type (Figure 30) and the size may be used to represent the magnitude of a concept (Figure 31). This is also inspired by the gestalt principle of breaking the pattern to attract attention. The reader of a risk diagram should quickly discover the most serious unwanted incidents, since they often represent major risks.

Figure 30 – Color

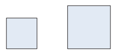


Figure 31 – Size

There are some aspects to be aware of when using color-coding. The number of different colors one may use is limited by the reader's ability to remember and distinguish the colors. Much research has been conducted to find the optimal number of colors, and the suggestions varies from 5 to 8 [27, 31, 32, 142, 163]. One should also have in mind how a model with colored symbols will look when printed in black-white and whether this may affect the interpretation of the symbols [49, 160].

Different types of coding in statistical graphs have been investigated with respect to *display search time*. The coding that gave the best performance was color, the second best coding was shape and then came letters/digits [30]. This top ranking of color has been confirmed by many other studies [88]. According to these results, one should benefit from using color to emphasize the most serious incidents, meaning that the reader should identify them more quickly compared to using other means. As in the case of colors, the number of different size-categories is not indefinite. The number of different size steps that can be distinguished from each others at a glance is as low as four (p. 182). An important aspect when using shapes is to avoid symbols of similar shapes. Shapes that are too similar have been found to increase search time and are therefore not recommended [148, 160].

Studies of *rapid processing of information* [28, 137] have found letters and digits to be some of the best coding forms. These investigations involved Sternberg tasks[1] which studies short-time memory and how graphical means are memorized. Digits, colors and letters were found to be the top three coding forms for rapid processing of information. Another investigation found digits, words and letters to be superior to colors and shapes when it comes to processing speed, and digits, letters and words were found to represent less subjective workload than colors and shapes [155]. These findings support the use of textual information in graphical modeling.

In some general rules regarding the use of shapes are described. For instance that the shape of a closed contour can be used to represent a concept type (Figure 32). Risk was one of the few relevant concepts for which the original UML profile did not have a special symbol.

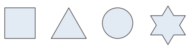


Figure 32 – Shapes

In [19, 160] the authors address evaluation of languages in terms of how easy they are to understand for users that are new to it. They look at it from the perspectives of philosophy, linguistics and cognitive psychology. They use a selection of the cognitive dimensions of [51, 52] to assess a languages ease of understandability. One of the dimensions is the number of different symbols in the language. If the number is high, the language will be

[1] For a limited time the subject is shown a set of objects which has to be memorized.

more difficult to learn, but the number of symbols must be high enough to ensure the language's expressive power. Other dimensions are the consistency and the discriminability of symbols in the language. Both inconsistent symbols and too similar symbols may cause misunderstandings. The degree of "motivation" of the symbols in the language plays an important role related to understanding. In a motivated language there exist a natural relation between the elements of the language and the concepts they represent [135]. The symbols in a motivated language will bear meaning in themselves and therefore make it easier to understand. As the authors point out, this is a challenge in software representations since it is often the effects and interfaces of the system that are interesting, not the "thing" itself. Motivation is also dependent on the culture the symbol is used in (e.g. a red cross vs. a red half moon).

3.6.4 Summary

This section summarizes the graphical mechanisms that may be used in a security risk modeling language, and also for what purpose they may be used.

Mechanism	**Should be tested for the following purpose**
Size	Size may be used to show likelihood of paths in a graph, the magnitude of threats, vulnerabilities, threat scenarios, unwanted incidents and risks.
Text labels	Text labels may be used to give additional information to the reader, possibly about magnitude or seriousness of risks, unwanted incidents and more. It may be used in combination with more graphical mechanisms.
Color	Color may be used to show the magnitude of threats, vulnerabilities, threat scenarios, unwanted incidents and risks
Shapes (symbols)	Intuitive shapes or symbols may be used to illustrate important concepts which contributes to the risks, e.g. vulnerability, different types of threats, assets or similar.

4 Research Method

This chapter describes the research method used in our thesis work. We first explain the concept of technology research, its methods and evaluation techniques. Then we present and motivate our instantiation of this, in particular how the results have been evaluated.

4.1 Technology Research

Within research on methods, tools, languages etc. for use in system development there are two main research paradigms: the behavioral-science paradigm and the design-science paradigm [56, 113]. The behavioral-science paradigm originates from research methods within natural science where it is used to develop and refine principles and laws. This is also considered to be the "classical" way of doing research [151] as we know it from mathematics, physics, chemistry and the social sciences. This type of research aims to explain phenomena in the world around us. When used in relation to system development it focuses on organizational or human aspects. The second research paradigm, design-science derives from engineering and the sciences of the artificial [144]. It aims to solve a problem by developing new artifacts which in relation to e.g. IT systems are defined to be *constructs, models, methods* and *instantiations* [113].

The research process used in behavioral-science can be summarized in three steps: (1) problem analysis, (2) innovation and (3) evaluation [151]. In the problem analysis phase the need for new explanations or adjusted theory is identified. The innovation involves coming up with new explanations or extend already existing theories. The evaluation should provide support to the new theory, i.e. show that it is in accordance with reality. This process may be repeated until the research goal is achieved. Design-science research, or technology research as it is called in [151], may use the same process as behavioral-science: the problem analysis focuses on identifying the needs with respect to a new artifact. The innovation consists in inventing and developing this artifact, and finally the artifact should be evaluated against relevant needs to show its usefulness.

Hevner et al. [56] have defined a set of guidelines for design-science in information systems. Our research domain, security risk analysis of computerized systems is also relevant to information systems and can therefore benefit from using the same set of guidelines. The guidelines are meant as assistance for conducting effective design-science research:

1. Design-science research requires the creation of an innovative, purposeful artifact.
2. The artifact must be for a specified problem domain.
3. Thorough evaluation of the artifact is crucial.
4. The artifact must be innovative, solving an, until now, unsolved problem or solving a known problem in a more effective or efficient manner.
5. The artifact itself must be rigorously defined, formally represented, coherent, and internally consistent.
6. The process by which it is created, and often the artifact itself, incorporates or enables a search process whereby a problem space is constructed and a mechanism posed or enacted to find an effective solution.
7. The results of the design-science research must be communicated effectively both to a technical audience and to a managerial audience.

As stated above, a thorough evaluation is crucial and in the section below we survey research methods for evaluation.

4.1.1 Methods for Evaluating the Results

As highlighted by McGrath [118], an ideal evaluation method should offer high precision, be realistic and facilitate generality. Unfortunately, these three aspects are fundamentally different from each other and cannot be reached using one evaluation method only. As Figure 33 shows, an evaluation method that gives high precision like a laboratory experiment is low on both realism and generality. The challenge for the researcher is therefore to select a set of evaluation methods that brings sufficiently support to the research results. McGrath classifies evaluation methods into eight categories as follows:

- *Laboratory experiment:* is an attempt to create a copy of an existing system where the researcher can control and possibly isolate (all) the variables to be examined. The method can give precise measurements, but lacks realism and its findings are not easily generalized.
- *Experimental simulation*: is a laboratory test aiming to simulate a real situation where the subjects participate for research purposes. When it comes to strengths and weaknesses, the method lies between a field study and a laboratory experiment.
- *Field experiment*: is a field study carried out in a natural environment, but in which the researcher intervenes and manipulates a certain factor. The method lies between a field study and a laboratory experiment when it comes to strengths and weaknesses.
- *Field study*: consists of direct observations of a "natural" (existing) system, with little or no interference from the researcher. The method is realistic, but lacks precision and cannot easily be generalized.
- *Computer simulation*: is to create a model of the real world. The results are low on precision, but score on realism and can to a certain degree be generalized.
- *Non-empirical evidenc*e *(formal theory)*: is argumentation based on logical reasoning rather than empirical evidence. The results from using the method are suitable for generalization, but scores low on realism and precision.
- *Survey*: is a collection of information from a broad and carefully selected group of informants. The results from a survey may be generalized, but there is less focus on realism and precision.
- *Qualitative interview (judgment study)*: means collecting information from a few selected individuals who provide more precise and detailed answers than obtained through a survey. The method gives results that are more precise than those of a survey, but cannot be generalized to the same degree and is low on realism.

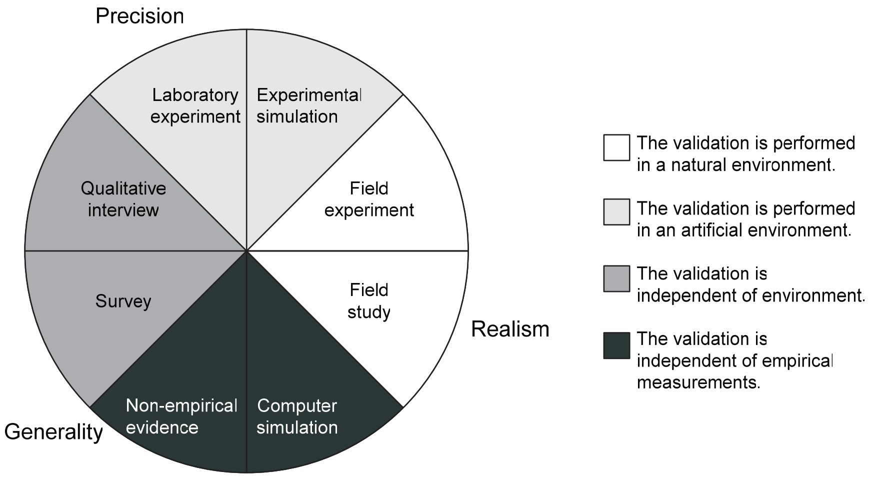

Precision
Laboratory experiment
Experimental simulation
Qualitative interview
Field experiment
Survey
Field study
Realism
Non-empirical evidence
Computer simulation
Generality
The validation is performed in a natural environment.
The validation is performed in an artificial environment.
The validation is independent of environment.
The validation is independent of empirical measurements.

Figure 33 – Evaluation methods [118]

4.2 Our Research Method

As already explained in *Chapter 3*, our thesis work has been carried out in the context of the research project SECURIS funded by the Research Council of Norway. The research activities in SECURIS have been driven by industrial filed trials. Each year from 2003 until 2007 there was 1-3 such trials. Methods and tools developed within the project were thereby tested out in real industrial settings to ensure that they met the needs of the industrial partners. Within the SECURIS project, and also in this thesis we refer to these trials as field trials although they were more like experimental simulations in the classification of McGrath.

The overall goals of our thesis work is characterized in the problem description in *Chapter 2* as a set of success criteria. The problem description was initially very general and unspecified, but as the research progressed it was refined into concrete criteria. This iterative and incremental research approach has been used throughout our thesis work. More specifically, for each research challenge that occurred we have typically proposed one or more solutions and evaluated these to select the most promising solution for further work. This iterative process has resulted in four new artifacts, but also new knowledge that have general interest beyond our specific approach to security analysis.

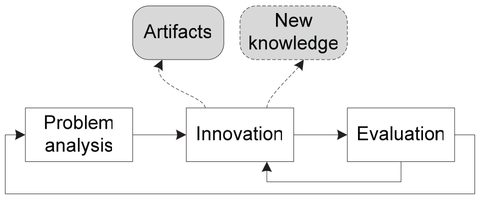


Figure 34 – Our research method

In the following we present the research methods used for each of the artifacts in the order they were presented in *Chapter 2*:
1. A common basis for security analysis terminology
2. A language for specifying and documenting security analysis findings

3. Modeling instructions for the user
4. A way of evaluating the quality of security risk modeling approaches

It is important to note that even though these artifacts are treated separately below, they were in fact developed in parallel, and one artifact may build on another (e.g. the language builds on the conceptual foundation).

4.2.1 Developing a Common Basis for Security Analysis Terminology

The CORAS security analysis method already had a conceptual foundation for security analysis terminology. However, this conceptual model had not been subject to empirical investigations addressing understandability.

In order to improve the original model we investigated it in two iterations, first using students as subjects (Empirical investigation of terminology I), and then professionals within system engineering and risk analysis (Empirical investigation of terminology II). *Chapter 5* summarizes the two studies, while the full details are available in *Appendices A* and *B*.

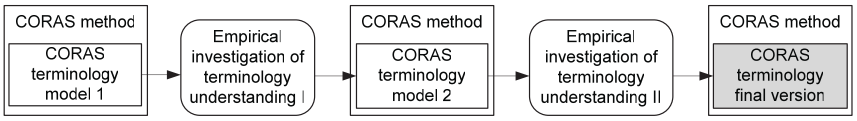

CORAS method
CORAS terminology model 1
Empirical investigation of terminology understanding I
CORAS method
CORAS terminology model 2
Empirical investigation of terminology understanding II
CORAS method
CORAS terminology final version

Figure 35 – Method used for developing the foundation

By investigating alternative solutions of the conceptual foundation, the evaluation became a part of its development. Both investigations were surveys based on questionnaires, where the knowledge of risk analysis terminology was tested. According to McGrath, surveys give results that may be generalized, but they will be less precise and low on realism. In this case we argue that the realism of the evaluation was higher than normal since we investigated the intuitive understanding of the concepts, something that may be relevant in any context. The precision of the results is however low since surveys written in natural language may give room for misunderstandings. Questions that seemed to have caused trouble for the subjects were discarded from the analysis. By doing this we reduced the possibility of misinterpreting the findings, which can be seen as an attempt to increase precision.

In the following tables, we present the set-up for the two investigations, including a description of subjects, material, analysis method, and how the results have been used.

Table 1 – Set-up for terminology investigation 1

Subjects:	31 master students in systems engineering from the University of Oslo.
Material:	A questionnaire with 20 short statements. Each covered the relationship between 2-4 security analysis concepts (in the conceptual model), and was either correct or incorrect.
Measuring:	The score for each statement, looking for particular easy or difficult concept-relationships.
Results used for:	Redesign of the conceptual model.

Table 2 – Set-up for terminology investigation 2

Subjects:	57 subjects: • 12 professional researchers, including 2 industrial psychologists, from the Institute for Energy Technology (http://www.ife.no), who had competence on developing safety critical systems for nuclear power plants. • 22 professional researchers, including 3 industrial psychologists, from SINTEF ICT (http://www.sintef.no) with competence on software development, modeling, and usability studies. • 23 master students in systems engineering from the University of Oslo and the University College of Østfold.
Material:	A multiple choice questionnaire with 16 questions or statements about risk analysis terminology based on the conceptual model that had been redesigned since terminology investigation 1.
Measuring and analyzing:	The data was analyzed with respect to three aspects: • to what extent do the subjects answer correctly without being properly trained? • is there any difference between the score of the students and the professionals? • which concepts and relations are particular difficult or easy to understand? We used the definitions of the conceptual model to decide whether an answer was correct or incorrect, and the results were given as percentage of all the subjects. The difference between the students and the professionals was investigated using a one tailed t-test, with a significance level α=0.05. 57 subjects is considered to be sufficient for assuming normal distribution and therefore a parametric t-test may be used (the limit is said to be about 30 subjects [159]). To identify particularly difficult or easily understandable parts of the terminology similar questions were grouped into a total of six groups and the differences between these six question groups were investigated using the Wilcoxon signed rank test [159] (p. 614), also with a significance level α=0.05. In this case a more conservative, non-parametric test was used since the number of questions for each category was different and therefore the data was not expected to be normally distributed. A non-parametric test is more conservative than its parametric alternative in the sense that it needs a larger sample set to discover differences than a parametric, but can on the other hand be used for smaller sets and does not assume anything about the population of the sample set. SPSS v13.0 on Windows XP was used for all the statistical tests.
Results used for:	Confirming the design of our conceptual foundation which forms the basis for the graphical security risk modeling language, in addition to pointing out possible final adjustments.

4.2.2 Developing a Language for Specifying and Documenting Security Analysis Findings

With the UML profile (Sect. 3.5.1) as the starting point, an improved language was developed. The process has been iterative and incremental (Figure 36) in the sense that the language has been tested in several field trials, each time in a improved version, based on the previous field trials and also findings from other investigations. After completing the final version of the graphical language, a structured semantics has been defined (Sect.

8.1.4) capturing the intended meaning of the CORAS diagrams in terms of paragraphs in English.

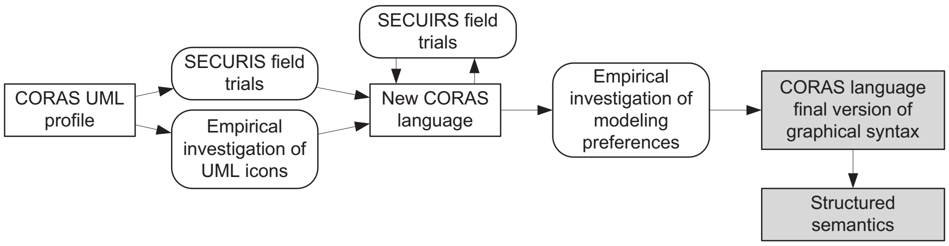

SECUIRS field trials
SECURIS field trials
CORAS UML profile
Empirical investigation of UML icons
New CORAS language
Empirical investigation of modeling preferences
CORAS language final version of graphical syntax
Structured semantics

Figure 36 – Method used for developing the security risk modeling language

Different evaluation methods have been used during the development of the CORAS language. The evaluations have given valuable input for further research and redesign. The evaluation methods used were: SECURIS field trials (what McGrath refers to as experimental simulation), "classroom experiment" (close to laboratory experiment), survey and a "modeling benchmarking test" (a kind of experimental simulation).

SECURIS field trials
The field trials in SECURIS also falls within the category of action research [13, 37] since the researchers played the role of analysts and therefore to some extent were within the scope of the object studied. The researchers facilitated and led the analysis, while the client decided the scope of the analysis and provided people and resources. Each security analysis required about 250 person hours from the analysis team, and 50-100 hours from the client. The security analyses produced two reports, one confidential report presenting the findings regarding security risks, and one presenting lessons learned from applying the CORAS method and the current version of the security risk modeling approach. The results were used to improve both method and modeling approach before the next SECURIS trial.

The first application of the UML profile in real security analyses provided experience with practical use, and formed the basic requirements to such a language. In the subsequent field trials, improved versions of the modeling language were tested and a modeling practice was gradually established. The field trials were a unique opportunity to test the method and modeling on real security issues since it was integrated in the client organizations regular work on security. The client and its representatives had a genuine interest in the findings from the analysis, and the results influenced decisions beyond the scope of the actual analysis, in some cases it had effects throughout the whole organization. To get the participants personal view on the method and modeling, we collected written feedback via evaluation forms.

Classroom experiment
One of the innovations of the UML profile was the use of specially designed icons related to security risks. Our first investigation aimed to gather support for this design decision by comparing models with and without special icons to identify differences. The special icons should help inexperienced users to understand the language better and faster, and therefore master students in system engineering were used as subjects. (See also *Chapter 5* (summary) or *Appendix A*.)

Survey
After several field trials, the language had "stabilized" and it was time to make the final design decisions regarding the modeling style and its graphical representation in the form of concrete syntax. To identify the preferences of the users, we presented several ways of modeling the same scenarios to a group of subjects and asked them to select the ones they preferred. We also tried to learn from related research within other domains; in particular, we made use of recommendations from the field of from graphical information visualization. (See also *Chapter 5* (summary) or *Appendix G*.)

Modeling benchmarking test
The modeling benchmarking test consists of a set of core security risk scenarios, defined on the basis of field trials and more general security risk related information (*Chapter 6*). The collection of scenarios is an example of information gathered during a typical security analysis. Modeling the core security scenarios with the CORAS language may be seen as kind of experimental simulation in McGrath's terminology.

With respect to Figure 33, we note that the icon investigation scores high on precision, the investigation of modeling preferences scores on generality, and the SECURIS field trials and the modeling-benchmarking test are fairly strong on realism. However, these evaluations have focused on slightly different aspects of the same modeling approach. The CORAS security risk modeling approach consists of both a language (syntax + semantics) as well as a modeling guideline, all strongly connected to the CORAS security analysis method. It is therefore unfeasible to continue focusing on separate evaluations of the different parts, and instead consentrate on experiences from practical application of the full CORAS approach.

The validity of the structured semantics has been ensured by using diagrams from real field trials as input for the pattern-matching rules. Since the structured semantics covers every legal combination of constructs, the developers had to precisely define de permitted modeling options and styles. This also provided valuable input to the modeling guideline. The research has so far completed the set of rules, and the next step will be to evaluate the diagram translation with respect to understandability and more.

In the following tables, we present the set-up for the classroom experiment (the icon investigation) and the survey (investigating modeling preferences), including a description of subjects, material, analysis method, and how the results have been used.

Table 3 – Set-up for the icon investigation

Subjects:	25 master students in system engineering from the University of Oslo. The experiment took place as an exercise in the master course "Unassessable IT systems".
Material:	The icon experiment material consisted of two sets of security risk scenarios modeled with the UML profile, one using standard UML icons and one using CORAS icons, and a related questionnaire. 13 of the 25 subjects received models with CORAS icons, 12 received standard UML icons. Both sets were stereotyped[2] with stereotype names (see Figure 37). The material had three parts: • Part 1 and 2 consisted of questions related to whether the subjects

[2] In UML one uses stereotyping to label elements of different kinds. This can be either textual, with icons or both. "<<Actor>>" is an example of textual stereotyping that often is added to the pin-man icon in UML.

	were able to quickly identify specific elements like treatment, risk, threat agent etc. in a model (i.e. whether they are able to navigating the models). The questions were of the type "How many assets are explicitly modeled in this diagram?" • Part 3 focused on how the models were understood and interpreted. An example of the type of question used here is "Which unwanted incidents affect the asset "Web server" in this model?" For part 1 and 2 the subjects had only a limited time to answer the questionnaire (3 minutes and 1,5 minutes) to stress the point that they should quickly identify the different diagram elements. For part 3 they had more time available (15 minutes).
Measuring and analyzing:	The two groups of subjects were compared to see whether the one with CORAS icons received a higher score than the group without. In the same manner we analyzed whether one group managed to complete more questions than the other group, or used less time pr. model-question-set. Since the number of subjects that answered within each part of the questionnaire decreased for each part, in addition to an overall limited sample set, we used a non-parametric test to ensure that our statistical analysis was conservative. The non-parametric Mann-Whitney test was used with alpha level = 0.05.
Results used for:	To decide whether one should continue to use special graphical icons in the security risk modeling language because it had positive effects on the interpretation of the models.

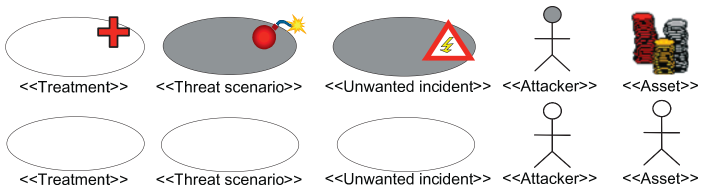


Figure 37 – Icons used in the icon investigation

Table 4 – Set-up for the modeling preferences investigation

Subjects:	33 professionals within system engineering: The majority were males between the age of 26 and 35 with 1-3 years of work experience.
Material:	The material was in the form of a questionnaire with alternative representations of a CORAS threat diagram. The diagram was a simplified version of a real threat diagram. The subjects were to prioritize between different modeling alternatives using the set-up shown in Figure 38. The survey was distributed via e-mail to people in various parts of the software industry.

Measuring and analyzing:	The investigation targeted three unsolved issues: • How should we visualize frequency measures in threat diagrams to ease graph navigation, especially the likelihood of single relations and combined relations? • What is the preferred way of visualizing vulnerabilities? • How can we visualize risks in the models to improve the understanding of this abstract concept? The null hypothesis and alternative hypothesis for all tasks were: H_0: the modeling alternatives are equally preferred by the subjects. H_1: the modeling alternatives are not equally preferred. The data was recoded into "0" and "1" to reflect which diagram that was preferred: a mark in one of the three fields to the left (see Figure 38) was transformed into "0" and marks in the three fields to the right were recoded as "1". The middle field was interpreted as missing data to obtain a cut-off point between the two sides. The frequency of 1's and 0's could thereafter be compared. The data was in the form of frequencies and not distributed normally; therefore the non-parametric Chi-Square test [143] was used to identify possible statistical differences between the representations with a significance level of 0,05. The Chi-Square test uses the assumption that the frequency of 1's and 0's are the same. SPSS v13.0 on Windows XP was used for the statistical tests.
Results used for:	Deciding how the final version of the security modeling language should look like with respect to design and layout for particular modeling tasks.

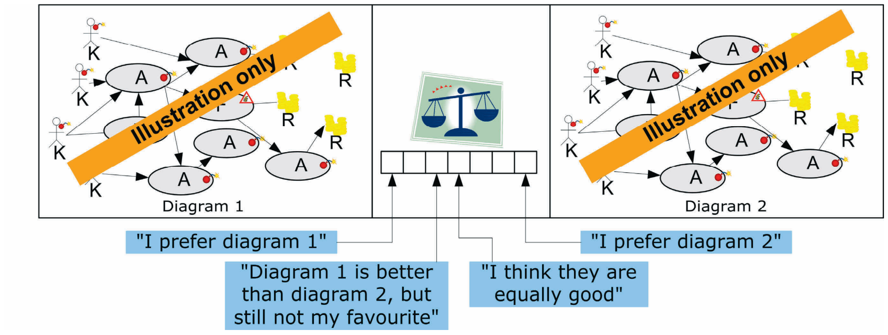


Illustration only
Diagram 1
Illustration only
Diagram 2
K
A
R
"I prefer diagram 1"
"Diagram 1 is better than diagram 2, but still not my favourite"
"I think they are equally good"
"I prefer diagram 2"

Figure 38 – Instructions for filling out the modeling preference tasks

4.2.3 Developing Modeling Instructions for the User

In order to use the security risk modeling language to its full potential there is a need for a modeling guideline, particularly targeting inexperienced users. The modeling guideline can be seen as the link between the language and the method, and changes to either one of them affects the guideline. The early use of the UML profile in the CORAS project had established a kind of modeling practice. This practice was applied in the SECURIS field trials, each time in a revised version based on experiences from the previous field trial.

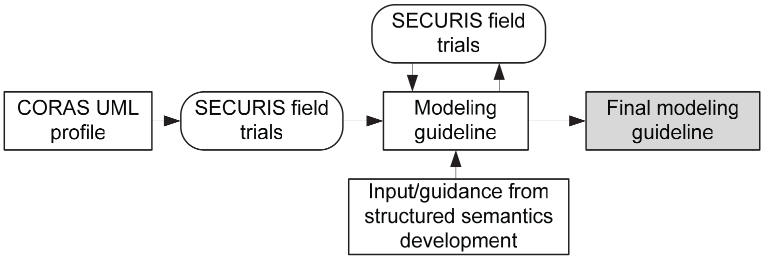


Figure 39 – Method used for developing the guideline

The evaluations of the modeling guideline are based on lessons learned from field trials, experimental simulation in terms of consistency checking against the structured semantics and modeling the benchmarking test case.

The SECURIS field trials differed a lot with respect to target, technology, scope and complexity. It was therefore unfeasible, and also impossible to define a strict guideline at an early stage. As the field trials progressed, the guideline was adapted to handle the various challenges that arose.

The development of the structured semantics provided valuable input to the modeling guideline. Before the structured semantics, the modeling style for each diagram type was a bit underspecified. The formal style of the structured semantics forced the developers to decide which combinations of constructs that should be allowed in a diagram. The guideline had to ensure that it did not conflict with the permitted and unacceptable constructs in the language. In this way, the structured semantics worked as a consistency, or quality check of the guideline. This work made the modeling guideline more complete and also helped define the exact content and purpose of each of diagram type.

In the same manner as the language itself must handle the core security risk scenarios (benchmarking test), it should be easy to arrive at these models by following the modeling guideline. This can be seen as a kind of experimental simulation where the use of the guideline is simulated. This simulation made it easier to identify inconsistencies, lack of guidance or similar, in the guideline.

It would have been extremely valuable to apply the guideline in a field study *without* the involvement of the researchers. After the field study, there should have been a qualitative interview of the users to obtain in-depth information about their experience with the guideline. However, this was not practical within the setting of the SECURIS project.

4.2.4 Developing a Way of Evaluating the Quality of Security Risk Modeling Approaches

The quality framework for security risk analysis modeling approaches has been developed to evaluate the new CORAS language, an also to make it easier for others to evaluate the quality and expressiveness of security risk modeling approaches.

Three main sources of information were used when developing the quality framework:

- SEQUAL [100-104] a quality framework for modeling languages within information modeling.

- Modeling needs within security analysis in general, particularly with focus on weaknesses in existing risk modeling notations.
- The use of the UML profile during field trials.

Evaluation of the framework has been made in two iterations; first integrated as part of its development, and second by applying it to the UML profile and the new CORAS language.

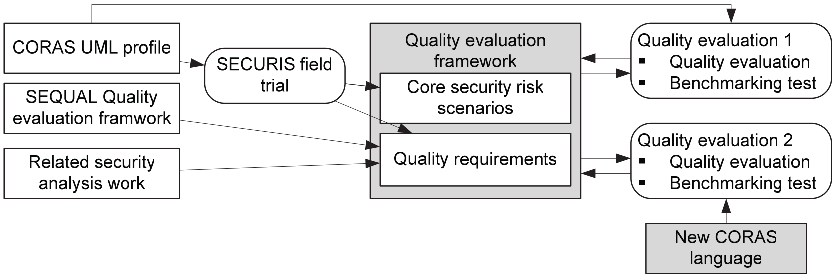


Figure 40 – Method used for developing the quality framework

When specifying the language requirements we used SEQUAL as basis. SEQUAL has been developed for comparing modeling languages, identify areas where a language needs improvements etc. In addition to the general quality requirements, more security analysis specific requirements were included into the SEQUAL structure. These were based on relevant modeling needs of existing security risk- and risk analysis methods. A security risk modeling language must be able to express realistic and relevant security risk scenarios. In the SECURIS field trials we have gained experience that we have used to construct a complete example of informal security analysis documentation called "the core security risk scenarios". (See also *Chapter 6*). The evaluation of the quality framework during its development is based on non-empirical evidence. Each requirement is followed by a characterization of the rationale for its inclusion into the framework (see *Chapter 6*). This makes it possible for a potential user of the framework to understand the background for the requirement and judge the relevance of a given requirement.

To evaluate the final quality framework it was first applied to the UML profile (See summary in *Chapter 7*, [57]). The results were important to the problem analysis (See *Chapter 2*) since it gave valuable insight into the weaknesses and strengths of the UML profile, and consequently which parts that we had to improve in the new CORAS language. The second evaluation of the framework was to apply it to the new CORAS language (*Chapter 9* and *Appendix H*). These two evaluations may be classified as experimental simulations where the framework is tested against real modeling approaches. After two evaluations, there were indications that some of the requirements originating from information modeling and some of the early requirements to security risk analysis modeling, were not as relevant as they had seemed[3]. An example is the general requirement that all concepts in a language must have a special symbol. Experience with the CORAS method in practice, does not support the request for a special symbol for e.g. "consequence", - but we cannot exclude that other approaches may need it. Because of this, we chose to keep these requirements in the framework since they may be relevant to other kinds of security risk modeling approaches than CORAS.

[3] The requirements that have been characterized as less relevant are listed in the framework description in *Chapter 6*, and also in *Appendix H*.

5 Initial Investigations

This chapter presents the initial investigations the results of which heavily influenced the design of the CORAS language. We first conducted a study of the interpretation of the conceptual model already developed in the CORAS project (5.2.1) prior to the start-up of our thesis work. On the basis of this investigation we came up with a revised conceptual foundation. We later conducted a second investigation focusing on improving finding ways to improve the understandability of certain important concepts (5.2.2). The last of the initial investigations was an experiment testing whether to use specialized graphical icons (5.3).

We start by explaining the original conceptual model developed in the CORAS project (5.1).

5.1 The Original Conceptual Model of the CORAS Project

Security analysis uses a terminology that includes general risk analysis concepts as well as more specialized security analysis terminology. A conceptual model may be used to explain how a concept like risk relates to for example threat or vulnerability. In the CORAS project, the conceptual model also played an important role as a kind of abstract syntax for the CORAS UML profile (presented in Sect. 3.5.1).

The original conceptual model of CORAS in Figure 41 using the class diagram notation from the Unified Modeling Language (UML) [125]. The associations between the elements have cardinalities that say how many instances of one element that can be related to one instance of the other (example: "a stakeholder has at least one and maximum an infinite number of assets; and an asset belongs to only one stakeholder").

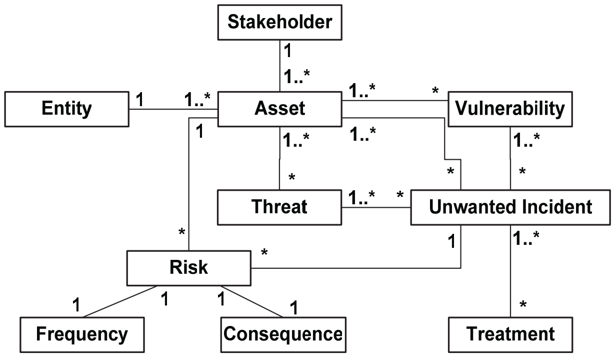


Figure 41 – The original conceptual model of CORAS

The concepts used in the model are defined as follows:

- **Stakeholders** are those people and organizations who may affect, be affected by, or perceive themselves to be affected by, a decision or activity regarding the target system [7].
- An **asset** is something to which a stakeholder directly assigns value and, hence, for which the stakeholder requires protection [54]. A stakeholder wants to protect his/her assets from being compromised.

- An **entity** is a physical or abstract part or feature of the target of evaluation that becomes an asset when assigned value by a stakeholder, for example a service provided by the target [112].
- A **vulnerability** is a weakness of an asset or group of assets which can be exploited by one or more threats [76].
- An **unwanted incident** [54] is an event that may harm one or more assets and something one wants to prevent. An unwanted incident is the result of a threat exploiting a vulnerability.
- A **threat** is a potential cause of an unwanted incident [76] i.e. someone or something that wants to harm, remove or interfere with the asset.
- A **risk** is the chance of something happening that will have an impact upon objectives (assets) [7]. A risk is the combination of an unwanted incident, the chance of it happening and its consequence.
- **Consequence** is the outcome of an event expressed qualitatively or quantitatively, being a loss, injury, disadvantage or gain [7].
- **Frequency** a measure of the rate of occurrence of an event expressed as the number of occurrences of an event in a given time [7].
- A **treatment** is the selection and implementation of appropriate options for dealing with risk [7].

5.2 Investigating Concepts and Terminology

5.2.1 The First Terminology Investigation

As described in the previous section, the CORAS method and UML profile had a conceptual foundation defined in a UML class diagram. Several of its concepts and definitions are used in daily language, but often in a different meaning and with no link to its original context. For instance, many of us have probably used the phrase "I won’t take the risk" without giving the definition of risk any thought.

The purpose of the first terminology investigation was to identify the parts of the CORAS conceptual model that are particularly difficult (or straightforward) to understand. The assumption for our investigation was that a precisely defined conceptual model would reduce the misunderstanding related to ambiguous terminology in a security analysis method. The questionnaire that was used tested definitions of concepts, as well as how the different concepts relate to each other (for more about the set up we refer to Sect. 4.2.1*)*.

The results showed that few subjects had problems with the relation between a stakeholder and an asset, or between a threat and a vulnerability. These concepts are part of the daily language and most people have an intuitive understanding of them. Concepts like frequency, consequence and probability created more confusion and a large majority was not aware that probability is kind of frequency. The results showed the importance of using terms that are intuitively understood, not conflicting with the daily language.

On the basis of these results we propose a set of changes to the original model (Figure 41) which are implemented in Figure 42:

- To avoid confusion about frequency, probability and consequence we have included probability as a specialization of frequency.

- In order to emphasize that a risk consists of a frequency value, consequence value and an unwanted incident, these terms have been grouped together in a logical manner that illustrates how they are components of a risk. The black diamond symbolizes that if a risk is eliminated this will also remove the frequency and consequence values (composition). The white diamond means that the unwanted incident is a part of the risk, but if the risk is eliminated the unwanted incident may still exist in other risks (aggregation). The direct relation between treatment and unwanted incident has been removed and instead we have connected treatment to risk. By doing this we specify that a treatment is directed towards a risk, but not whether it targets a vulnerability, a threat, an unwanted incident or a combination of these.
- The association between asset and risk was a major source for misunderstandings. Nevertheless, we have decided to keep the relation because we believe it is important. The subjects seem to think of an unwanted incident in the way we use the term risk, and by removing the direct association between asset and unwanted incident we hope that this misunderstanding will be less common.
- We chose to remove "entity" from the model. The term was often misunderstood (48.9% incorrect or uncertain answers). In addition, it is not natural to speak of general entities in a structured brainstorming, only entities assigned value which then are called assets.
- The association between asset and vulnerability has been changed from a regular relation to become a part of the asset in the sense that an asset can have a vulnerability.
- In the original conceptual model it is not explicitly modeled that "a threat exploits a vulnerability to initiate an unwanted incident". This is an intuitive relation that achieved a high percentage of correct answers and we feel it is correct to make it clearer than it was in the original model.
- The new model tries to emphasize that an unwanted incident is a part of a risk and therefore one or more threats are always associated with the risk through its unwanted incident.
- We have also chosen to highlight concepts grouped together in the form of compositions and aggregations, i.e. vulnerability is tightly connected to asset, and risk is a concept that includes three other concepts.

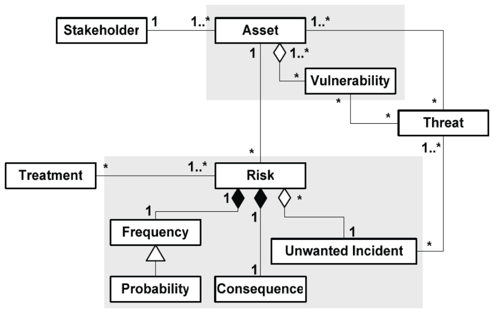


Figure 42 – The first revision of the conceptual foundation

5.2.2 The Second Terminology Investigation

The second terminology investigation considered how the revised conceptual foundation was understood among professionals and students within software engineering and risk analysis. The subjects had various competences in one or more of the following domains:

(1) system modeling and design,
(2) risk analysis and system development in the safety domain and
(3) system development and security.

The investigation used a similar set up as the first terminology investigation. The model in Figure 42 had after the previous investigation been extended with the concept **likelihood**. Likelihood was in [7] defined as a qualitative description of probability or frequency, and we decided to include it as a specialization of frequency.

The findings showed major differences between the various risk-related concepts. Asset and vulnerability related questions received the highest number of correct answers, while the frequency related obtained the lowest number. Many of the terms we have in our conceptual model (Figure 42) are well understood, even by people without training in risk analysis. Being more experienced in risk analysis, the professionals obtained a higher number of correct answers than the students.

The results from this study show that the revised model successfully reflects the subjects' common understanding of the risk analysis terminology. Nevertheless, we made some adjustments based on the findings from this investigation:

- To increase the awareness of non-human threats we have specified threat into human threat and non-human threat.
- The specialization of frequency into probability and likelihood was not satisfactory. According to [6] likelihood is a general descriptions of frequency or probability and can be expressed *both qualitatively and quantitatively*. Even though likelihood covers the other terms, it has turned out as one of the most difficult concepts to understand. The conceptual foundation will therefore not distinguish between the frequency measures (probability/frequency/likelihood), but leave it to the security analysis team to decide which one to use.

The UML 2.0 class diagram in Figure 43 shows the final model (the shaded areas are not part of the UML notation and merely highlight concepts that are closely related to each other).

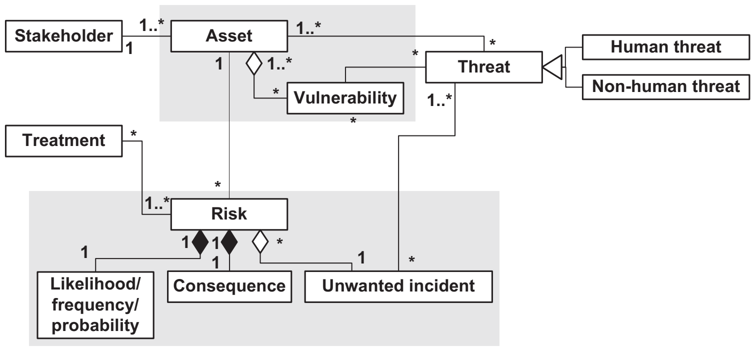


Figure 43 – The final revision of the conceptual foundation

5.3 Investigating the Use of Specialized Graphical Symbols

In addition to the terminology investigations, we explored the effect of using graphical, risk related symbols in risk modeling. This investigation is called the icon experiment.

The UML use case notation has its own set of standardized icons, but, the UML profile is recognized by its specialized graphical icons that symbolize the different terms in the conceptual model. The icons were believed to make the models easier to read and understand. Often graphical icons are seen as merely "decoration" just to make the models look nicer, but it is often believed that they may also improve the understandability of the models.

In this experiment we investigated whether the use of specialized icons make the models easier to understand. The difference between the standard UML icons and the specialized CORAS icons are shown in Figure 44.

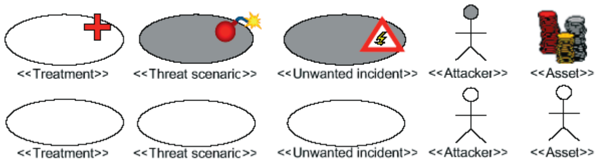


Figure 44 – Standard UML icons and the UML profile symbols

The results from the icon experiment showed that stereotyping with CORAS icons vs. UML icons improves the speed of navigation, i.e. identification of specific model elements. The results showed a statistical significant difference in the number of questions completed in favor of the group using CORAS icons. The icons did not affect the correctness of interpretation of models. Hence, the number of misinterpretations did not increase.

The conclusion from our icon experiment is that using the specialized graphical icons helps the participants in a structured brainstorming session to identify the same model elements faster than if standard icons are used without increasing the number of misinterpretations.

6 Language Requirements Defined within a Quality Framework

In this section we present our requirements to an ideal security risk modeling language in a framework that focuses on the quality of the language. This quality framework consists of two parts, first we describe the core modeling needs related to the different phases in the security analysis process. Then we use a quality evaluation framework for modeling languages called SEQUAL to structure and identify the detailed requirements.

In the following terms like *model, conceptual model, modeling language* and *diagram* are used. A model is a collection of diagrams that describes a system. The diagrams can show different views of the system's architecture, functions, processes etc. and are made with a modeling language. Figure 45 illustrates the relationships between these terms. A conceptual model is a description of important concepts, or terms, and how they relate to each other.

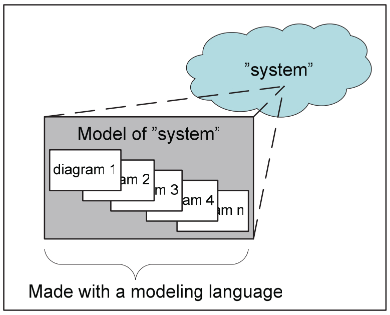


Figure 45 – Relationships between system, model and diagrams

To our knowledge it does not exist any predefined "benchmarking" scenarios we can use to evaluate a security risk modeling languages, therefore we have defined a set of test scenarios representing typical modeling needs identified through experience from several industrial security risk analyses. The phases in CORAS have different purposes and therefore different modeling needs. In the following we give an overview of the general modeling needs for each of the phases and a description of the core modeling needs using a security risk case as a benchmarking test. This test case inspired by, but not similar to, the field trials and should therefore represent realistic security analysis information. The information is structured according to which phase it is gathered in, using the CORAS method. The idea is that a good security risk modeling language should be able to model all aspects of this information, which we call the core security risk scenarios.

6.1 The Core Security Risk Scenarios

In order to evaluate a security risk modeling language we need a benchmarking case representing core security risk scenarios to test the notation against. Through experience from several major security analyses in the SECURIS project we have gathered typical security analysis information into a complete set of scenarios from (1)-context establishment through (2)-risk identification, (3)-risk estimation, (4)-risk evaluation and (5) treatment identification. For each of the five phases we explain what the purpose of the phase is and what tasks it includes. Then we present relevant security analysis information that may be modeled graphically using a security modeling language. The characteristics of the core security risk scenarios are among others:

- There are two stakeholders of the system (one client, one other interested party. The distinction lies in whether the stakeholder is economically responsible for the analysis (e.g. a client like a company) or not (e.g. other interested party like the authorities). Multiple stakeholders in a security analysis makes it more complicated when it comes to judging: *what are the assets of the analysis?* And *what risk level can be accepted*?
- Assets may depend on each others, meaning damage to one asset may indirectly damage another asset. This has also been addressed in the core security risk scenarios.
- Often the statistical material on which to make judgments about how often an incident may occur is limited. In the example case we have used an approach that does not require precise, numerical input for the likelihood of unwanted incidents.
- When a risk is found acceptable, it should be documented but kept out of the treatment identification phase to ensure enough focus on the unacceptable risks.

6.1.1 Phase I: Establishing the Context

In this phase it is important to obtain an understanding of the target of analysis and its assets. The context establishment also focuses on the stakeholder's main concerns regarding target vulnerabilities and threats. The following aspects needs to be modeled (in one or more diagrams)

- Target overview diagram: an overview of the target (system or part of a system) that will be analyzed, annotated with the stakeholder(s), the assets, the main vulnerabilities and the main threats

An overview diagram of this type will help scoping the analysis during the preliminary security analysis and ensure that the correct level of details is established at an early point in time.

The assets are important in the CORAS method and therefore it can be useful to model them explicitly in a separate diagram:

- Asset diagram: describing the assets, how they relate and a ranking of their perceived importance. Assets that are affected by risks often have relations to other assets which indirectly may be affected. Even if some assets are defined to be out of the scope of the analysis, it is useful to model them to see the overall asset-picture.

6.1.1.1 Core Security Risk Scenarios in Phase I

Target description: The target of analysis is a web portal that serves as a communication medium between ordinary citizens and various public entities. The information provided is confidential personal information. The company that develops the portal will gather information from several databases within the public entities. The users authenticate themselves to the web portal using a password and username, while the authentication mechanism is simpler for the developers. Inside the developers network one uses a simple "remember-this-computer" mechanism to access the portal without being prompted for username and password each time. The users are free to set their own passwords without restrictions. Due to the importance of the information provided to its users, the service must be available 24/7. The web portal will gradually put into service and the security analysis will look at the security during development, testing and maintenance.

Stakeholders, clients and other interested parties: There are two stakeholders in this case where the company management is the *client* of the analysis, i.e. the one paying for the analysis. The other stakeholder is the central authority who is considered to be one of the "other parties". Other parties are not paying for the analysis itself, but have the authority to set requirements to the risk acceptance levels. In this case the analysis has been initiated as a consequence of the security requirements from the central authorities and therefore they are included in the analysis:

- **Company management (CM)** that develops and delivers the service, and therefore bears all the costs related to the development and maintenance.
- **Central authorities (CA)** that regulate what information the service should provide and how it should be protected. They have the authority to close down the service if it fails to fulfill their regulations.

Assets: The assets related to the target of analysis are ranged according to its values to the client and other parties. The potential damage a threat may cause to the asset (lost asset value) is not specified in details but described shortly in the parentheses:

The company management's assets:

- **CM1 – Users personal information** (damage is measured by the type of information disclosure, for example major = the information is available to the public for one day, medium = the information is available to the public for one hour, minor = the information is available to an unauthorized employee in the company for a week)
- **CM2 – Company reputatio**n (damage is measured by the type of negative media publicity in, TV, radio, large newspaper or small newspaper, negative rumors etc.)
- **CM3 – Availability of service** (damage is measured as down time of the service)
- **CM4 – User efficiency** (damage is measured as the increased effort needed to use the functions provided by the service)

The central authorities' assets:

- **CA1 – Users personal information** (measured as above)
- **CA2 – Availability of the service** (measured as above)
- **CA3 – User efficiency** (measured as above)

The assets are not independent, meaning if one asset is harmed it may affect other assets. An in some cases an asset has to be harmed first before another asset can be harmed. In this case the first asset is called a *direct asset* and the second an *indirect asset*. The identified relations between assets are:

- Company reputation (CM2) can only be harmed if one of the other assets is harmed first. This means that CM2 is an indirect asset.
- Within the scope of the analysis, only damage to the availability of service may affect user efficiency (CM4, CA3), making User efficiency (CA3) an indirect asset.

Measures: One needs to define several different measures like asset values, likelihood of risk and consequences of risks:

- **Assets**: may be ranged according to their perceived importance, monetary values etc. but is not required.
- **Likelihood** is measured qualitatively in three categories: *seldom* (= 1 time per 5 years or less), *sometimes* (= more than 1 time per 5 years and less than 1 time per year) and *often* (= 1 time per year or more). The categories may be mapped to intervals of probabilities if such data is available and appropriate to use.

- **Consequence** is measured qualitatively in three categories: *minor, moderate* and *major*. The consequence categories should to be mapped to actual damage for each asset. A "minor" damage could in some cases mean "system down in 2 minutes" while in other cases a minor consequence could be "system down in 4 hours". This should be specified in accordance with the client and other parties. In this generic case we will only use the category names since substituting them with numbers or text would have no impact on how it is modeled graphically.
- **Risk value** is measured in three categories: *low, medium* and *high.* The risk value is based on a combination of likelihood and consequence, either as a risk matrix (like in this case) or a risk function that computes a value based on probability and consequence (this requires the consequence to be measured quantitatively).

	Likelihood		
Consequence	Seldom	Sometimes	Often
Minor	*Low*	*Low*	*Medium*
Moderate	*Low*	*Medium*	*High*
Major	*Medium*	*High*	*High*

Figure 46 – Risk value matrix

- **Risk reduction** is measured in terms of decreased risk value (based on reduction in consequence and/or likelihood).

Risk evaluation criteria: The tolerance levels, or acceptance levels, for risks against specific assets are decided already at this stage in the analysis. In this case the two stakeholders, company management and the central authority value the assets differently, and consequently have different risk acceptance levels. These levels are later used during risk evaluation to decide which risks that can be accepted and which that need to be treated.

Table 5 – Risk evaluation criteria

Asset	**Max accepted risk level**	
	CM (Client):	**CA (Other parties):**
User's personal information	low risk	low risk
Company's reputation	medium risk	any risk
Availability of service	medium risk	medium risk
User efficiency	high risk	medium risk

6.1.2 Phase II: Identifying Risks

This phase needs models of two main types: system models (UML etc.) and risk models. The first type depends on the target type and is not part of the core modeling scenarios. The risk models must include one or more diagrams that include:

- the threats and threat scenarios related to assets.
- the vulnerabilities and which threats that can exploit them (i.e. the threat's way "into" the target).
- the unwanted incidents the threats may cause

6.1.2.1 Core Security Risk Scenarios in Phase II

Threats: There are both human and non-human threats, that either can be threats by an accident or have more deliberate motives (these distinctions are in accordance with the security standard ISO/IEC13335 [76]):

- **Company employee (human, accidental)**: an employee may make a mistake causing an unwanted incident or unintentionally infect the server with malicious code during an update.
- **Company employee (human, deliberate)**: an employee may use his or her access rights to intentionally cause an unwanted incident.
- **Hacker (human, deliberate)**: a hacker may want to harm the users or the company for fun or for economical reasons (e.g. blackmailing).
- **Internal infrastructure (non-human)**: hardware or software, part of the service, may fail and initiate unwanted incidents.
- **External resources (non-human)**: resources that deliver data to the service.
- **Virus attack (non-human)**: an environmental circumstance outside the company's control.
- **Web portal service user (human, accidental)**: a user may for example use the service incorrectly

Vulnerabilities:

- **Insufficient authentication mechanisms**: within the company development team the authentication mechanism only requires a username and a password, no secure ID or similar identification. Inside the company network the user is not prompted for username or password when applying the "remember-me" function.
- **System design weakness**: the development environment used by the company has very few restrictions on what an employee may modify and does not provide warnings related to critical updates.
- **Unsecured WLAN**: the company has an open WLAN in their development environment which is possible to detect outside the company building.
- **Too simple password**: there is no control on whether the user of the portal changes his initial password, and if he does there are no rules for how the new password should look like (length and combination of letters, numbers)
- **Shared infrastructure resources**: the service runs on hardware or software that is shared with other less critical services. This means that if one of the other services encounter a problem it may affect the service.
- **Low robustness**: in cases of high traffic to the portal, the server tends to degrade in performance and response time increases.
- **External resource failure:** a resource that provides data to the web portal service may fail, and the service is dependent on the availability of these databases.
- **Internal hardware or software failure:** the internal infrastructure may fail due to hardware or software errors.
- **Insufficient logging**: access and modification of user's personal information is insufficiently logged, meaning that one cannot be sure who has made the changes (i.e. the user or one of the company's employees).
- **Unclear security update routines**: security updates are communicated via e-mail or the intranet and the individual employee is responsible for keeping his or her computer updated.

Unwanted incidents: The unwanted incidents that may happen are listed below followed by a description of the threat scenarios that may lead to the incident (the threat is marked with bold fonts):

U1: Disclosure of users' personal information:

1. An employee may unintentionally modify the system making it disclose personal information of one or more users to all the other users.
2. An employee may use her or his job privileges to access users' personal information and use it for blackmailing (no internal logging in the company).
3. An employee in the company may use another employee's computer that has the "remember-me" function enabled and thereby get access to users' personal information.
4. A hacker may attack the service via the WLAN and eavesdrop to the data transmission.
5. A hacker may exploit the simple password policy to access users' personal information.

U2: Unauthorized modification of users' personal information:

1. An employee may use her or his job privileges to modify users' personal information without being logged.
2. An employee may unintentionally update the system causing it to modify or delete users' personal information.
3. A user may enter information repeatedly if the service's response time is too long, accidentally making the information incorrect.
4. A hacker may attack the data transmission via the WLAN and tamper with users' personal information.
5. Virus attack on the service may cause the server to crash, deleting all active user sessions and their previous data modifications leaving the information partly incorrect.
6. A user may unintentionally enter wrong or incomplete information to the service, affecting already stored data. Without any logging it is impossible to prove who is responsible for the changes.

U3: Unavailability of service due to hackers:

1. A hacker may cause a denial-of-service-attack to the service making it unavailable to both customers and employees.
2. A hacker may use the WLAN to obtain a password and a username and then use this to log in as an employee with authorized access to servers, and therefore be able to tamper with the server or databases.

U4: Unavailability of service due to infrastructure failure:

1. Hardware or software, part of the service infrastructure, may fail or malfunction and make the service fully or partly unavailable.
2. External sources of information may fail or malfunction making the service unavailable.

U5: Unavailability of service due to malicious code:

1. An employee may unintentionally infect the server with malicious code using a false security or operative system patch.
2. A virus attack may cause extensive traffic and thereby make the service unavailable.

U6: Damage to company reputation:
1. If users' personal information is disclosed to media it may harm the company's reputation.
2. If the possibility to modify users' personal information is disclosed to the press it may harm the company's reputation.
3. If the service is unavailable it may harm the company's reputation.

U7: Reduced user efficiency:
1. If the service is unavailable it may reduce the users' efficiency.

6.1.3 Phase III: Estimating Risks

Estimating risks is to provide likelihood and potential consequence estimates for each risk. The modeling needs in this phase are:

- A description of the risks that includes both the threat's method(s) and which assets that are harmed. (giving a possibility to annotate threat scenarios and unwanted incidents with likelihood estimates.)
- A description of the associations between an unwanted incident and an asset, representing risks. This can be annotated with the most likely consequence value (e.g. "loss of 1-10K €", "10-20% reduced user efficiency") reflecting the scale the asset is measured in.

If the proper data is available one can apply statistical methods and conventional modeling notations like fault trees [66] and event trees [64].

6.1.3.1 Core Security Risk Scenarios in Phase III

Each risk is given a consequence and likelihood estimate as shown in the following table.

Table 6 – Risks with consequence and likelihood estimates

Risks	Asset harmed	Consequence estimate	Likelihood estimate
R1CM) Disclosure of users' personal information	CM1	Major	Sometimes
R1CA) Disclosure of users' personal information	CA1	Major	Sometimes
R2CM) Unauthorized modification of user's personal information	CM1	Major	Seldom
R2CA) Unauthorized modification of user's personal information	CA1	Major	Seldom
R3CM) Unavailability of service due to hackers	CM3	Major	Seldom
R3CA) Unavailability of service due to hackers	CA2	Major	Seldom
R4CM) Unavailability of service due to infrastructure failure	CM3	Moderate	Sometimes
R4CA) Unavailability of service due to infrastructure failure	CA2	Moderate	Sometimes
R5CM) Unavailability of service due to malicious code	CM3	Moderate	Seldom
R5CA) Unavailability of service due to malicious code	CA2	Moderate	Seldom
R6CM) Damage to company reputation	CM2	Moderate	Seldom
R7CM) Reduced user efficiency	CM4	Minor	Sometimes
R7CA) Reduced user efficiency	CA3	Moderate	Sometimes

6.1.4 Phase IV: Evaluating Risks

When evaluating the risks one decides which ones that are most serious. The risks are prioritized according to their gravity and the ones that cannot be tolerated are subject to treatment identification.

In this phase we need to model an overview of the acceptable and unacceptable risks.

6.1.4.1 Core Security Risk Scenarios in Phase IV

Table 7 – Risks with risk values

Risks	**Asset harmed**	**Computed risk value***
R1CM) Disclosure of users' personal information	CM1	High risk
R1CA) Disclosure of users' personal information	CA1	High risk
R2CM) Unauthorized modification of user's personal information	CM1	Medium risk
R2CA) Unauthorized modification of user's personal information	CA1	Medium risk
R3CM) Unavailability of service due to hackers	CM3	Medium risk
R3CA) Unavailability of service due to hackers	CA2	Medium risk
R4CM) Unavailability of service due to infrastructure failure	CM3	Medium risk
R4CA) Unavailability of service due to infrastructure failure	CA2	Medium risk
R5CM) Unavailability of service due to malicious code	CM3	Low risk
R5CA) Unavailability of service due to malicious code	CA2	Low risk
R6CM) Damage to company reputation	CM2	Low risk
R7CM) Reduced user efficiency	CM4	Low risk
R7CA) Reduced user efficiency	CA3	Medium risk

*The risk value is set using the risk matrix in Figure 46.

Comparing the risk values with the risk tolerance levels from phase 1 (Table 5) gives the following risk evaluation (shown in Figure 47):

- Company management: R1CM and R2CM are higher than the accepted risk level.
- Central authorities: R1CA and R2CA are higher than the accepted risk level.

	Likelihood		
Consequence	Seldom	Sometimes	Often
Minor		R7CM	
Moderate	R5CM, R5CA, R6CM	R4CM, R4CA, R7CA	
Major	***R2CM, R2CA***, R3CM, R3CA	***R1CM, R1CA***	

Figure 47 – Risks placed in the risk evaluation matrix

6.1.5 Phase V: Identifying Treatments

The purpose of this phase is to decide which risks that need treatments, i.e. are too serious to be left unattended and what kind of treatments. In this phase it is useful to have an overview diagram of the risks (including the vulnerabilities, threats and unwanted incidents involved) with risk values as input and extend it with various treatments options.

One uses the risk value to decide which risks that needs to be treated. The client (the person or organization that initiated the analysis in the beginning, often identical to the stakeholder, but not always) decides the risk tolerance level.

6.1.5.1 Core Security Risk Scenarios in Phase V

Treatment options:

- **TO1: Upgrade to more robust infrastructure solution** that have lower failure rate.
- **TO2: Install redundant system** that will take over in case of infrastructure failure or attack.
- **TO3: Install improved firewall** that will make it more difficult for a hacker to find vulnerabilities.
- **TO4: Install intrusion detection system** that will detect the attack rapidly and make it possibly to switch to manual routines.
- **TO5**: Remove the possibility employees have to **access other users' personal information**.
- **TO6: Remove the unsecured WLAN**.
- **TO7: Remove the "remember me"-function** for employees.
- **TO8: Involve users** in the development of an improved system.
- **TO9**: **Implement logging facilities**.
- **TO10**: The user and the service provider should share the responsibility for modification of data due to user errors in combination with slow response from the service. This should be stated in a **legal contract**.

Treatment effects: Estimated effects on likelihood and/or consequence are shown in the table below. These are only for example purposes and have not been estimated on basis of any expert judgments or other sources of information.

Table 8 – Treatment effects

	R1CM	R1CA	R2CM	R2CA
TO1	-	-	Reduce likelihood	Reduce likelihood
TO2	-	-	Reduce consequence	Reduce consequence
TO3	Reduce likelihood	Reduce likelihood	Reduce likelihood	Reduce likelihood
TO4	Reduce consequence	Reduce consequence	Reduce consequence Reduce likelihood	Reduce consequence Reduce likelihood
TO5	Reduce likelihood	Reduce likelihood	Reduce likelihood	Reduce likelihood
TO6	Reduce likelihood	Reduce likelihood	Reduce likelihood	Reduce likelihood
TO7	Reduce likelihood	Reduce likelihood	Reduce consequence	Reduce consequence
TO8	Reduce likelihood	Reduce likelihood	Reduce likelihood	Reduce likelihood
TO9	Reduce likelihood	Reduce likelihood	Reduce consequence Reduce likelihood	Reduce consequence Reduce likelihood
TO10	-	-	Reduce consequence	Reduce consequence

The individual treatment options' effects on risk values are shown in the table below (unc hanged risk values means that the single treatment is not sufficient to reduce the risk value, "-" means the treatment is not applied for the risk). Since these estimates are included in the core security risk scenarios with the purpose of showing how they are dealt with in the models, they are only example estimates without a thorough rationale.

Table 9 – Treatment effects on risk values

	R1CM	R1CA	R2CM	R2CA
TO1	-	-	medium → low	medium → low
TO2	-	-	medium → low	medium → low
TO3	high → high	high → high	medium → medium	medium → medium
TO4	high → high	high → high	medium → medium	medium → medium
TO5	high → high	high → high	medium → medium	medium → medium
TO6	high → low	high → low	medium → low	medium → low
TO7	high → high	high → high	medium → medium	medium → medium

TO8	high → medium	high → medium	medium → medium	medium → medium
TO9	high → medium	high → medium	medium → low	medium → low
TO10	-	-	medium → medium	medium → medium

The final treatments selected for implementation are typically decided upon after a cost-benefit assessment of each of the treatment options. However, such an assessment is outside the scope of this core security risk scenario example.

6.2 Quality Requirements

Modeling is a commonly used technique within system development and there exist a number of modeling languages which are more or less suitable for this purpose. Graphical models are often used to ease the understanding of complex systems. In security analyses one deals with critical systems that may contain confidential data, provide critical services or have a high demand for availability. A correct understanding of the system and its risks is highly important, and this is where graphical models are found useful. The participants in the analysis normally have much competence on different parts of the target and may therefore view it differently. This may cause problems in understanding each other or problems in agreeing on the appropriate level of precision or scope of the analysis. By graphically modeling the target with its threats, assets and risks one can easily reduce the number of misunderstandings since this clarifies these aspects. The challenge is to find a modeling language that people understand, preferably suitable for use in a computerized modeling tool.

The quality of a modeling language depends on several factors, a language which is excellent for one task may be inappropriate for a different task. Through our experience with modeling in industrial security risk analyses, we have developed a set of detailed requirements to this kind of modeling language. To structure these requirements we have implemented them in the quality framework for modeling languages called SEQUAL developed by Krogstie, Sindre, Lindland and Sølvberg [100-104]. The framework can be used when selecting between different languages, investigating possible improvement areas for a language, or as the basis requirements to a new modeling language. The framework deals with both the quality of a particular model, and with the quality of modeling languages. In this work we will only use the part related to modeling languages, or what we call the language's "appropriateness factors".

The appropriateness factors of a modeling language are related to the modeling task definition, i.e. the goal of the modeling task (***G***), its domain (***D***), the knowledge of the people involved in the modeling task (***Ks, Km***), the interpretation of the models (***I***), the language that is used (***L***) and the tools (***T***) (illustrated in Figure 48). Figure 48 also shows the graphical model (model externalization, ***M***), but as mentioned above, this work will not go into the quality aspects of a concrete model. In this evaluation, the purpose is to evaluate the UML profile's appropriateness factors for security risk modeling in structured brainstorming sessions.

The six appropriateness factors are:

- Domain appropriateness: to be appropriate for the domain, the language should include all concepts necessary to express anything within the domain it is meant for.

- Participant language knowledge appropriateness: to be appropriate for the participants' language knowledge, the concepts and constructs in the languages should be as close as possible to the participants' understanding of the "real world".
- Knowledge externalizability appropriateness: to be appropriate for the knowledge externalizability the language should be able to express all aspects of the domain that the users are interested in.
- Comprehensibility appropriateness: to have an appropriate comprehensibility the language should be understandable for the users.
- Technical actor interpretation appropriateness: to be considered appropriate for the technical actors the language should have a syntax and semantics that a computerized tool can understand.
- Organizational appropriateness: to be appropriate for the organization the language should fit into existing technology, work processes and modeling methods that are used by the organization.

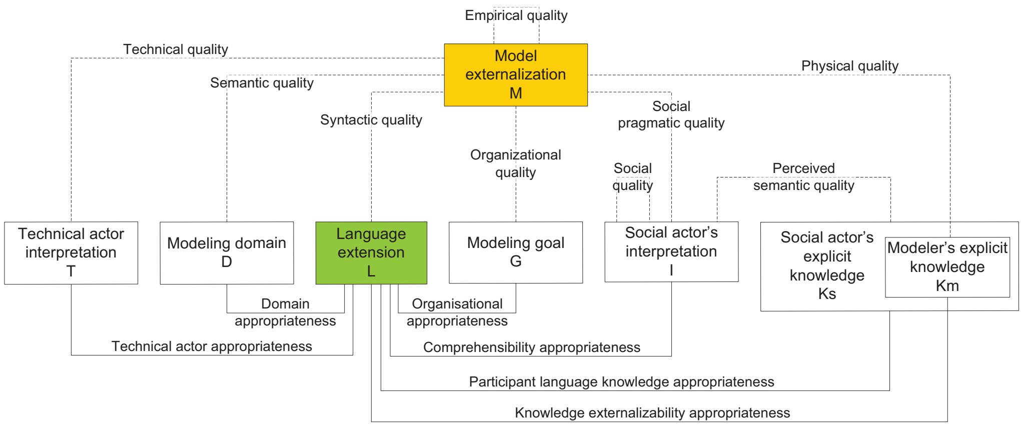

Empirical quality
Technical quality
Model externalization M
Physical quality
Semantic quality
Syntactic quality
Social pragmatic quality
Organizational quality
Social quality
Perceived semantic quality
Technical actor interpretation T
Modeling domain D
Language extension L
Modeling goal G
Social actor's interpretation I
Social actor's explicit knowledge Ks
Modeler's explicit knowledge Km
Domain appropriateness
Organisational appropriateness
Technical actor appropriateness
Comprehensibility appropriateness
Participant language knowledge appropriateness
Knowledge externalizability appropriateness

Figure 48 – The quality framework

6.3 Adapting SEQUAL to the Security Analysis Setting

Before assessing the appropriateness of a language, it is necessary to define the modeling task for which the modeling language should be used.

Table 10 – SEQUAL aspects

The goal of modeling (***G***)	the goals of the modeling task (normally organizational)
The modeling domain (***D***)	the domain, i.e., the set of all statements which can be stated about the situation at hand.
The relevant explicit knowledge ($\boldsymbol{K_s}$) $\boldsymbol{K_m}$ (a subset of $\boldsymbol{K_s}$)	the relevant explicit knowledge of the set of stakeholders being involved in modeling (the audience). A subset of the audience is those actively involved in modeling, and their knowledge is denoted Km.
The social actor interpretation (***I***)	the social actor interpretation, i.e., the set of all statements which the audience thinks an externalized model consists of.
The model externalization (***M***)	the externalized model, i.e., the set of all statements in someone's model of part of the perceived reality, written in a language.
The language extension (***L***)	the language extension, i.e., the set of all statements that are possible to make according to the graphemes, vocabulary, and syntax of the modeling languages used

The technical actor interpretation (***T***)	the technical actor interpretation, i.e., the statements in the model as 'interpreted' by different model activators (for example modeling tools).

In order to reflect the security analysis setting, or more specifically the structured brainstorming setting, the definition in Table 10 is translated from its general form into a more specialized form (Table 11). A structured brainstorming is a methodical "walk-through" of a target system with purpose of identifying as much information about the target as possible for each step. The brainstorming is a group exercise involving highly skilled people with competence of relevant parts of the target of analysis.

Table 11 – Modeling task definition in a security analysis setting

G: to reduce the effort needed to identify and understand the overall risk picture for the system assessed in a structured brainstorming, and thereby contribute to increased control of the risks for the organization
D: the security risks towards the system assessed.
K_s: the knowledge of the people who contributes to the models in the structured brainstorming. This will typically be experts on various parts of the system assessed, in addition to the analysis leader and the secretary. ***K_m***: the knowledge of the security analysis leader (or analysis secretary) which gathers information from the participants and models it during the structured brainstorming.
I: the interpretation of models seen from the participant's point of view (e.g. system owner's, developer's and user's).
M: the model of the system's risks.
L: all the statements that is possible to make in the UML profile according to its definition.
T: the statements in the model that can be interpreted by other modeling tools or specialized security analysis tools.

In the following we discuss our requirements to a security risk modeling language for each of the appropriateness factor categories. Throughout the evaluation we will refer to the modeling task definition using the letters from Table 11 (modeling domain = *D*, modeling goal = *G* etc.). Within each category we have included a number of extra requirements based on experiences with the CORAS security analysis method in industrial field trials.

The requirements are not assigned explicit weights to show their importance since they all represent desired language features, instead we use the terms "must" and "should" to indicate their importance (requirements described with "must" are more required than those explained with "should"). The requirements are numbered sequential and "S-x" means that the requirement originates directly from SEQUAL, while "C-x" means that the requirement comes from user-experiences with the CORAS method.

6.3.1 Domain Appropriateness

The domain appropriateness of a modeling language relates to how much of a domain one is capable of modeling (defined in the language's internal representation) and how it is expressed in a diagram (defined in the language's external representation).

In information modeling there are several types of modeling perspectives to choose between. According to [104] there are 7 general modeling perspectives: *structural, functional, behavioral, rule-oriented, object-oriented, language-action-oriented* [168] and *role & actor-oriented modeling perspective*. Security risk modeling has many similarities with information modeling. Information modeling often describes the behavior of a workflow or process, whereas security risk modeling deals with describing the "workflow" of a threat that initiates an unwanted incident. For security risk modeling we have identified the need of describing *how* incidents can happen, *who is* initiating them and *what* will be affected. There is also a need for showing more static relations that can specify dependencies between assets, treatments, risks and more. Based on this we require our modeling language to provide the following modeling perspectives:

Table 12 – Domain appropriateness: modeling perspectives

Requirements	Explanation
C-1 Structural modeling	***L*** must provide a structural modeling perspective. Structural modeling is used to illustrate for example how assets relate to each other, how one can group similar risks or treatments, how vulnerabilities relate to assets and more.
C-2 Behavioral modeling	***L*** must provide a behavioral modeling perspective. The behavior modeling perspective shows how a threat can initiate one or more unwanted incidents to harm assets, or how various treatments can be applied to risks and more.

SEQUAL distinguishes between *symbol* and *concept*: a concept is a phenomena/something one wants to express, while a symbol is the graphical notation used to model the concept.

In terms of requirements to domain appropriateness, ***L*** must support the concepts and relations according to the revised CORAS conceptual model for security analysis terminology (specified in *Chapter 5*). By enforcing this requirement we will insure that ***L*** covers all terms and relations that we find relevant for security analysis. To fulfill this requirement ***L*** must be able to express what is described in the explanation of the conceptual model, i.e. which concepts relate to each other and in what way.

Table 13 – Domain appropriateness: concepts

Requirements	Explanation
C-3 Asset	***L*** must include the concept "asset". *An **asset** is something to which an organization directly assigns value and, hence, for which the organization requires protection* [55]. The concept is very central in a security analysis. It should be up to the modeler to decide how much details the specification of an asset should include. Often an asset can be affected by other assets, for example a company's reputation is affected by the quality of the product they deliver, the service they provide, the employee's satisfaction etc. The modeler should have the option to create groups of assets that are similar or affect each other.
C-4 Vulnerability	***L*** must include the concept "vulnerability". *A **vulnerability** is a weakness with respect to an asset or group of assets that can be exploited by one or more threats* [76]. Vulnerabilities are critical parts of the system and important to establish early in the security analysis. Vulnerability can be composed into composite vulnerabilities according to the chosen level of detail. It would be useful to group similar vulnerabilities since they may be treated in the same way.
C-5 Risk	***L*** must include the concept "risk". *A **risk** is the chance of something happening that will have an impact upon objectives* [6]. Similar risks may be grouped and risk can be composed into composite risks.

C-6 Stakeholder	***L*** must include the concept "stakeholder". ***Stakeholders** are those people and organizations who may affect, be affected by, or perceive themselves to be affected by, a decision or activity* [6]. By identifying a system's stakeholders one gets an overview of for example which people or systems that use it or depend on its functions.
C-7 Threat	***L*** must include the concept "threat". *A **threat** is a potential cause of an (unwanted) incident which may result in harm to a system or organization* [76]. It should be possible to specify whether a threat is human or non-human. Other more specified threats can be specified within these categories, e.g. deliberate threats and non-deliberate (accidental), but we do not find it necessary to define these as separate concepts in ***L***.
C-8 Unwanted incident	***L*** must include the concept "unwanted incident". An **unwanted incident** may result in harm to a system or organization [55]. In [76] called information security incident. An unwanted incident can vary in the level of details and ***L*** should provide the possibility to model both very detailed and more general. An unwanted incident may very well consist of more incidents.
C-9 Treatment	***L*** must include the concept "treatment". *A **treatment** is the selection and implementation of appropriate options for dealing with risk* [55, 76]. It should be possible to specify the treatment strategy. According to [76] treatment strategies can be categorized into four groups: 1) reduce likelihood of risk, 2) reduce consequence of risk, 3) transfer risk in full or part, 4) avoid risk or 5) retain risk. We do not require ***L*** to support exactly these categories, but there should be an option to specify a treatment in more detail if the user finds it necessary. In our opinion treatment strategies are of two types, either their purpose is to reduce the likelihood of a risk, or reduce its consequence(s). Transferring a risk partly or full can for example mean outsourcing the risky part to someone more qualified (reduce the likelihood of risk) or buying insurance against the risk (reduce the consequence of the risk). The strategy of avoiding a risk is really reducing its likelihood or consequence to zero. Based on this a treatment can either be regarded as *preventive* (i.e. reduce the likelihood) or *repairing* (i.e. reduce the consequence) and these interpretations should be included in the treatment concept in ***L***. The term ***safeguard*** defined as *a practice, procedure or mechanism that reduces risk* [55] are sometimes used instead of treatment. These terms are two names for the same concept. One could possibly argue that *safeguard* is more representative for a protection mechanism against threats that already exist in the system than treatment, which sounds like something applied after an unwanted incident. Establishing existing safeguards is relevant both in context- and treatment identification.
C-10 Likelihood/ frequency/ probability	***L*** must include the concepts "likelihood, frequency, probability". Likelihood, frequency and probability are all measures for variants of "how likely is it that this will happen". ***L*** must support all these measure-types. Likelihood is a qualitative or quantitative description of frequency or probability [6]. Frequency and probability is quantitative measures with a higher degree of precision. Frequency is an exact number of occurrences, while probability is a number between 0-1 where 0 = unlikely and 1 = will happen (in practice one normally says 50% instead of 0.5). Depending on the statistical data available for the security analysis the user should decide which measure to use.

C-11 Consequence	***L*** must include the concept "consequence". A consequence *is the outcome of an event expressed qualitatively or quantitatively, being a loss, injury, disadvantage or gain. There may be a range of possible outcomes associated with an event* [6]. Sometimes consequence is called "Impact" but have the same meaning: *the result of an unwanted incident* [76].
C-12 Threat scenario	***L*** must include the concept "threat scenario". A **threat scenario** is a description of how a threat may lead to an unwanted incident by exploiting vulnerabilities. A threat scenario must be able to express all from very detailed scenarios to vaguer scenario descriptions. A threat scenario can be a chain of threat scenarios, meaning that a threat scenario can lead to one or more scenarios depending on the level of details required. A threat scenario may be given a likelihood estimate. This is used in cases where the likelihood of an unwanted incident cannot be decided precisely but is best estimated from the likelihoods of each threat scenario that may lead to the incident.
C-13 The relations in our conceptual model (Chapter 5)	***L*** must support the explicit relations in the conceptual model, but it must also be possible to specify relations between concepts that are not part of the model (e.g. threat scenario). The symbols used to represent these relations are discussed in the symbol section. The following relations must be supported: 1. A stakeholder can be associated with one or more assets, but an asset can only be associated with one stakeholder 2. An asset is associated with at least one vulnerability, a vulnerability can be associated with more than one asset. 3. A threat must exploit (go via) at least one vulnerability in order to harm an asset. 4. A threat is associated with at least one unwanted incident via one or more threat scenarios. An unwanted incident is associated with at least one threat. 5. A threat is associated with at least one asset via threat scenario(s) and unwanted incident. An asset is associated with at least one threat via unwanted incident and threat scenario(s). 6. A threat is indirectly associated with a risk through its relation to unwanted incident (not explicitly shown in the model). This means that all threats are associated with at least one risk and every risk is associated with at least one threat. 7. An unwanted incident is indirectly associated with at least one asset through risk. In special cases one may experience that an asset is not related to any unwanted incidents, meaning that it has not been identified any risks for this asset. 8. A treatment is always related to one or more risks. 9. A treatment is indirectly related to an unwanted incident, threat, or vulnerability (or a combination of these) via risk (not explicitly shown in the model). Still it should be possible to model a treatment towards one of these concepts. 10. A risk is always associated with one asset, while an asset may be affected by many risks. 11. A risk always includes estimates of likelihood and consequence. 12. A risk always includes one unwanted incident, while an unwanted incident may participate in several risks, meaning that there must be a relation between risk and unwanted incident.

C-14 And/or operators	There must exist and/or operators in ***L***. In practice an unwanted incident often requires two or more threat scenarios to occur simultaneously before it occurs. To handle these cases we require ***L*** to provide "and" and "or" operators. If the outcomes of two or more events are joined in an and-operator they all are required to initiate a new event. If the operator is an or-operator it is sufficiently that only one event occurs to initiate a new event. Operators will provide additional information about the behavior in the threat scenarios, specifying the alternative ways a threat can behave. Operators will also increase the possibilities of using conventional fault tree techniques to compute the likelihood of risks.
C-15 ***L*** must be independent of the target that is assessed	This requirement means that ***L*** must be useful for describing any type of security critical system. It should not have any concepts that depend on a specific system type or technology. Since the CORAS method can be applied to any security critical target, the modeling language should have the same level of flexibility.
C-16 Region	There must exist a concept similar to region. A region is a logical or physical part of the target that can be used as the link between the risk specific documentation and the target documentation. This is useful for structuring the documentation and helps the reader understand the risk models.

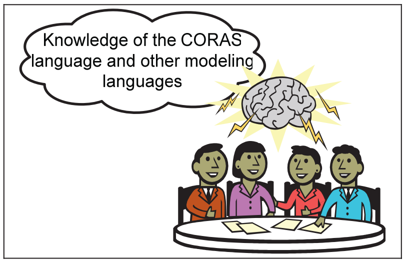
As already mentioned, SEQUAL distinguishes between concepts and symbols. To be a language with high domain appropriateness ***L*** must provide symbols for the set of the concepts identified in the discussion above:

Table 14 – Domain appropriateness: symbols

Requirements	**Explanation**
C-17 Asset	There must be a symbol for asset.
C-18 Vulnerability	There must be a symbol for vulnerability.
C-19 Aggregated vulnerability	There should be symbols for aggregated vulnerabilities
C-20 Risk	There must be a risk symbol that can show a risk value.
C-21 Risk theme	There should be symbols for risk themes (similar types of risks).
C-22 Aggregated risk	There should be symbols for aggregated risk.
C-23 Stakeholder	There must be a stakeholder symbol.
C-24 Threat	There must be a general threat symbol.
C-25 Human threat	***L*** should provide symbols for human- and non-human threat. Any other type of threat can be specified within one of these categories and extra symbols can be added if desired.
C-26 Non-human threat	
C-27 Threat scenario	There must be a symbol for threat scenario. The symbol should have the option to illustrate its estimated likelihood.
C-28 Unwanted incident	There must be a symbol for unwanted incident.
C-29 Treatment	There must be a symbol for treatment.
C-30 Treatment type	It should be possible to specify the type of treatment (e.g. preventive or repairing)
C-31 Likelihood, frequency, probability	There should be a symbol for likelihood/frequency/probability that could be used if particular attention to this concept is required.
C-32 Consequence	There should be a symbol for consequence that could be used if particular attention to this concept is required.

C-33 Association type	We must have both directed (arrow) and undirected associations (line only). A concept may initiate another, a concept may affect another, or one concept has some relation to another without specifying direction. It should also be possible to assign these relations with a description. We need: a) a undirected association between two symbols (a line with no arrow ends): b) a directed association pointing from X to Y, with the option to annotate it with a description (e.g. "initiates", "affects", "reduces frequency" etc.):
C-34 Operators	a) There must be an and-symbol for fault tree modeling where two or more relations are joined together and initiate a new event(s) in the meaning "if both A and B, then C". b) There must be an or-symbol for fault tree modeling where two or more relations are joined and initiate to a new event(s) in the meaning "if either A or B, then C".
C-35 Region	There should be a symbol for logical or physical regions that can be used to specify target.

6.3.2 Participant Language Knowledge Appropriateness


Knowledge of the CORAS language and other modeling languages

Participant language knowledge appropriateness is related to ***L*** and ***Ks***. ***Ks*** is the participant's knowledge about ***D*** and ***L*** (including all other modeling languages). ***M*** (the external representation) is made on the basis of ***Ks***. In this setting "participants" means those who are involved in modeling, but *without* doing the actual modeling (for example expert participants in the brainstorming). In order for ***L*** to be appropriate for the participant's language knowledge, ***L***'s internal representation should not conflict with the participants understanding of ***D***, and ***M*** should relate to ***D*** in an intuitive manner (this is also relevant for comprehensibility appropriateness).

With respect to participant knowledge appropriateness we have identified the following requirements:

Table 15 – Participant language knowledge appropriateness

Requirements	**Explanation**
S-1 ***L***'s external representation must imitate the real world	If the interpretation of ***M*** corresponds to the participants' understanding of the real world (***Ks)*** there will be less confusion and misunderstandings. An example is: "an unwanted incident affects three assets simultaneously" this can be modeled several ways but to avoid conflicts with the modelers understanding of the real world it should clearly illustrate the "simultaneously" aspect.
S-2 The symbols used in ***L*** must be based on most common interpretation of concept-symbol	This means that the symbols used in ***L*** represent ***D*** better or are more intuitive than other symbols that could have been used.
C-36 ***L*** must be understandable for people unfamiliar with modeling and without specific training	***M*** made with ***L*** should be easy to understand (read), even for people without modeling experience. To find how much effort is needed to learn ***L***, one should base oneself upon experiences with similar modeling languages like UML use cases, activity diagrams or flow charts.

6.3.3 Knowledge Externalizability Appropriateness

This appropriateness factor focuses on how ***Km*** (relevant modeler knowledge) may be articulated in ***L*** (the modeling language). Is it possible for the modeler to express his or her knowledge about for instance the target threats with ***L***? To achieve high score on knowledge externalizability appropriateness the modeler (the one who creates the actual model ***M***) should be able to use ***Km*** to learn ***L*** faster, be able to express all ***Km*** with ***L***, and design better ***M***.

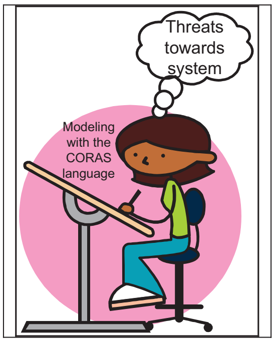


Table 16 – Knowledge externalizability appropriateness

Requirements	Explanation
S-3 ***L*** must help externalize tacit knowledge	This means that ***L*** should use well-known metaphors/analogies to explain/model more complicated relations to lower the effort needed to understand the models
S-4 It must be easy to model as part of actual work	The modeling should not require extensive training and heavy tool-support. The models should be easy and quick to create as part of the security analysis, and update during maintenance of the system. This means that modeling should not only be done before (planning) or after (post-hoc rationalization), but support interactive modeling.
C-37 It must be possible to model fault trees with ***L***	Fault tree analysis (FTA) (Sect. 3.4.4) is a well known risk analysis technique and it must be possible to draw fault tree diagrams using ***L.*** A conventional fault tree diagram has a restricted expressiveness which means that the notation is a subset of ***L***.
C-38 It should be possible to model event trees with ***L***	Event tree analysis (ETA) (Sect. 3.4.5) is a well known risk analysis technique and it should be possible to draw event tree diagrams using L (see **Figure 9 – Event** tree example).

6.3.4 Comprehensibility Appropriateness

Comprehensibility appropriateness relates ***L*** (the modeling language) and ***I*** (the social actor's interpretation) and focuses on how easy it is to interpret ***M*** (models) made with ***L***. There are requirements both to the internal and external representation of ***L***.

So these are the 3 threats towards the target of analysis
I see 5 risks against the target of analysis
CORAS language diagram

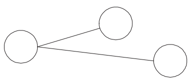
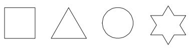
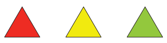
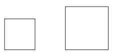
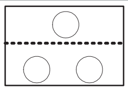
Table 17 – Comprehensibility appropriateness: internal representation

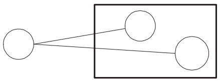
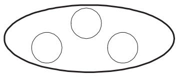
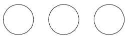
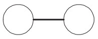
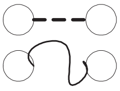
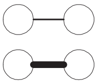
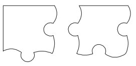
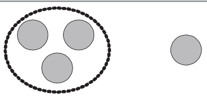
Requirements	Explanation
C-39 ***L*** must build on most-common ***I*** of risk specific concepts	If the interpretation of the terms corresponds to the participants' intuitive understanding of security analysis specific terms, there will be less confusion and misunderstandings.

S-5	The concepts in ***L*** must be general rather than specialized	The more delimited *D* is the more specialized concepts can be used, but the use of specialization should correspond to the specialization level in the security analysis, i.e. if the analysis has a high-level scope, *M* should also be high-level. The security analysis domain is in itself specialized, but within this domain we want *L* to be general (i.e. independent of target type, analysis scope and technology).
S-6	The concepts in ***L*** must be composable	*L* should make it possible to group related statements in a natural way. This means that we use compose in the sense "aggregate", meaning a form for grouping, e.g. vulnerabilities may be aggregated into a group of vulnerabilities etc.
S-7	***L*** must be flexible in precision	*L* must be able to express both precise knowledge and more vague information, meaning it must include both precise and vague constructs. Initially in a security analysis one often deals with incomplete or incorrect information about threats or assets, but they still must be modeled to provide an overview (even though they may be changed as more information is gathered). In some cases the natural language description of the modeling element will decide the model's precision level.
S-8	If the number of concepts has to be large, the phenomena must be organized hierarchically	In *D* there exists many types of threats, vulnerabilities, assets and so on, but it should be possible to see these concepts as a hierarchy where the top concepts are more general than the ones on the lower levels.
S-9	The concepts of ***L*** must be easily distinguishable from each other.	By using easily distinguishable concepts the models will be easier to understand.
S-10	The use of concepts must be uniform throughout the whole set of statements that can be expressed within ***L***	Meaning that a concept like 'risk' is to mean the same thing every time it is used
S-11	***L*** must be flexible in the level of detail	This is especially relevant early in the security analysis, for example it must be possible to specify some threat scenarios very detailed, while others will be more high level. In architectural descriptions one has different viewpoints of the architecture that shows specific parts or aspects of the architecture. The same idea can be used in security risk modeling, some diagrams may show the overall risk picture, some are dedicated to show details about threats etc. Modeling tools often have functions for showing/hiding diagram details, and the idea can also be used even without computerized tool. E.g. *L* should be able to provide an overall risk picture for a system, describing risks and threats that can harm the various assets in the system without details about "how" (threat scenarios + unwanted incidents)

Table 18 – Comprehension appropriateness: external representation

Requirements	Explanation
C-40 It should be possible to associate the overall threat picture to target descriptions	It must be possible to specify a general target with boundaries, its threats, assets and vulnerabilities in an abstract manner to facilitate integration with target documentation. This will typically be useful in the preliminary security analysis where it can help guiding the scope of the analysis. The target specification should initially contain as little as possible technology related aspects (only a logical or physical region can sometimes be sufficient), but be extended with more details as the analysis progresses. If the threat diagrams can be mapped to target descriptions they will increase their value and not be a "stand-alone" documentation type.
S-12 ***L*** must contain constructs that can represent the intention of the underlying conceptual model	This means that there should be possible to externally represent the concepts and relations in the underlying conceptual model. This requirement is covered by the requirements in the domain appropriateness section.
S-13 Symbol discrimination in ***L*** must be easy.	To avoid confusion, misunderstandings, irritation and frustration the symbols must be easy to distinguish.
S-14 It must be easy to distinguish which of the symbols in ***L*** any graphical mark in ***M*** is part of	These are means to achieve what Goodman [49] terms syntactic disjointness: • The use of symbols should be uniform, i.e. a symbol should not represent one concept in one context and another one in a different context. Neither should different symbols be used for the same concept in different contexts. • One should strive for symbolic simplicity. • One should use a uniform writing system for concepts at a comparable level. All symbols (at least within each sub-language) should be within the same writing system (e.g. non-phonological such as pictographic, ideographic, logographic, or phonological such as alphabetic). Obviously a modeling language contains both graphics and text, but then the text is labels to concepts, not 1.order concepts. • If using colors to mark semantics, one should not use more than 5-6 different colors in a given view [142]. The color of the label-text will depend on the color of the symbol. One also should have in mind how a model with colored symbols will look when printed (i.e. if the semantic differentiation meant to be carried by the coloring is retained)

S-15 The use of emphasis in ***L*** must be in accordance with the relative importance of the statements in the given ***M***	• Size (the big is more easily noticed than the small) • Solidity (e.g. bold letters vs. ordinary letters, full lines vs. dotted lines, thick lines vs. thin lines, filled boxes vs. non-filled boxes) • Difference from ordinary pattern (e.g. slanted letters, a rare symbol will attract attention among a large number of ordinary ones) • Foreground/background differences (if the background is white, things will be easier noticed the darker they are) • Color (red attracts the eye more than other colors). • Change (blinking or moving symbols attract attention) • Pictures vs. text (pictures usually having a much higher perceptibility, information conveyed in pictures will be emphasized at the cost of information conveyed textually) • Position (Westerners tend to read it from left to right) • Connectivity (objects able to connect to many others (having a high degree) will attract attention compared to objects making few connections)
S-16 Composition of symbols should be made in an aesthetically pleasing way	These are means for achieving an aesthetically pleasing ***M*** (according to [154]): • Angles between edges should not be too small • Minimize the area occupied by the drawing • Balance the diagram with respect to the axis • Minimize the number of bends along edges • Maximize the number of faces drawn as convex polygons • Minimize the number of crossings between edges • Place nodes with high degree in the centre of the drawing • Minimize differences among nodes' dimensions • Minimize the global length of edges • Minimize the length of the longest edge • Have symmetry of sons in hierarchies • Have uniform density of nodes in the drawing • Have verticality of hierarchical structures
S-17 ***L*** should not have empty symbols.	Symbols that are not related to a specific concept should not be used.
S-18 ***M*** must adhere to the most common principles from gestalt psychology	This does not mean that ***L*** should incorporate all the principles listed below, but rather avoid direct violation of them. Within the area of gestalt psychology, a number of principles for how to convey meaning through perceptual means is provided [160]:
	• A closed contour in a node-link diagram generally represents a concept of some kind. 
	• The shape of a closed contour is frequently used to represent a concept type. 
	• The color of an enclosed region represent a concept type 
	• The size of an enclosed region can be used to represent the magnitude of a concept. 
	• Lines that partition a region within a closed contour can delineate subparts of a concept 

	• Closed-contour regions may be aggregated by overlapping them. The result is readily seen as a composite concept 
	• A number of closed-contour regions within a larger closed contour can represent conceptual containment 
	• Placing closed contours spatially in an ordered sequence can represent conceptual ordering of some kind 
	• A linking line between concepts represents some kind of relationship between them 
	• A lined linking closed contours can have different colors, or other graphical qualities such as waviness, and this effectively represents an attribute or type of relationship 
	• The thickness of a connecting line can be used to represent the magnitude of a relationship (a scalar attribute) 
	• A contour can be shaped with tabs and sockets that can indicate which components have particular relationships 
	• Proximity of components can represent groups 
S-19 The most common modeling tasks should be as efficient as possible	Meaning that the most common modeling tasks should take less effort to model than more unusual tasks. "Common" in a CORAS setting must be interpreted as the most high-level modeling tasks, or the minimal modeling effort required by the method. The modeling test case in 6.1 provides an example of the minimum amount of information that needs to be modeled using the CORAS method.
C-41 There must be a reasonable number of diagram types	***L*** must not include numerous different diagram types, ideally it should concentrate on one main diagram type with the possibility of specify details of this diagram in separate diagrams.
C-42 ***L*** must have a precise semantics	To ensure that the language is understood in the same manner by all readers, *L* should have a precise textual semantics.

6.3.5 Technical Actor Interpretation Appropriateness

Technical actor interpretation appropriateness relates to how well ***L*** can be used in computerized tools. With respect to this, we have identified the following requirements:

Table 19 – Technical actor interpretation appropriateness

Requirements	**Explanation**
S-20 ***L*** must have a formal syntax.	Formal syntax defines what a modeler is permitted and prohibited from drawing. Having a formal syntax insures that ***M*** made by different modelers have consistent notation.

S-21 ***L*** should have a formal semantics.	A formal semantics defines what the elements in the syntax means, described in a mathematical way. Having a formal semantics will make ***T*** consistent in different modeling tools, making automatic reasoning like consistency checks, translation of models to tables (and vice versa) and more.

6.3.6 Organizational Appropriateness

According to SEQUAL, this section should be based upon requirements from a specific organization that evaluates a language. In this report we do not have one particular organization in mind, but we present some general requirements drawn from our experience with industrial risk analyses. With respect to organizational appropriateness we have identified the following requirements:

Table 20 – Organizational appropriateness

Requirements	**Explanation**
S-22 ***L*** must contribute to reach ***G***	*M* must contribute to *G,* meaning it must reduce the effort needed to identify and understand the overall risk picture for the system assessed in a security analysis, and thereby contribute to increased control of the risks for the organization
C-43 ***M*** must ease the explanation of risks	*M* must ease the explanation of security risks related to the system assessed
C-44 ***L*** must be usable without investing in expensive software	There should be a version of *L* that can be used in general drawing or modeling tool (i.e. a version providing the symbols only, a plug-in or *L* must come with its own, free modeling environment). The simple version must not require the organization to purchase a new and different modeling tool if it already has one.

7 Evaluation of the UML Profile

This section summarizes the results evaluating the UML profile using the quality framework in *Chapter 6*. We first summarizes the findings regarding modeling the core security risk scenarios, and then the results from evaluating against the SEQUAL-based quality requirements. For the full version of the evaluation we refer to the technical report [57].

7.1 Modeling the Core Security Risk Scenarios (summary)

In order to evaluate a security risk modeling language we needed a benchmarking case representing core security risk scenarios to test the notation against. Through experience from several major security analyses in the SECURIS project we have gathered typical security analysis information into a complete set of scenarios covering all the phases in a security analysis. In the following sections we summarizes whether the UML profile was able to model the information.

Phase 1 - Context establishment:
It was unclear how to model the he distinction between direct and indirect assets within the asset diagram. This means that one cannot specify how damage to one asset may cause damage to other assets. However, the UML profile provides an option to model relationships between concepts like assets by using other UML notations, e.g. class diagrams. The standard does not provide any examples of this, but it means that one can create a hierarchy of assets and show the relationships between them. The UML profile does not let you model the risk acceptance level set by each client for each asset.

Phase 2 - Risk identification
It was no obvious way of modeling the indirect assets in threat and unwanted incident diagrams. If modeled as ordinary direct assets, we would loose the extra information about their status as indirect. If left out of the diagrams they have to be remembered or dealt with in some other way. Modeling this fairly small example of security analysis information resulted in five diagrams that are partly overlapping. To follow the path from the threat "Hacker" to the asset "CM2-Company reputation", it is necessary to look at as many as three diagrams.

Phase 3 - Risk estimation & Phase 4 - Risk evaluation
All the information about the risks and their consequence and likelihood estimates was modeled using the UML profile's risk diagrams. The thirteen risks resulted in as many as 7 figures. Also in these diagrams, information that already has been modeled is repeated (vulnerabilities). To find out which threats that may cause a risk one has trace the path backwards from unwanted incident via each of its threat scenarios to find the initiating threats. This makes it difficult to get a complete overview of the risk picture. The UML profile also lacks the option for differentiating between acceptable and non-acceptable risks.

Phase 5 - Treatment identification
On the positive side, the UML profile was able to express all the information in this phase, but on the negative side it was necessary to use 14 figures (approx. 9 pages) for the

information that is presented on 1,5 A4 page in *Chapter 6*. If we were to complete the diagrams (i.e. include R2CA) it would have required another five figures.

7.2 Evaluation of the Language Quality (summary)

The table below summarizes the average scores for each appropriateness factor discussed in *Chapter 6*. The scores provides an *indication* of weak and strong aspects of the UML profile, but since the level of details in the requirements can vary and no weighting has been included to adjust the score for this, we only use this as an indication of how well the language meets our requirements.

Table 21 – Evaluation scores

Category	**Must-requirements**			**Should-requirements**		
	Score	**# req.**	**Avg.**	**Score**	**# req.**	**Avg.:**
Domain appropriateness:						
• modeling perspectives	5	2	2,5		-	-
• concepts	34,7	14	2,5	4	-	-
• symbols	21,5	10	2		3*	1,3
Participant language knowledge appropriateness	5	3	1,7		-	-
Knowledge externalizability appropriateness	4	3	1,3	0	1	0
Comprehensibility appropriateness:						
• internal representation	18	7*	2,6		-	-
• external representation	8	6*	1,3	3	2	1,5
Technical actor interpretation appropriateness	3	1	3	0	1	0
Organizational appropriateness	5	3	1,7		-	-

* one or more requirements were omitted, see Sect. 6.7.1 in [57]

As we can se from Table 21, the UML profile has both weak and strong aspects.

The knowledge externalizability appropriateness receives a low score due to the illogical representation of the threats' paths via threat scenarios to the unwanted incidents where they cause harm to assets. The natural way of describing this is a logical sequence of events starting with the threat that initiates a threat scenario which leads to an unwanted incident that harms an asset. This idea is part of the language, but unfortunately the external representation fails to show its logical order. The UML profile makes use of the UML construct "include" which intend to show how a use case is included in another use case, but at first glance they look like they are on the same "level". In the unwanted incident diagram (Figure 20) in Sect.3.5.1, the threat scenario is included in the incident scenario "Denial-of-service", but for an untrained reader they seem like two equal entities, a clear violation of common gestalt principles. The language also fails to show which vulnerabilities a threat may exploit since these are only modeled as properties of the asset. The fact that more than one threat potentially can harm the same asset using different vulnerabilities is not possible to model.

Often a risk picture includes several dependencies, i.e. events that simultaneously may lead to an unwanted incident. This is traditionally modeled using fault trees, but the lack of AND/OR-operators in the UML profile makes fault tree modeling impossible.

Comprehensibility appropriateness of the external representation receives a low score, mainly related to the lack of adherence to common gestalt principles. The use of colors and size are not well considered and the include arrow in threat diagrams can easily be misunderstood. When it comes to judging the modeling effort required to make diagrams as a natural part of the analysis process there are differences between the various diagram types. While threat and unwanted incident diagrams are fairly straightforward and does not take much effort, the rest of the diagrams are unnecessarily time- and space consuming from a usability perspective. To ensure a common understanding of the diagrams, a precise mapping from diagrams to text should be used. This mapping is not defined for the UML profile.

The rather low score for *participant language knowledge appropriateness* is caused by the symbols (icons) used. The UML profile symbols we have evaluated are based on standard UML use case symbols, but they are neither uniform, intuitive nor well designed and give an amateurish impression. Since the symbols are not finalized and just sketches of possible representations, the low score for this factor should not be emphasized as much as the other appropriateness factors of the language.

The UML profiles *organizational appropriateness* could have been better. It will probably provide a consistent and complete way of documenting the analysis information, but used as defined in [124] would not help reducing the effort needed to understand the organization's overall risk picture. Creating and understanding the models requires thorough knowledge of the target and the modeling notation itself and is not suitable for bridging the gap between system modelers and more ordinary system users. The ideas behind the various diagrams are on the other hand good, but they need to become simpler and more manageable in terms of complexity.

7.3 Conclusion

As we summarize in Sect. 7.1 it was possible to model almost all the information in the core security risk scenarios with the UML profile. However, being able to express the core security risk scenarios is not sufficient. The models should also present the information in the core security scenarios in a better way than the textual description alone. The diagrams should be understandable and manageable in terms of complexity, and provide a good overview of the information. With respect to this, many of the diagram types are not suitable for presentation of security risk analysis information. They are often characterized by duplication of information, and information that is spread out over several diagrams which makes it difficult to get an overview. The diagrams tend to be extremely space consuming (particularly the treatment effect and risk evaluation diagrams), and require the modeler to repeatedly model almost identical diagrams.

Also the quality evaluation points at the language's external representation as its main weakness. Its internal representation and underlying concepts related to expressing the domain are appropriate for the security analysis setting. Also its technical actor appropriateness receives a high score. The language should be a tool that helps visualizing, explaining and documenting the security risk analysis and this highly depends on its external representation. Unfortunately, some of the weaknesses are related to the underlying UML notation itself. This makes it difficult to redesign the language without violating fundamental UML constructs. There seems to be an inevitable choice to make, either use a non-optimal language with a notation that conforms to UML, or develop a new language without the restrictions from UML. A new language will on the one side not have the support and strength from UML, but on the other side it can have its main focus on understandability and usability. The two languages could be seen as two different versions based on the same underlying basis and with different strengths and application areas.

8 The CORAS Security Risk Modeling Language and Guideline

This section presents the syntax and semantics of the CORAS language and the guideline that explains how to use the CORAS language during the security analysis.

8.1 The Syntax and Semantics

For the CORAS language, we have the following syntaxes and semantics: the conceptual foundation (abstract syntax), the graphical syntax, and the textual syntax with its precise semantics in the form of translation rules into paragraphs in English.

8.1.1 The Conceptual Foundation

The conceptual foundation of the CORAS language can be seen as an abstract syntax where the concepts in the language match this syntax. The development of this foundation is described in *Chapter 5*, but the model (Figure 49) and its definitions are briefly repeated here. The class diagram notation is read as follows: "a threat is associated with at least 1 asset and maximum infinite assets, while an asset may have from 0 to an infinite number of associated threats".

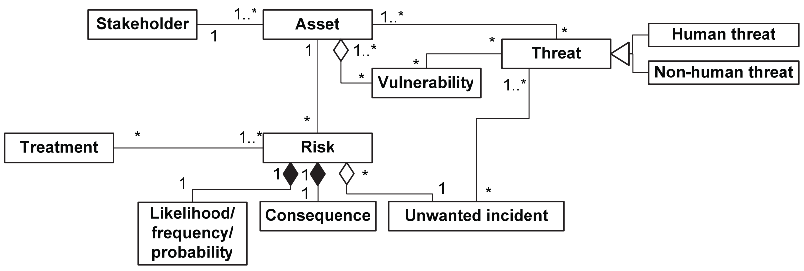


Figure 49 – The conceptual foundation

Figure 49 can be explained as follows: **Stakeholders** are those people and organizations who may affect, be affected by, or perceive themselves to be affected by, a decision or activity regarding the target of analysis [6]. In practice we often use the term "client" for the organization that is paying for the actual analysis, while other stakeholders are called "other interested parties" for instance authorities, interest groups etc. An **asset** is something to which a stakeholder directly assigns value and, hence, for which the stakeholder requires protection [55]. Assets are subject to **vulnerabilities,** which are weaknesses which can be exploited by one or more threats [76]. A **threat** is a potential cause of an unwanted incident [76]. According to [6, 76], threats can be categorized as either human or non-human (environmental). Human threats can also be said to have accidental or deliberate origin. An **unwanted incident** is an event that may harm or reduce the value of assets and is something we want to prevent [55]. A **risk** is the chance of something happening that will have an impact upon objectives (assets) [6]. Our model captures this interpretation by defining a risk to consist of an unwanted incident, a likelihood measure and a consequence. The abstract concept "risk", the more concrete "unwanted incident", and their respective relationships to "asset" require some explanation. In our definition, an unwanted incident that harms more than one asset gives rise to one unique risk for each asset it harms. This enables us to keep the consequences for different stakeholders separate, since an asset is always defined with respect to a single stakeholder. The level of risk is measured by a **risk value** (e.g. low, medium, high or other scales) which is based upon the estimated **likelihood** (a general description of frequency or probability), **frequency** or **probability** for the unwanted incident to happen and its **consequence** in terms of damage to an asset [6]. A **treatment** is the selection and implementation of appropriate options for dealing with risk [55, 76].

8.1.2 The Graphical Syntax

Figure 50 presents the syntactical representation of the conceptual foundation that is used in the CORAS diagrams, i.e. the graphical syntax. The stapled arrow is used for treatments, while the solid arrow is used between threats, vulnerabilities, threat scenarios, unwanted incidents and assets. The logical and/or gates are used when drawing fault trees [66] with the CORAS language.

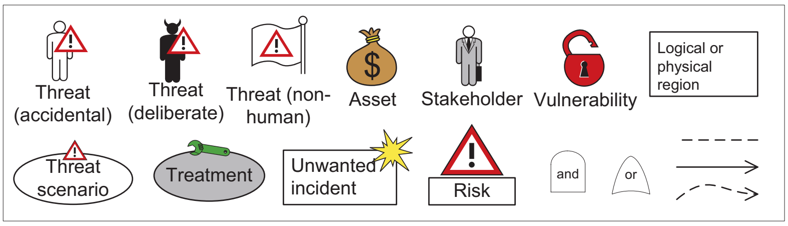


Figure 50 – The graphical syntax

The graphical syntax is further described in the example driven introduction in the modeling guideline in Sect. 8.2.

8.1.3 The Textual Syntax

The textual syntax of the CORAS language [36] (full details in *Appendix D*) has been defined for the elements in Figure 50, except for the AND/OR gates and the region. Ongoing research [21] is investigating the concept of region and this work may be included into the textual syntax in the future. Defining a textual syntax for the logical gates has so far been out of the scope of our work, but it may be a subject for future work.

To characterize the textual syntax for the CORAS language, we have used the ISO standardized Extended BNF notation [81]. The vertical bar _|_ represents options, braces {_} (respectively {_}¯) means an ordered sequence of zero (respectively one) or more repetitions of the enclosed element, and square brackets [_] denotes optional features. The terminal operators are surrounded by quotes: '_'.

To improve readability, we do not resolve all the EBNF definitions completely into terminal operators, but stop at the level of *identifier, linguistic term* and *numerical value*. An identifier is a name or a natural language description of an element, i.e. a finite sequence of upper- and lower-case letters, numbers, spaces and punctuation marks. Its precise definition in EBNF is:

$$\begin{aligned}
\textit{identifier} &= \{\textit{upper-case letter}|\textit{lower-case letter}|\textit{digit}|\text{` '}|\textit{punctuation mark}\}^{-}\ ;\\
\textit{uppercase letter} &= \text{`A'}|\text{`B'}|\text{`C'}|\ldots\\
\textit{lowercase letter} &= \text{`a'}|\text{`b'}|\text{`c'}|\ldots\\
\textit{punctuation mark} &= \text{`.'}|\text{`,'}|\text{`:'}|\ldots\\
&\ldots
\end{aligned}$$

Linguistic terms are identifiers that are members of a totally ordered set, for example likelihoods given as 'high', 'medium' and 'low'. Numerical values are just that: numbers allowing precise calculations of likelihoods, consequences and risk values. In the translation rules, we use $\Rrightarrow$ to denote the translation from graphical to textual syntax. The CORAS diagrams consist of two kinds of elements: vertices and relations between these. The elements may be annotated with additional information. The translation rules for the vertices and annotations of the diagrams are given by the naming conventions in Table 22.

Table 22 – Naming conventions

Vertex	**Instance**
party	p
asset	a
deliberate threat	dt
accidental threat	at
non-human threat	nht
threat scenario	ts
unwanted incident	ui
risk	r
treatment scenario	trs

Annotation	**Instance**
vulnerability	$v = \{v\} = V_1$
vulnerability set	$V_n = \{v_1, \ldots, v_n\}$
likelihood	l
consequence	c
risk value	rv
risk function	rf

When there is more than one instance of a type, we use subscripts, for example dt_1, dt_2 if there are two deliberate threats. The translation from the textual syntax into English is defined by a function [[_]] on textual expressions. We use _ := _ to denote "defined as".

Each diagram type has its own textual syntax described in *Appendix D*. In Figure 51 we provide an example of an asset diagram and its graphical syntax (note that stakeholder in *Appendix D* is named party, and therefore is denoted with a *p* in the diagram).

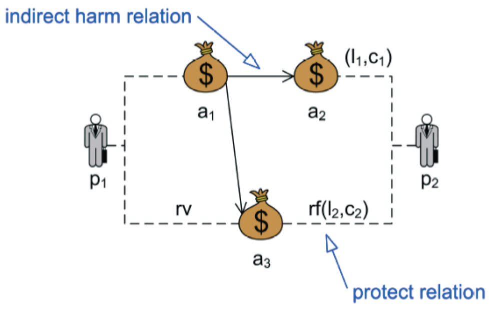


Figure 51 – The graphical syntax of asset diagrams

Below we describe the textual syntax of the asset diagram in Figure 51:

$$\begin{aligned}
\mathit{diagram} &= (\{\mathit{vertex}\}^{-}, \{\mathit{relation}\});\\
\mathit{vertex} &= \mathit{party} \mid \mathit{asset};\\
\mathit{relation} &= \mathit{protect} \mid \mathit{indirect\ harm};\\
\mathit{protect} &= \mathit{party} \overset{[\mathit{risk\ level}]}{\cdots} \mathit{asset};\\
\mathit{indirect\ harm} &= \mathit{asset} \rightarrow \mathit{asset};\\
\mathit{party} &= \mathit{identifier};\\
\mathit{asset} &= \mathit{identifier};\\
\mathit{risk\ level} &= \mathit{risk\ value} \mid \mathit{risk\ function}(\mathit{likelihood}, \mathit{consequence}) \mid\\
&\quad\ (\mathit{likelihood}, \mathit{consequence});\\
\mathit{risk\ value} &= \mathit{linguistic\ term} \mid \mathit{numerical\ value};\\
\mathit{risk\ function} &= \mathit{identifier};\\
\mathit{likelihood} &= \mathit{linguistic\ term} \mid \mathit{numerical\ value};\\
\mathit{consequence} &= \mathit{linguistic\ term} \mid \mathit{numerical\ value};
\end{aligned}$$

8.1.4 The Structured Semantics

The translation of each diagram type from its graphical syntax to its textual syntax is described in full details in *Appendix D*. In this section we provide an example based on the asset diagram above. The translation is done schematically, vertex by vertex and relation by relation. The set of textual expressions we obtain is the translation of the complete diagram.

The translation of a vertex in the textual syntax is the label of the icon representing it in the graphical syntax. For example, the translation of the set of vertices in Figure 51 is the set:

$$\{p_1, p_2, a_1, a_2, a_3\}\,.$$

Note that in this translation, the type of each node is preserved through the naming conventions from Table 22. When the naming convention is not used we need to type each vertex, for example in a type table, or decompose the diagram into three sets: sets of parties, sets of assets and sets of relations.

There are two kinds of relations in the asset diagram in Figure 51:

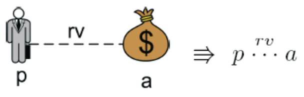


Figure 52 – Protect relation

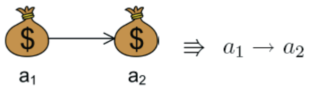


Figure 53 – Indirect harm relation

The translation of the protect relation similar when the relation is annotated with a risk function or a likelihood and consequence pair instead of a risk value. In either case, replace *rv* with the appropriate annotation. If the protect relation is not annotated, then remove *rv* altogether.

Translation from the Textual Syntax to English:
A diagram $D := (\{v_1, \ldots, v_n\}, \{r_1, \ldots, r_m\})$, $n > 0$, $m \geq 0$, is translated by translating each of its vertices and relations. Note that the logical conjunction implicit in the commas of the sets of vertices and relations translates into the period at the end of each English sentence representing a vertex or relation:

$$[\![D]\!] := [\![v_1]\!] \ldots [\![v_n]\!][\![r_1]\!] \ldots [\![r_m]\!]$$

The vertices:

$$\begin{aligned} [\![p]\!] &:= \ \boldsymbol{p} \text{ is a party.} \\ [\![a]\!] &:= \ \boldsymbol{a} \text{ is an asset.} \end{aligned}$$

The protect relation:

$$[\![p \cdots a]\!] := \boldsymbol{p} \text{ wants to protect the value of } \boldsymbol{a}.$$

$$[\![p \overset{rl}{\cdots} a]\!] := \boldsymbol{p} \text{ wants to protect the value of } \boldsymbol{a}\text{, but accepts } [\![rl]\!] \text{ or less.}$$

The second expression is the translation of a protect relation annotated with a maximum acceptable risk level. The *rl* for risk level is replaced by either *rv, rf(l, c)* or *(l, c)* depending on how the risk level is formulated.

The indirect harm relation:

$$[\![a_1 \rightarrow a_2]\!] := \boldsymbol{a}_2 \text{ is harmed indirectly via } \boldsymbol{a}_1.$$

The annotations:

$$\begin{aligned} [\![rv]\!] &:= \text{ risk value } \boldsymbol{rv} \\ [\![rf(l, c)]\!] &:= \text{ risk function } \boldsymbol{rf} \text{ of } [\![(l, c)]\!] \\ [\![(l, c)]\!] &:= [\![l]\!] \text{ and } [\![c]\!] \\ [\![l]\!] &:= \text{ likelihood } \boldsymbol{l} \\ [\![c]\!] &:= \text{ consequence } \boldsymbol{c} \end{aligned}$$

We illustrate the translation of asset diagrams with the following small example:

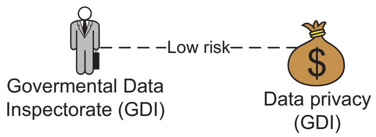


Figure 54 – Example of an asset diagram

A translation of the diagram above consists of two steps: first, we translate the vertices of the diagram:

- **Governmental Data Inspectorate (GDI)** is a party.
- **Data privacy (GDI)** is an asset.

Then we translate the relation using the vertices above:

- **Governmental Data Inspectorate (GDI)** wants to protect the value of **Data privacy (GDI)**, but accepts risk value **Low risk** or less.

8.2 The Guideline (an Example Driven Introduction)

This guideline follows the CORAS security analysis approach from the context establishment, via the risk identification, estimation and evaluation, and ending with the risk treatment. Within these phases there are a total of seven steps, summarized in Table 23:

Table 23 – Steps within the different phases in the CORAS approach

Phases	**Steps**
1. Context establishment:	**Step 1**: The first step involves an introductory meeting. The main item on the agenda for this meeting is to get the representatives of the client to present their overall goals of the analysis and the target they wish to have analyzed. Hence, during the initial step the analysts will gather information based on the client's presentations and discussions. **Step 2**: The second step also involves a separate meeting with representatives of the client. However, this time the analysts will present *their* understanding of what they learned at the first meeting and from studying documentation that has been made available to them by the client. The second step also involves a rough, high-level security analysis. During this analysis the first threats, vulnerabilities, threat scenarios and unwanted incidents are identified. They will be used to help directing and scoping the more detailed analysis still to come. **Step 3**: The third step involves a more refined description of the target to be analyzed, and also all assumptions and other preconditions being made. Step three is terminated once all this documentation has been approved by the client.
2. Risk identification:	**Step 4**: This step is organized as a workshop gathering people with expertise on the target of analysis. The goal is to identify as many potential unwanted incidents as possible, as well as threats, vulnerabilities and threat scenarios.
3. Risk estimation:	**Step 5**: The fifth step is also organized as a workshop. This time with the focus on estimating consequences and likelihood values for each of the identified unwanted incidents.
4. Risk evaluation:	**Step 6**: This step involves giving the client the first overall risk picture. This will typically trigger some adjustments and corrections.
5. Risk treatment:	**Step 7**: The last step is devoted to treatment identification, and one may address cost/benefit issues of the treatments. This step is also best organized as a workshop.

Throughout this section we give an example of how information from a security analysis may be modeled with the CORAS language, and guidelines for how and where in the process the diagrams are made. The guideline also provides instructions for modeling the target of analysis, and it suggests which roles that should be present at the different steps of the analysis. In the end of the guideline we present a selecting of useful recommendations, tips and hints from our experiences with CORAS in industrial field trials.

The target of analysis in our example is a web-based application that communicates confidential information between an insurance company and its customers. The development project is expensive and prestigious to the company, but the governmental data inspectorate is concerned whether the level of privacy is sufficient.

8.2.1 Context Establishment

The purpose of the context establishment is to characterize the target of the analysis and its environment. In our example the web application is represented as a logical region (inspired by [141]) with two independent stakeholders: the company management, which is the "client", and the governmental data inspectorate which represents "other interested parties". The governmental data inspectorate has "Data privacy" as its only asset, while the company management identifies three additional assets: "Application availability", "Application interface usability" and "Company brand & reputation". "Company brand & reputation" is the only "indirect" asset which means that the asset can only be harmed if one of the direct assets is harmed first (Figure 55). The separation between indirect and direct assets implies that in risk identification should be at least one threat diagram for the direct assets, and one for the indirect assets.

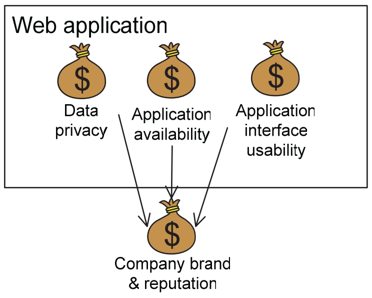


Figure 55 – Asset diagram: indirect and direct assets

The company management and the governmental data inspectorate value the assets differently. The company management wants to protect all assets, while the governmental data inspectorate is mainly interested in "Data privacy". The difference is described in the asset diagram in Figure 56, showing which assets each of the stakeholders are interested in and the risk value they are willingly to accept. If one has specified more detailed risk acceptance levels like maximum acceptable likelihood- and consequence values, they may be used instead of the risk values.

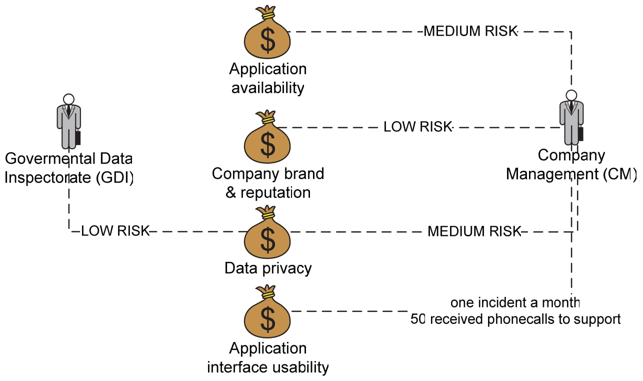

MEDIUM RISK
Application availability
LOW RISK
Govermental Data Inspectorate (GDI)
Company brand & reputation
Company Management (CM)
LOW RISK
MEDIUM RISK
Data privacy
one incident a month
50 received phonecalls to support
Application interface usability

Figure 56 – Asset diagram: risk acceptance values

Since the two stakeholders value "Data privacy" differently, the asset is represented as two separate assets "GDI1. Data privacy" and "CM2. Data privacy" (Figure 57). The rest of the assets are also numbered to reflect their stakeholder: "CM1. Company brand & reputation", "CM3. Application availability" and "CM4. Application interface usability".

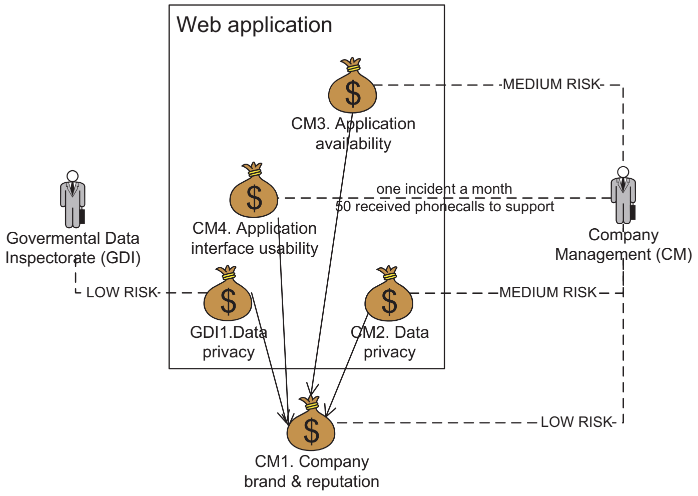

Web application
-MEDIUM RISK-
CM3. Application availability
one incident a month
50 received phonecalls to support
Govermental Data Inspectorate (GDI)
CM4. Application interface usability
Company Management (CM)
LOW RISK-
-MEDIUM RISK-
GDI1.Data privacy
CM2. Data privacy
-LOW RISK-
CM1. Company brand & reputation

Figure 57 – Asset diagram: complete with all details

If the analysis has several assets and stakeholders with conflicting concerns, it may be a good idea to keep the asset related information in two separate diagrams. One diagram may show the assets (direct & indirect) and the relations between these, and the other diagram may specify the stakeholders' risk acceptance values for each asset (without relations between the assets). When combining these two views, one faces the risk of having a potential confusing and cluttered diagram.

Each phase in this guideline summarizes the instructions in a table like the one shown below. Since the context establishment includes three steps, there are also three sections in this table:

The introductory meeting step	
Analysis tasks: • The security analysis method is introduced. • The client presents the goals and the target of the analysis. • The focus and scope of the analysis is set. • The meetings and workshops are planned.	People that should participate: • Analysis leader (required) • Analysis secretary (required) • Representatives of the client: - Decision makers (required) - Technical expertise (optional) - Users (optional)
Modeling guideline: 1. At this early stage of the analysis it can be useful to describe the target with informal like drawings, pictures or sketches on a blackboard. 2. The presentation can later be supplemented with more formal modeling techniques such as UML or data flow-diagram.	
The high level analysis step	
Analysis tasks: • The target as understood by the analysts is presented. • The assets are identified. • A high-level analysis is conducted.	People that should participate: • Security analysis leader (required) • Security analysis secretary (required) • Representatives of the client: - Decision makers (required) - Technical expertise (required) - Users (optional)

Modeling guideline:	
Asset diagrams: 1. Draw a region that logically or physically represents the target of analysis. 2. Place the assets within the region. 3. Indicate with arrows which assets may affect other assets. 4. Move assets, that are found to be indirect, outside the region. Indirect assets are only harmed as a consequence of a direct asset being harmed first. 5. Associate stakeholders (clients and other interested parties) with their assets. 6. Annotate each stakeholder-asset-relation with the maximum acceptable level of risk the stakeholder is willingly to accept. *Target descriptions:* 1. Use a formal or standardized notation such as UML, but ensure that the notation is explained thoroughly so that the participants understand it. 2. Create models of both the static and the dynamic features of the target. Static may be hardware configurations, network design etc., while dynamic may be work processes, information flow etc. 3. For the static parts of the description, UML class diagrams and UML collaboration diagrams (or similar notations) are recommended. 4. For the dynamic parts we recommend UML activity diagrams and UML sequence diagrams (or similar notations)	
The approval step	
Analysis tasks:	People that should participate:
• The client approves target descriptions and asset descriptions. • The assets may be ranked according to importance (optional). • Consequence scales must be set for each asset within the scope of the analysis. • A likelihood scale must be defined. • The client must decide risk evaluation criteria for each asset within the scope of the analysis.	• Security analysis leader (required) • Security analysis secretary (required) • Representatives of the client: - Decision makers (required) - Technical expertise (required) - Users (optional)
Modeling guidelines:	
- Modeling is not part of this step.	

8.2.2 Risk Identification

In the risk identification phase the security analysis leader and the brainstorming participants must find answers to questions like: *what are you most concerned about with respect to your assets* (modeled as threat scenarios and unwanted incidents), *who/what initiates these* (threats), and *what makes this possible* (vulnerabilities). This information is modeled in *CORAS threat diagrams*.

Consider the threat diagram in Figure 58 (and the extracted threat diagram for indirect assets in Figure 59). The diagram in Figure 58 focuses on network related threat scenarios towards the direct assets. The asset "CM4. Application interface usability" is not affected by these kinds of incidents and therefore left out. We have two threats: employees who may cause incidents by accident (human, accidental) and IT-infrastructure (non-human threat). Since this is a small example, we model both human and non-human threats in the same diagram, but often it can be useful to keep these in separate diagrams. The company management and the governmental data inspectorate are concerned about the following unwanted incidents: "disclosure of data", "corruption of data", and "unavailability of application".

We now explain one chain of events from the initiation caused by a threat to the left, to its impact on an asset to the right. The threat "IT-infrastructure" may first exploit the vulnerability "hardware failure" to make the server crash. Then it uses the vulnerability "poor backup solution" to initiate the threat scenario "application database fails to switch to backup solution". This threat scenario may again lead to the unwanted incident "unavailability of application", which impacts the availability of the application.

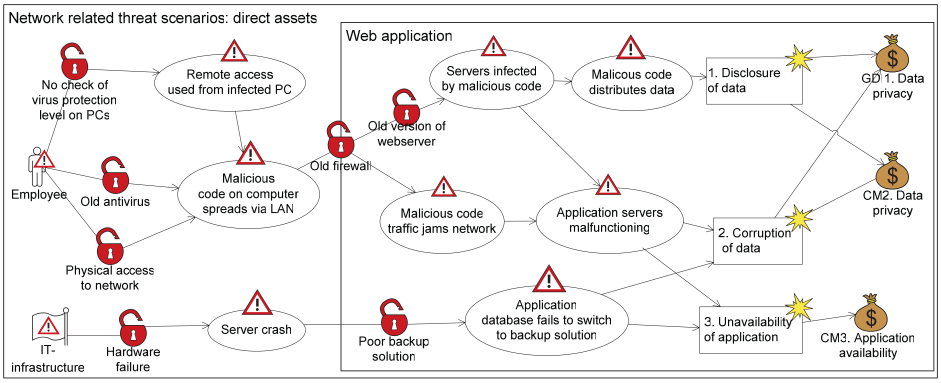

Network related threat scenarios: direct assets
Web application
No check of virus protection level on PCs
Remote access used from infected PC
Employee
Old antivirus
Malicious code on computer spreads via LAN
Physical access to network
Old firewall
Old version of webserver
Servers infected by malicious code
Malicous code distributes data
1. Disclosure of data
GD1. Data privacy
Malicious code traffic jams network
Application servers malfunctioning
2. Corruption of data
CM2. Data privacy
IT-infrastructure
Hardware failure
Server crash
Poor backup solution
Application database fails to switch to backup solution
3. Unavailability of application
CM3. Application availability

Figure 58 – Threat diagram: direct assets

To assess the indirect assets, the unwanted incidents that indirectly harm assets are specified separately (Figure 59). If they were to be modeled in the threat diagram above they would have been placed to the utmost right, but by extracting them into a separate diagram we can also include relevant incidents from other threat diagrams. Often indirect assets are of a more organizational or enterprise related kind, and therefore likely to be affected by several different types of incidents described multiple, different threat diagrams. To keep the unwanted incidents relation to their initiating threat, the threats are included in the diagram, while information about the vulnerabilities and threat scenarios are left out.

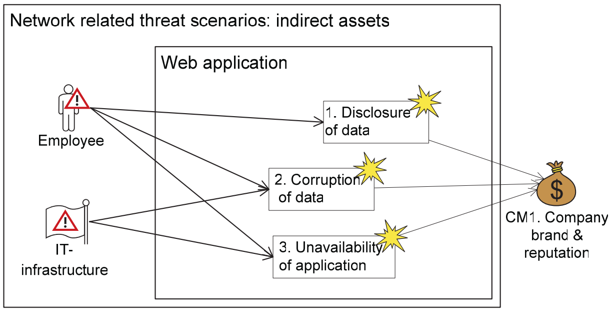

Network related threat scenarios: indirect assets
Web application
Employee
1. Disclosure of data
2. Corruption of data
IT-infrastructure
3. Unavailability of application
CM1. Company brand & reputation

Figure 59 – Threat diagram: indirect assets

The risk identification step	
Analysis tasks: The initial threat diagrams should be completed with identified threats, vulnerabilities, threat scenarios and unwanted incidents.	People that should participate: • Security analysis leader (required) • Security analysis secretary (required) • Representatives of the client: - Decision makers (optional*) - Technical expertise (required) - Users (required) **because this workshop often has a technical focus and the decision makers' competence is more relevant in the next step.*
Modeling guideline:	
Threat diagrams: 1. Use the region from the asset diagram. You may add more regions that group elements to increase the readability of the diagram. See also Sect.8.2.6 for further information about regions. 2. Decide how to structure the threat diagrams. A diagram may either focus on one asset at the time, a particular aspect of the target, or different kinds of threats. For instance, deliberate sabotage in one diagram, mistakes in an other, environmental in a third etc. ([76] contains a useful classification). This makes it easier to generalize over the risks, for example "these risks all harm asset X", "these risks are caused by human errors" or "these risks are related to the network". 3. Threats are placed to the left in the region; threats that can be classified as external to the target (hackers, intruders, third party services etc.) are placed outside the region. 4. Assets are listed on the right. 5. Unwanted incidents are placed within the region with relations to the assets they impact. 6. Assets that are not harmed by any incidents can be removed from the diagram. 7. Add threat scenarios between the threats and the unwanted incidents in the same order as they occur in real time (i.e. in a logical sequence). See Sect. 8.2.6 for tips and hints. 8. Insert the vulnerabilities before the threat scenario or unwanted incident they lead to. E.g.: a vulnerability called "poor backup solution" is typically placed before the threat scenario "the backup solution fails to run the application database correctly". 9. If specifying indirect assets at the outmost right of the threat diagram makes it too complex, draw separate threat diagrams for the indirect assets that shows the threats, unwanted incidents and the indirect assets only.	

8.2.3 Risk Estimation

The threat diagrams are input to the risk estimation where threat scenarios and unwanted incidents are assigned likelihood values and consequences. Likelihood values may be either qualitative (e.g. often, seldom etc.) or quantitative (e.g. 2 times per year, 50% probability etc.) depending on the type of analysis. If information on likelihood of threat scenarios and unwanted incidents is available, it may be added to the threat diagrams. For threat scenarios and unwanted incidents that are difficult to estimate, the analysis leader may give suggestions based on historical data like security incident statistics or personal experience. The likelihood of the threat scenarios can be used to extract a combined likelihood for unwanted incidents. In Figure 60 the final likelihood for "corruption of data" is based on the likelihood of the two threat scenarios "application servers malfunctioning" and "application database fails to switch to backup solution". Consequences are estimated for each "unwanted incident-asset" relation. The consequence value is taken from the asset's consequence scale that was defined during the context establishment.

There are different ways of computing the likelihood of an incident that may be caused by more than one threat scenario. If the estimates are suitable for mathematical calculations a computerized tool may be used. Since the likelihood scale in our case is in the form of intervals, the analysis leader decides to use an informal method that is quite straight forward and transparent, suitable for the brainstorming setting. For more precise calculation of probabilities fault tree analysis (FTA) [66] may be used. It is of course important that the combined estimates reflect reality, meaning that the combined estimates must be validated by the participants in the analysis.

Table 24 – Example of how one may combine likelihood estimates

Threat scenario	Likelihood	Unwanted incident	Combined likelihood
Application servers malfunctioning	1 occurrence per 5 years (1:5y)	Corruption of data	(1:5y)+(1:1y) = 6:5y which is so close to 1:1y that this value is used as a combined likelihood estimate.
Application database fails to switch to backup solution	1 occurrence per year (1:1y)		

We here use the consequence scale: large (3), medium (2) and small (1), but in a real security analysis more precise scales for each asset should be used.

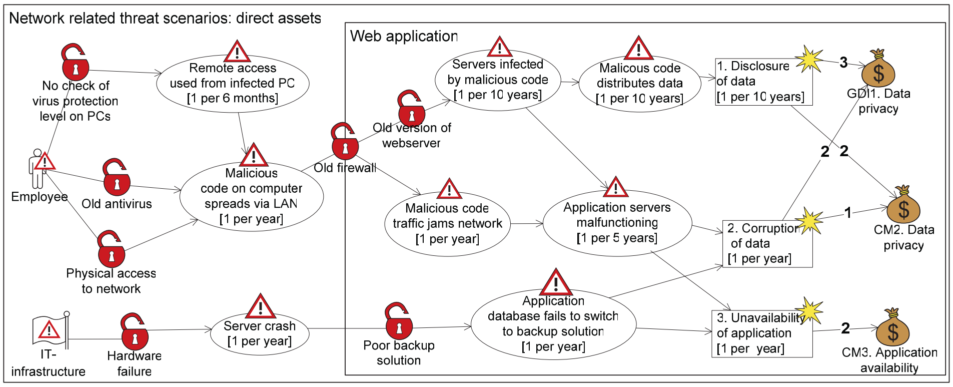

Network related threat scenarios: direct assets
Web application
No check of virus protection level on PCs
Remote access used from infected PC [1 per 6 months]
Employee
Old antivirus
Malicious code on computer spreads via LAN [1 per year]
Physical access to network
Old firewall
Old version of webserver
Servers infected by malicious code [1 per 10 years]
Malicous code distributes data [1 per 10 years]
1. Disclosure of data [1 per 10 years]
3
GDI1. Data privacy
2
2
Malicious code traffic jams network [1 per year]
Application servers malfunctioning [1 per 5 years]
2. Corruption of data [1 per year]
1
CM2. Data privacy
IT-infrastructure
Hardware failure
Server crash [1 per year]
Poor backup solution
Application database fails to switch to backup solution [1 per year]
3. Unavailability of application [1 per year]
2
CM3. Application availability

Figure 60 – Threat diagram (direct assets): likelihood and consequence estimation

In the threat diagram showing indirect assets (Figure 61), there are no threat scenarios that can be used as input for estimating the likelihood of the unwanted incidents. Instead one uses the estimates from the equivalent unwanted incidents in the direct asset threat diagrams and adjusts them either up or down according to the participants judgments. With respect to estimating consequences, we also use the 1-3 scale in the example, but could very well use a scale that specify in more details exact consequences for "CM1. Company brand & reputation". It is also permitted to split an incident into two incidents if the initiating threats give rise to potentially different consequences. In Figure 61 "Unavailability of application" is estimated to have a more serious consequence if it is caused by an employee, than if it is caused by IT-infrastructure.

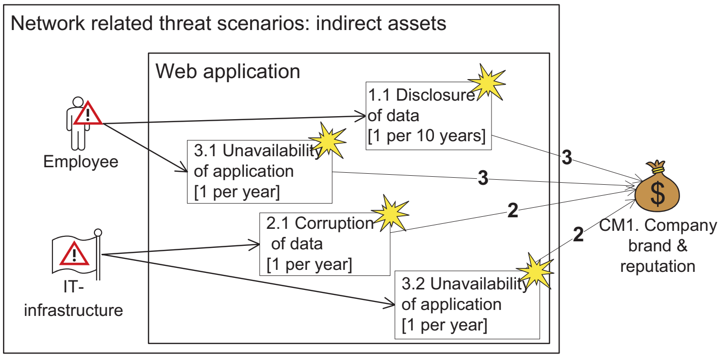

Network related threat scenarios: indirect assets
Web application
Employee
IT-infrastructure
1.1 Disclosure of data [1 per 10 years]
3.1 Unavailability of application [1 per year]
2.1 Corruption of data [1 per year]
3.2 Unavailability of application [1 per year]
3
3
2
2
CM1. Company brand & reputation

Figure 61 – Threat diagram (only for the indirect asset): likelihood and consequence estimation

The risk estimation step	
Analysis tasks: • Provide likelihood estimates for the unwanted incidents • Estimate consequences of the unwanted incidents. • Estimate risks on basis of the likelihood and consequence estimates.	People that should participate: • Security analysis leader (required) • Security analysis secretary (required) • Representatives of the client: - Decision makers (required) - Technical expertise regarding the target (required) - Users (required)
Modeling guideline: *Risk estimation on threat diagrams:* 1. Add likelihood estimates to the threat scenarios if this kind of information is available. 2. Add likelihood estimates to each unwanted incident, either directly, or based on the threat scenarios or unwanted incidents that lead up to it. 3. Annotate each unwanted incident-asset relation with a consequence taken from the respective asset's consequence scale. 4. If there are extracted threat diagrams for the indirect assets, do step 2 and 3 also for these. If the consequences are different according to which threat that gives rise to the unwanted incident, one may split the incident.	

8.2.4 Risk Evaluation

On the basis of the risk estimation we model the risks with their resulting risk values in risk diagrams, and optionally also in risk evaluation diagrams. Risk diagrams are used to give an overview of the magnitude of each risk, including the threats that are involved and the assets that are harmed. The risk evaluation diagram is similar, but shows whether the risks are acceptable or unacceptable according to the risk acceptance criteria defined during context establishment (Figure 57).

The risk diagram for direct assets (left diagram in Figure 62) illustrates that there is a low risk for disclosure of data caused by an employee, towards the asset "CM2. Data privacy", but a medium risk for the same incident towards "GDI1. Data privacy". For the indirect assets (right diagram in Figure 62) we see that the risk value of "Unavailability of application" is different according to which threat that is initiating it.

The preferred numbering convention is up to the modeler, we recommend that risks referring to the same unwanted incident start at the same number (e.g. "1.1 Disclosure of data" and "1.2 Disclosure of data"). We also recommend that the risks in the diagram for indirect assets refer to their "parent incident" or origin, starting with the same number as its parent (e.g. "1.1.1 Disclosure of data" in Figure 62).

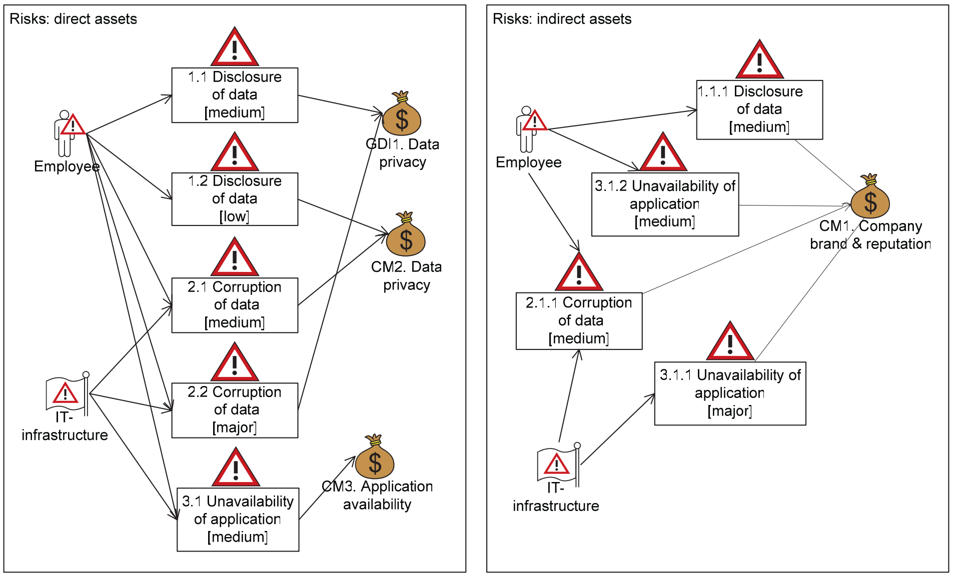

Risks: direct assets
Employee
IT-infrastructure
1.1 Disclosure of data [medium]
1.2 Disclosure of data [low]
2.1 Corruption of data [medium]
2.2 Corruption of data [major]
3.1 Unavailability of application [medium]
GDI1. Data privacy
CM2. Data privacy
CM3. Application availability
Risks: indirect assets
Employee
IT-infrastructure
1.1.1 Disclosure of data [medium]
3.1.2 Unavailability of application [medium]
2.1.1 Corruption of data [medium]
3.1.1 Unavailability of application [major]
CM1. Company brand & reputation

Figure 62 – Risk diagrams

To indicate those risks in the risk diagrams that are above risk acceptance criteria, one may draw a risk evaluation diagram. This is made by replacing the risk value with "acceptable" or "unacceptable" according to the risk acceptance criteria for each stakeholder and asset (Figure 63). We recommend that the difference between the two categories is also visualized using a color (in this case the acceptable risks are made gray). In a modeling tool this would typically be an automatic function.

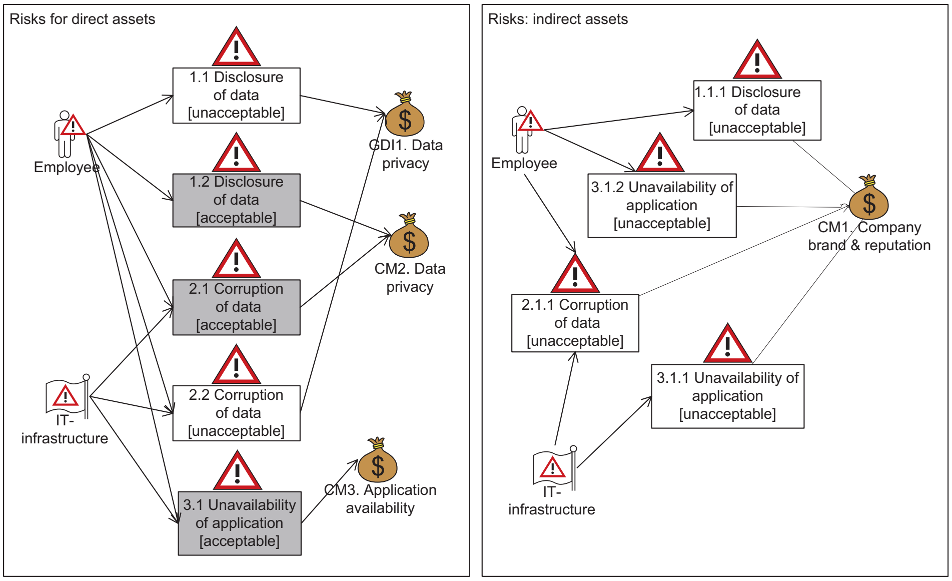

Risks for direct assets
1.1 Disclosure of data [unacceptable]
1.2 Disclosure of data [acceptable]
2.1 Corruption of data [acceptable]
2.2 Corruption of data [unacceptable]
3.1 Unavailability of application [acceptable]
Employee
IT-infrastructure
GDI1. Data privacy
CM2. Data privacy
CM3. Application availability
Risks: indirect assets
1.1.1 Disclosure of data [unacceptable]
3.1.2 Unavailability of application [unacceptable]
2.1.1 Corruption of data [unacceptable]
3.1.1 Unavailability of application [unacceptable]
Employee
IT-infrastructure
CM1. Company brand & reputation

Figure 63 – Risk evaluation diagrams

The risk evaluation step	
Analysis tasks: • Likelihood and consequence estimates should be confirmed or adjusted. • The final adjustments of the acceptable risk levels should be made (if needed). • An overview of the risk may be given in a risk diagram.	People that should participate: • Security analysis leader (required) • Security analysis secretary (required) • Representatives of the client: - Decision makers (required) - Technical expertise regarding the target (required/optional*) - Users (required/optional*) *Depends on whether this step is included in risk estimation or not. If it is part of the risk estimation workshop, all representatives should be present; otherwise it may be sufficient with only the decision makers.
Modeling guideline:	
Risk diagrams: 1. Use the threat diagrams and replace all unwanted incidents with risk symbols, showing the short risk description and the risk value. 2. Remove threat scenarios and vulnerabilities, but keep the relations between the threats and the risks. 3. If useful, split the risk diagrams into several diagrams according to type of threat, part of the target or asset importance (for instance, show all risks related to network, all risks for specific assets etc.). 4. If there are threat diagrams for indirect assets, do step 1 for these. 5. You may indicate those risks that are acceptable using risk evaluation diagrams. Replace the risk value in the risk diagram with "acceptable" or "unacceptable" according to the risk acceptance criteria. We recommend that the difference between the two categories is highlighted using a color label.	

8.2.5 Treatment Identification

During treatment identification, all the risks that are unacceptable are evaluated in order to find means to reduce their likelihood and/or consequence. Since treatments can be costly, they may be assessed with respect to their cost/benefit, before a final treatment plan is made.

The initial treatment diagrams are similar to the final threat diagrams except that every relation between an unwanted incident and an asset representing an *unacceptable risk* is symbolized with a risk icon and an identifier. Threats, threat scenarios, vulnerabilities and unwanted incidents that constitute acceptable risks only, should not be part of the treatment diagram.

Figure 64 shows the proposed treatments for the asset "GDI1.Data privacy": "Upgrade server", "Limit access to network" and "Increase awareness of security risks". The treatments in our example all address vulnerabilities, but one may also specify treatments for threats, threat scenarios or unwanted incidents. In our experience, the participants find it most intuitive to focus on treating vulnerabilities, using the analogy of closing a padlock in the meaning of closing the specific path through the diagram,.

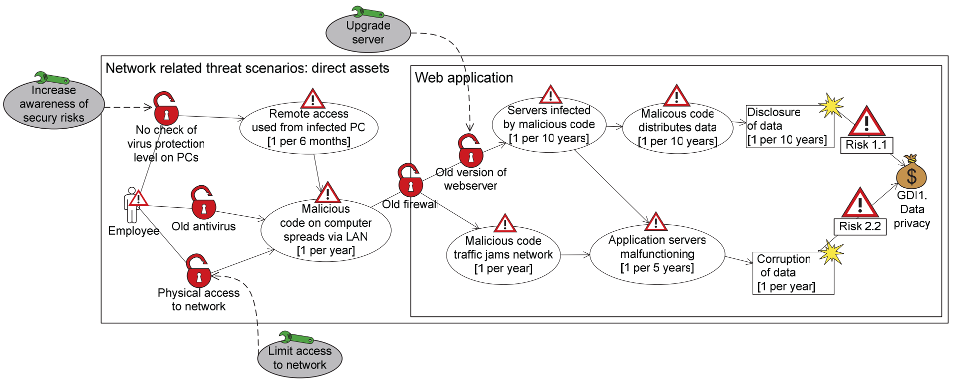

Upgrade server
Network related threat scenarios: direct assets
Web application
Increase awareness of secury risks
No check of virus protection level on PCs
Remote access used from infected PC [1 per 6 months]
Servers infected by malicious code [1 per 10 years]
Malicous code distributes data [1 per 10 years]
Disclosure of data [1 per 10 years]
Risk 1.1
Old version of webserver
Employee
Old antivirus
Malicious code on computer spreads via LAN [1 per year]
Old firewal
GDI1. Data privacy
Risk 2.2
Malicious code traffic jams network [1 per year]
Application servers malfunctioning [1 per 5 years]
Corruption of data [1 per year]
Physical access to network
Limit access to network

Figure 64 – Treatment diagram for the direct asset "GDI1.Data privacy"

Similarly, the treatments for the indirect asset "CM1.Company brand & reputation" includes the already mentioned treatments, in addition to "New backup solution" (Figure 65).

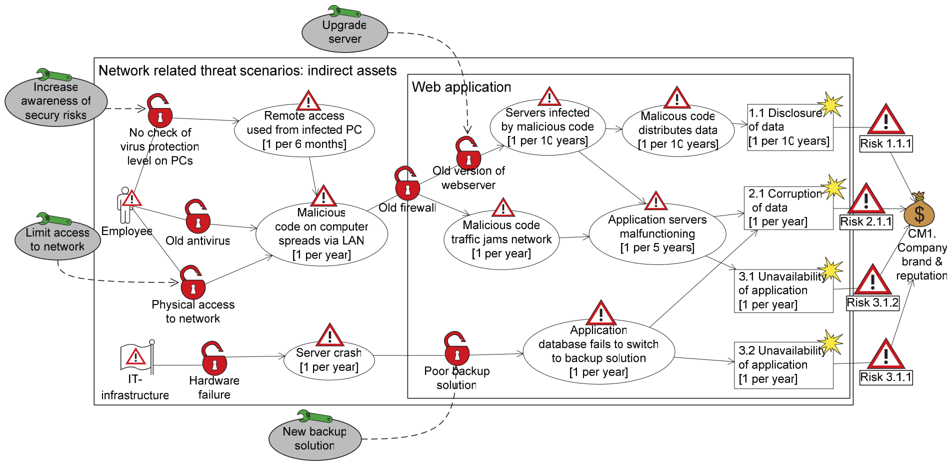

Upgrade server
Network related threat scenarios: indirect assets
Increase awareness of secury risks
No check of virus protection level on PCs
Remote access used from infected PC [1 per 6 months]
Web application
Servers infected by malicious code [1 per 10 years]
Malicous code distributes data [1 per 10 years]
1.1 Disclosure of data [1 per 10 years]
Risk 1.1.1
Old version of webserver
Old firewall
Employee
Limit access to network
Old antivirus
Malicious code on computer spreads via LAN [1 per year]
Malicious code traffic jams network [1 per year]
Application servers malfunctioning [1 per 5 years]
2.1 Corruption of data [1 per year]
Risk 2.1.1
CM1. Company brand & reputation
Physical access to network
3.1 Unavailability of application [1 per year]
Risk 3.1.2
Application database fails to switch to backup solution [1 per year]
IT-infrastructure
Hardware failure
Server crash [1 per year]
Poor backup solution
3.2 Unavailability of application [1 per year]
Risk 3.1.1
New backup solution

Figure 65 – Treatment diagram for the indirect asset "CM1.Company brand & reputation"

The treatments can be summarized in treatment overview diagrams shown in Figure 66 and Figure 67. To help avoiding cluttered diagrams we recommend gathering treatments that address the same risk in so called "junction points", and only use one arrow from the junction point to the particular risk.

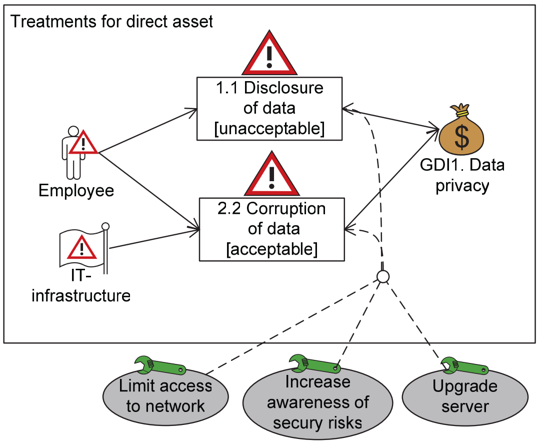

Treatments for direct asset
1.1 Disclosure of data [unacceptable]
Employee
GDI1. Data privacy
2.2 Corruption of data [acceptable]
IT-infrastructure
Limit access to network
Increase awareness of secury risks
Upgrade server

Figure 66 – Treatment overview: direct asset

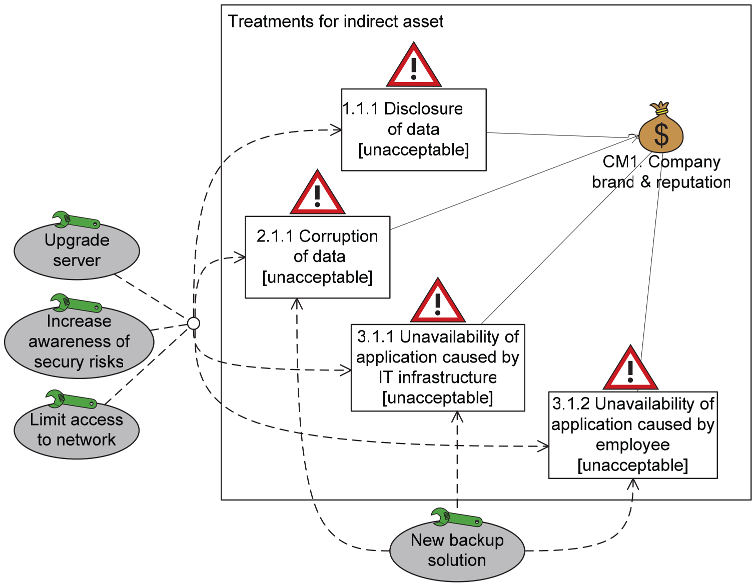

Treatments for indirect asset
1.1.1 Disclosure of data [unacceptable]
CM1. Company brand & reputation
2.1.1 Corruption of data [unacceptable]
Upgrade server
Increase awareness of secury risks
Limit access to network
3.1.1 Unavailability of application caused by IT infrastructure [unacceptable]
3.1.2 Unavailability of application caused by employee [unacceptable]
New backup solution

Figure 67 – Treatment overview: indirect asset

The risk treatment step	
Analysis tasks: • Add treatments to threat diagrams. • Estimate the cost/benefit of each treatment and • decide which ones to use. Show treatments in risk overview diagrams.	People that should participate: • Security analysis leader (required) • Security analysis secretary (required) • Representatives of the client: - Decision makers (required) - Technical expertise (required) - Users (required)
Modeling guidelines: *Treatment diagrams:* 1. Use the threat diagrams as a basis and annotate all arrows from unwanted incidents to assets representing *unacceptable* risks with risk icons. Relations and elements representing acceptable risks can be removed (this includes unwanted incidents, vulnerabilities, assets, threats and threats scenarios that do not constitute unacceptable risks). 2. Annotate the diagram with treatments, pointing to where they will be applied 3. If several treatments points towards the same risks (a many-to-many relation) we recommend using "junction points" to avoid multiple, crossing lines *Treatment overview diagrams:* 1. Use the risk diagrams as a basis; remove the acceptable risks (including threats and assets that are not associated with an unacceptable risk). 2. Add treatments as specified in the treatment diagram(s). Treatments directed towards vulnerabilities or threat scenarios should point towards the risks they indirectly treat. 3. If several treatments points towards the same risks (a many-to-many relation) we recommend using "junction points" to avoid multiple, crossing lines.	

8.2.6 Additional Explanations, Tips and Hints

The following explanations and recommendations are based on experiences from the industrial field trials in the SECURIS project. They are meant as additional help to the user.

Choosing the appropriate level of abstraction: this is often a difficult task for an inexperienced modeler. When identifying threat scenarios one should follow the strategy used in fault tree analysis. When modeling a fault tree, one starts with the top node (the unwanted incident) and works oneself down the tree by asking "what may cause this incident". When reaching an appropriate level of detail (i.e. manageable in complexity) the analysis stops. The same tactic should be used for threat diagrams. When adding an unwanted incident one should focus on identifying the threat scenarios that may cause the incident, thereby adding threat scenarios in a "backward" manner towards the threats. This way one avoids starting at a too detailed level by specifying extremely detailed initial threat scenarios, which may not contribute to the overall risk picture.

Regions and asset types: the regions are used as the link between threat diagrams and the target description. When designing the threat diagrams they should have a clear link to the target description, preferably visualized through the use of regions. The threat diagrams should be a more abstract view of the target descriptions, that includes risk related information but suppresses detailed technical information. The target descriptions function as underlying documentation that can be consulted in cases where the participants are uncertain or need to refresh their memory with respect to the target. If the target consists of logical or physical parts with different security levels this may be considered as different regions (e.g. a company’s intranet and extranet). When the guideline recommends to place the direct assets within the region and the indirect outside, it really means that the modeler should separate between assets at different abstraction levels. An indirect asset is an asset that is not directly part of the target of analysis, but nevertheless may be indirectly harmed in cases where assets within the target are harmed. We will explain this with an example: the emergency preparedness instructions for a major, international company states that the director of communication should give a statement in a press release if information about security incidents leaks to the press. The communication director must appear trustworthy in order to regain public confidence in the company. In a security analysis of the company’s information handling system, which also includes the emergency preparedness related to security incidents, the trustworthiness of the communication director has been defined as an asset. One day it is disclosed that the communication director has a secret Swiss bank account that has been withheld from the national tax authorities. Suddenly the trustworthiness of the communication director is questioned, and indirectly also the trustworthiness of the company itself. Indirect assets are therefore part of the CORAS method to illustrate that the consequences of a harmed asset may have rippling effects beyond the scope of the analysis. When deciding upon the direct assets, it is important to have in mind that these are strongly related to the target of analysis. They may represent functions, services, data or other aspects of the target that are particularly valuable to the company. During a security analysis an indirect asset may change status and become a direct asset and vice versa if findings suggest so or the scope is changed.

To ensure a good understanding of terminology and notation one should:

- Briefly repeat the meaning of central concepts in the beginning of each meeting to avoid confusion.
- Limit the amount of information in one diagram, and rather split it according to threat type, scenario type or asset type.
- Strive towards correctness from the very beginning. This means that the participants must be involved early, enabling them to adjust the diagram as they find appropriate.
- Provide a proper introduction to the notation and the diagrams that will be presented to the participants.

- Keep the assets within the scope of the analysis at the same abstraction level.
- Utilize checklists ensure that all relevant aspects are covered.
- Check whether assets, consequence scales and likelihood scales from previous analyses are relevant and can be reused.
- Document any changes to the diagrams during the brainstorming session including their rationale and origin since this represents important information.

How to decide the correct level of abstraction for the analysis, especially for the consequence scale? This is related to the whole security analysis process, and not specific to the modeling language used. It depends on the scope of the analysis and the experience of the analysis team. Or recommendation is to consult other analysts and previous analyses if available, or study the example given in the guideline to get an impression of how this has been solved in other analyses

How much time to spend on each step and how to prioritize between the activities? In general no analysis is the same, and the time spent depends on the target complexity, the competence of the participants and more. The guideline provides a suggestion for the number of meetings and workshops that is expected during a CORAS analysis. It is up to the analysis team to adjust this on a case to case basis. The analysts should consult examples or previous analyses to get an impression of how much information it is normal to handle during one meeting or workshop.

How to structure the brainstorming session so that most of the participants would be actively involved? If the participants cannot contribute to each others areas, split the brainstorming session into sub sessions with different focus. If participants feel they waste time by being present, they will become demoralized and not contribute to a fruitful discussion. Also in this case it may be a good idea to split up in smaller groups.

Is it necessary to follow all the steps in the guideline? If one chooses not to follow the guideline, it places more responsibility on the analysis team. In our opinion, the only situation where this may be an option is if the analysis team is highly competent in security analysis and thereby knows when to take short-cuts, but also recognize when certain aspects need more detailed analysis. However, for the non-experienced user we recommend following the guideline in detail.

9 Evaluation of the CORAS Language and the Guideline

This summarizes the results from the evaluations of the CORAS language and guideline. The evaluations are based on different approaches and have been conducted at different stages during design and development.

Some evaluations were carried out initially to explore, and capture design recommendations. They were:

- The quality evaluation of the UML profile using the quality evaluation framework described in *Chapter 6*. The UML profile was the starting point of our work. This evaluation has already been summarized in *Chapter 7*, and the full documentation is available in [57].
- The use of questionnaires on both students and professionals to evaluate the conceptual foundation on which our language has been built. See *Chapter 5* for a summary and *Appendices A* and *B* for the full documentation.
- The use of questionnaires on students to evaluate the use of risk related symbols vs. basic UML symbols. See *Chapter 5* for a summary and *Appendix A* for the full documentation.

Some evaluations were carried out during design and development. They were:

- The use of a questionnaire to evaluate and select graphical constructs. The results from this are summarized in Sect. 9.1. See *Appendix G* for the full documentation.

Some evaluations were carried out after, or when our approach was close to completion. They were:

- The use of experience reports from field trials to evaluate the language and guideline in an industrial context. See Sect. 9.2.
- The use of the quality evaluation framework described in *Chapter 6* to evaluate the CORAS language. See Sect. 9.3 below and *Appendix H* for the full details.

9.1 Evaluation Using Questionnaires

As explained in full details in *Appendix G*, questionnaires filled in by a group of professional system developers were used to select between different syntactic alternatives. We looked for ways of improving the readability and comprehensibility of threat- and risk diagrams by adding information in terms of text labels or simple graphical means. From this study, one conclusion was that text labels are preferred over visualization mechanisms like size, color or line type. For our language, this meant for example that that we decided to tag a major risk with the text label "major", rather than representing it with a large risk icon. In the following, we summarize the most important decisions made, based on this study.

To assess threats in a security analysis, it is useful to describe the paths via which they are most likely to harm the assets. By highlighting (attract attention to) or "sublightning" (remove attention from) paths in the threat diagram, one may visualize the most or least likely paths. None of the alternatives we investigated for these purposes were preferred by the subjects, except for the use of AND-gate symbols. The AND-gate is used to show how the occurrence of a threat scenario in a path depends on the preceding threat scenarios.

There may be different reasons for why none of the alternatives in the other cases were clearly preferred. The alternatives we tested may have been too similar, which mean that the subjects considered them equally good or bad. It may also be the case that the subjects paid little attention to likelihoods of relations *between* threat scenarios, but were more interested in the likelihoods of the actual scenarios. The study resulted in the following decisions regarding the CORAS language:

- the logical AND and OR-gate symbols from FTA and circuit design were included in the language (Figure 68 illustrates the use of the AND gate, for a full explanation of the figure we refer to Sect. 8.4 in *Appendix G*).
- the likelihoods of paths are not shown, likelihood estimates are only assigned to threat scenarios and unwanted incidents (Figure 68).

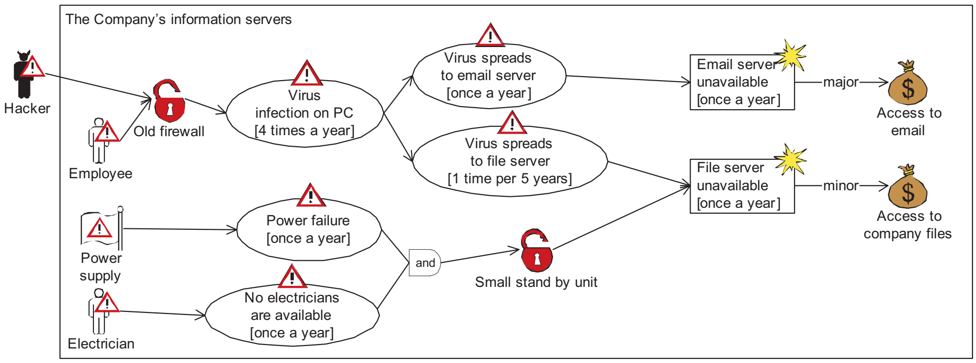

The Company's information servers
Hacker
Old firewall
Employee
Virus infection on PC [4 times a year]
Virus spreads to email server [once a year]
Virus spreads to file server [1 time per 5 years]
Email server unavailable [once a year]
File server unavailable [once a year]
major
minor
Access to email
Access to company files
Power supply
Power failure [once a year]
Electrician
No electricians are available [once a year]
and
Small stand by unit

Figure 68 – Example of the use of AND-gate in the threat diagram

Vulnerabilities are interesting in two different settings: it is useful to get an overview of the vulnerabilities of each asset, and it may be interesting to get an overview of which vulnerabilities a threat may exploit along its way in a threat diagram. We did not find it preferred to illustrate both views in the same diagram. One may instead leave out threats, threat scenarios and relations between these when only interested in the assets and their vulnerabilities. Since the CORAS language should be able to express both views, the first should be covered in threat diagrams and the second in specialized asset diagrams. This resulted in the following design decision regarding the language:

- vulnerabilities may annotate all relations in a threat diagram (Figure 68). This way of representing vulnerabilities attracts attention to the actual point where they are exploited. To identify which vulnerabilities an asset is exposed to, one may follow each path from the asset in question towards the threats. Nevertheless, we believe it is better to present this latter information in a specialized asset diagram (Figure 69).

The asset diagram in Figure 69 specifies which vulnerabilities each asset is subject to:

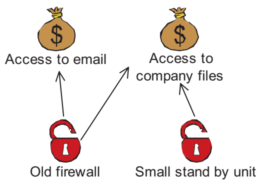


Figure 69 – Example of an asset diagram with particular focus on vulnerabilities

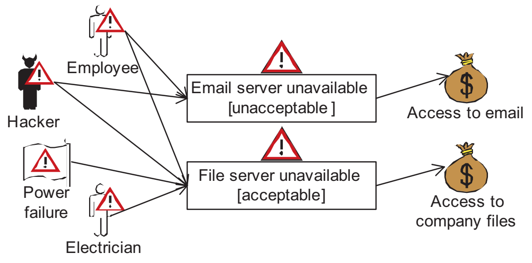


Figure 70 – Risk evaluation diagram illustrating whether the risks are acceptable or not

The risk evaluation diagram in Figure 70 illustrates which risks that are unacceptable/acceptable, the threats that may cause them and the assets that may be harmed. As we see, the employee and the hacker may cause both risks, while power failure and the electrician may only make the file server unavailable.

An important aspect of a security analysis is to identify the risks that are above the level of tolerance. The study indicates that it is preferred to use textual information labels to indicate the severity of risks and unwanted incidents, since size or color is too unambiguous. For the CORAS language, this means that textual information in the diagrams, possibly in combination with visual effects, is preferred over visual mechanisms alone. This resulted in the following design decisions regarding the language:

- we use text labels to indicate whether a risk is acceptable or unacceptable (Figure 70). The severity of unwanted incidents is not specified explicitly, but these are implicitly shown via their textual consequence- and likelihood labels in the threat diagram (Figure 68).

9.2 Evaluation Based on Experience Reports from Field Trials

In *Chapter 2*, we defined a set of success criteria for the thesis work. In this section, these are evaluated with respect to experience reports from industrial field trials. The field trials were carried out within the setting of the research project SECURIS (152839/220) funded by the Research Council of Norway. Each field trial consisted in a security analysis requiring about 250 person hours from the SINTEF analysis team, and 50-100 hours from the client. The analysis reports are confidential, but the scope and type of organization, and whether the author participated is shortly described in Table 25.

Table 25 – Recent field trials

Organization			Target of analysis	Hogganvik participated?
Name	Year	Type (Approx. # employees)		
DNV	2004	Vessel classification company (6400)	Information sharing service	Yes
NetCom	2004	Telecom company (700)	Mobile access to personal	Yes

			information	
Statnett	2005	Energy company (550)	Control and supervisory system	Yes
Hydro CSI	2005	Metal production company (1000, on plant)	Control and supervisory system	Yes
Hydro ICT	2006	ICT division, (metal, oil & energy) in Hydro (33 000, world wide)	Information sharing service	No
FLO/IKT	2006	Military organization (1000)	Procedures for handling restricted information	No
Statnett	2007[4]	Energy company (550)	Control and supervisory system	No

Two reports document every field trial: one presents the results from the security analysis and one summarizes experiences with the analysis methods and tools that were used. The first report was made available to the client only. The second report was open to the whole consortium. In this evaluation we consider only experience reports from the last five field trials: Hydro-05 trial [22], Statnett-05 trial [58], Hydro-06 trial [40], FLO/IKT-06 trial [132] and Statnett-07 trial [20]. This is to ensure that the statements reported are of adequate relevance since the way of conducting security analysis according to the CORAS method has evolved over the years (for example, before 2005 only the UML profile was used).

The experience reports provide information on issues such as what worked well, what did not work and why. The reports also contain data on how well the participants understand the diagrams, method, concepts etc. The author of the thesis did not participate in the last three field trials, but played a major role in the others.

The analysis team in the FLO/IKT-06 trial consisted of people with limited knowledge of the CORAS language. This field trial can therefore be seen as an exploration of the language and guideline by non-experienced modelers and their experiences are therefore extremely valuable. Some of their comments identified the need for additional explanations in the guideline that probably has been regarded as implicit by those familiar with the language. In most cases, their comments resulted in changes or the inclusion of new recommendations in the guideline.

In *Chapter 2* we defined the success criteria for the thesis work, targeting four main issues: the conceptual foundation, the language itself, the modeling guideline and the quality evaluation framework. In the evaluation based on the experience reports, only the first three are relevant. In the following, we discuss to what degree the validity of each success criterion is supported or not supported by the experience reports, and explain how identified problems have been addressed. All four success criteria are evaluated further in *Chapter 10*.

[4] Partly overlapping with the FLO/IKT-06 trial

9.2.1 The Conceptual Foundation

The following statements, relevant to the conceptual foundation, were found in the experience reports from the Hydro-05 trial [22], Hydro-06 trial [40], FLO/IKT-06 trial [132] and Statnett-07 trial [20] (nothing of particular relevance to the conceptual foundation was found in the Statnett-05 trial). Each statement is numbered, and followed by a discussion of its relevance including how we have dealt with it.

1. The CORAS terms used in the Hydro-05 trial were for instance threat, vulnerability, frequency, consequence and unwanted incident. Out of these, threat and unwanted incident were judged to be the easiest to grasp. Certain participants also found frequency quite understandable, while others thought this term was more problematic. Risk, consequence and asset were reported as less intuitive in the evaluation forms.

The reported difficulties in understanding the term "asset" is in our view mainly related to the abstract nature of the term itself. We have not seen the need for changing the definition, but have updated the guideline with an explanatory section addressing assets and asset types. Since the term "consequence" depends on the understanding of the term asset, we believe this will also reduce problems related to the notion of consequence. Although the notion of risk has proved difficult to understand in empirical investigations (*Chapter 5*), it has not been reported to be a problematic term in other field trials than Hydro-05. We therefore assume that this was a problem specific to this trial. It is however advisable to provide a careful introduction to the various terms at the start-up of a new analysis session, maybe even a discussion of them, to reach a common understanding. This has been included as a general recommendation in the guideline.

2. The asset identification and asset value estimation were found difficult during the Hydro-06 trial since the purpose of the activity and the asset-term itself were found difficult to understand. Also in the Statnett-07 trial and the FLO/IKT-06 trial, the analysis teams were uncertain of the difference between direct and indirect assets, and the purpose of this differentiation.

Our remarks regarding statement 1 with respect to the term assets is also relevant with respect to statement 2. The conceptual foundation does not distinguish between indirect and direct assets. In order to comply with international standards and be general enough to have interest beyond CORAS, the conceptual model is kept at a high level of abstraction. The differentiation between indirect and direct assets has been introduced in the guideline to help modelers achieve more precise diagrams, but is not really an issue of the conceptual model.

3. The ranking of assets according to importance created much discussion in the FLO/IKT-06 trial. The analysts assumed that this was the result of not all participants having a clear understanding of the concepts "asset", "consequence" and "frequency". There was a tendency to rank assets according to (expected) risk or frequency of related incidents, rather than importance.

The statement above shows that not having a clear understanding of the asset concept may affect also the understanding of related concepts. When the assets are precisely understood, the consequences in terms of potential damage to the asset are much easier to define. The misunderstood use of frequency during the ranking process is probably related to uncertainty with respect to the ranking of risks. Frequency is one of the most commonly used risk-related terms in the common language. We believe its definition is quite easy to understand, but assigning frequencies, or likelihoods, during an analysis is always difficult. Therefore, we assume the difficulties regarding frequency in this trial was related to identifying the *correct* frequency of threat scenarios and unwanted incidents, rather than understanding the definition itself. Sect. 9.2.3 also addresses the problem of ranking assets.

4. In the FLO/IKT-06 trial, it was found useful to briefly repeat the meaning of central concepts in the beginning of each meeting to avoid confusion.

Although this may appear obvious, it was included into the guideline to help non-experienced security analysis teams.

5. The analysis team in the Hydro-06 trial suggests that the assets within the scope of the analysis should be kept at the same abstraction level.

Also this has later been included in the guideline since it may help non-experienced modelers to manage their challenges related to asset identification.

Conclusion: The first success criterion for the thesis work is defined as follows:

1. The work of this thesis should include a conceptual foundation for security analysis, that:

a. specifies the main security analysis concepts and their relationships
b. uses a terminology that is easy to understand
c. uses a terminology that is in accordance with international standards
d. is described in a simple and understandable manner
e. provides a solid, underlying basis for a security risk modeling approach

Of these only 1b and 1d are of relevance here. The main problem related to the conceptual foundation, which has been reported in the five experience reports addressed here, is related to the concept asset (statements 1, 2 and 3). Except from these statements, we have not found any complaints concerning the understandability of terms in the experience reports. This fact may actually suggest that the terminology used in CORAS is well defined and quite easy to understand.

Statement 4 and 5 are not criticisms of the conceptual foundation itself, but rather recommendations based on practical use of the CORAS method that may be helpful to inexperienced users and therefore have been included in the guideline.

9.2.2 The CORAS Language

Statements relevant to the CORAS language were found in experience reports from these field trials: Hydro-05 [22], Statnett-05 [58], FLO/IKT-06 [132] and Statnett-07 [20]. Each statement is numbered, and then followed by a discussion of its relevance, including how we have dealt with it.

1. The use of the graphical models made it easier to involve the participants actively, and helped ensure an effective communication between the analysis team and the participants [Hydro-05].

This statement supports the appropriateness of the CORAS language for use in structured brainstorming sessions.

2. The overall experience from the use of the threat-modeling tool[5] during the Statnett-07 trial to model threats "on-the-fly" compared to the previous approach, where the analyst team made the threat diagrams between workshop 1 and 2, were good. The threat diagrams capture threat scenarios in an intuitive manner and it seemed easy for the participants to relate to the diagrams and provide input during the brainstorming sessions. The subjective experience of the analyst secretary is that threat diagrams are much better suited for storing threat related information during sessions than tables, as the notation is more compact and flexible and more suited to the format of the information obtained.

This statement supports the appropriateness of the CORAS language for being used actively to model the findings during structured brainstorming sessions. The language is considered easy to understand by the participants, and in particular suitable for documenting the information that is gathered during each step of a security analysis.

3. The participants emphasized that they need proper introduction to the notation and sufficient time to understand the information presented to them [Hydro-05].

Since one may schedule the meetings in a security analysis with weeks in between, the representatives of the client may not remember the details of the language from the previous meetings. Moreover, some representatives may be new to the analysis. We have therefore updated the guideline to remind the analysis team to repeat definitions of notation in each meeting.

4. One should limit the amount of information in one diagram, and rather split it according to threat type, scenario type or asset type [Hydro-05].

It may be challenging to a non-experienced modeler to know how much information to capture in a single diagram. As a general recommendation, it is better to include too little information per diagram than too much. A rule of thumb is that the diagrams must be readable when projected on a whiteboard in a normal meeting room. To help non-experienced modelers we updated the guideline to reflect this.

5. It is important to strive towards correctness from the very beginning. This means that the participants must be involved early, enabling them to adjust the diagrams as they find appropriate [Hydro-05].

This is particularly important with respect to the target descriptions since they often are prepared before the meeting. The client should be able to recognize their intended target of analysis in the descriptions. One should not waste precious time in the meetings to correct erroneously diagrams since this draws attention away from the purpose of the meeting. To ensure this, it is advisable that the client reviews the target descriptions before the actual meeting. We later updated the guideline to reflect this.

[5] The CORAS tool for security analysis and threat modeling (http://coras.sourceforge.net)

6. The diagrams capture information gathered during brainstorming sessions and one of the main concerns of the participants was whether the diagrams would be complete [Hydro-05].

To help overcome this we advice the analysis team to utilize relevant checklists to ensure that every important aspect is covered. This kind of checklist may be part of the CORAS method. This is a recommendation in the guideline.

7. Any changes to the diagrams during the brainstorming session represent important information and the analysis secretary should capture their rationale [Hydro-05].

Gathering some notes on the decisions made during the analysis is an important task of the analysis secretary. Participants may be present at one meeting, but not the next and vice versa, and therefore it is necessary to capture the rationale behind each decision. This includes information about who provided the information, why and how it affected the diagrams. Depending on how the analysis is organized, it may be the responsibility of the analysis leader to take these notes so that the secretary may concentrate on modeling. To stress this issue, we have included it as a general recommendation in the guideline.

8. In the Hydro-05 trial, the participants found the notation itself easy to understand and remember. It was considered to be a good way of visualizing threat scenarios and very suitable for presentations. According to one of the participants, this type of visualization emphasizes the "message" or the purpose of the analysis.

This statement supports the CORAS language's appropriateness for brainstorming sessions. It also shows that the participants in the analyses perceive it to be easy to understand, and that it successfully supports and documents the findings from each step of the analysis process.

9. In the Statnett-05 trial, the client found the CORAS language of high value when conducting threat and vulnerability assessments of complex ICT systems. The visualization method was in particular suitable for presenting problems and alternative solutions, prior to a final decision. In fact, the graphical visualization provided by the threat diagrams revealed the need for some specific security measures that were directly accepted by the participants. The client also considered the method useful in training of personnel to increase their awareness of security related issues of complex ICT systems [9].

This emphasizes the previous statement since it is taken from a different field trial.

10. The client in the Hydro-05 trial considered the way the CORAS language helps specifying the relations between threats and the chain of events that they may cause, the various states of the target, and potential incidents, as one of its main benefits. According to the participants, the modeling method made them more conscious of the target of analysis and its risks by representing threats and vulnerabilities more explicitly than "just talking" about them.

This statement supports one of the main goals of the CORAS language: to facilitate a precise documentation of the security analysis relevant information. It also shows that the participants in the analyses consider our graphical way of illustrating risks to make the security analysis more effective.

11. In the Hydro-05 trial, the participants said that the language provided an opportunity for documenting cause-consequence relations in a precise and detailed manner.

This statement also supports one of the main goals of the CORAS language, which is being a precise way of documenting the security analysis.

12. One of the hypotheses of the FLO/IKT-06 trial was that the use of CORAS diagrams facilitates the risk analysis process and documentation. The analysis team concluded that the hypothesis was confirmed, at least to the extent that it could be done without comparing the use of CORAS diagrams to alternative approaches. CORAS diagrams were used extensively in meetings #4 and #5 [risk identification and estimation]. The participants seemed to have a good understanding of the diagrams, and there were very little communication problems between the participants and the analysts. The analysis team assumed that one of the reasons for this was that the CORAS symbols are intuitive. The team also considered the technique of adding frequency values to the unwanted incidents, and consequence values to the unwanted incident-asset relations, during the risk estimation meeting to work very well.

In addition to fit the nature of the brainstorming session, the CORAS language should be easily understandable for the participants, and support and document the findings from each step of the analysis process. The result from the hypothesis testing in the statement above suggests that the CORAS language meets these requirements.

13. A justified question from the FLO/IKT-06 team is why threats but not vulnerabilities are included in the risk diagrams. As they point out, vulnerabilities are just as important as threats with respect to treatments.

The risk diagram was developed to provide a simple overview of the risks, from the initiator to the affected asset without any unnecessary "clutter". We understand the analysis team’s concerns and recommend a feature like this to be implemented in the CORAS tool. This could for example be a function that lets the modeler right-click and choose "hide" on any element within a threat diagram.

14. Representatives from the Norwegian Defense and Research Establishment (FFI) observed the use of the CORAS approach during the Statnett-05 trial. Their evaluation summary [10] with respect to the language concluded: "Modeling = useful! Facilitates a common understanding and makes it easier to communicate the results".

Also the independent observer of this field trial agreed that the CORAS language facilitated the analysis process.

Conclusion: The second success criterion for the thesis work is defined as follows:

2. The work of this thesis should specify a language for describing security risks, that:

- *a. is suitable for use in structured brainstorming sessions*
- *b. is easily understandable for the participants in the brainstorming, including those who receive the analysis results afterwards*
- *c. has a precise syntax, meaning its design should be based on:*
    - *i. best practice within information visualization*
    - *ii. experiences with realistic security risk scenarios*
    - *iii. users' preferences*
    - *iv. existing risk modeling techniques*
- *d. has a structured semantics that translates arbitrary diagrams into English*
- *e. supports and documents the different steps in the security analysis process*

Of these, only 2a, 2b and 2e are of relevance here.

2a) The suitability of the CORAS language for use in structured brainstorming is supported by statements 1, 2, 8, 9, 12 and 14.

2b) According to statements 2, 8, 9, 12 and 14, the language is easy to understand for the participants in the analysis. However, there is little empirical data on how well people who have not participated in the actual analysis understand the language (see discussion in Sect. 9.4.2).

2e) Statements 2, 8, 9, 10, 11, 12 and 14 together indicate that the language is suited for documenting each step in a security analysis.

Statements 3, 4, 5, 6, 7 and 13 provide advice and hints related to the use of the language that we have later included as recommendations in the guideline, or suggestions for tool support.

9.2.3 The Guideline

The modeling guideline was applied for the first time in the Hydro-06 field trial. Statements relevant to this guideline are therefore found in experience reports from the Hydro-06 [40], FLO/IKT-06 [132] and Statnett-07 [20] field trials.

1. The guideline was applied for the first time in the Hydro-06 trial, and was found to be very helpful in risk identification.

Since the modeler in this field trial was a non-experienced user of the CORAS language, we interpret this statement as a confirmation of the guidelines suitability for non-experienced modelers.

2. In the Statnett-07 field trial, the team chose not to conduct the risk treatment as a separate workshop, but rather include it at the end of the risk estimation. This meant that the analysts did not have time to remove the acceptable risks before treatments were suggested.

Better tool-support is probably the best way to address problems of this kind.

3. In the FLO/IKT-06 trial, the analysts chose not to follow the guidelines for the risk identification. Afterwards they concluded that by following the guidelines a better result could have been achieved, in terms of simpler diagrams and less work before and after the meeting.

We have developed the guideline based on a large amount of experience from several field trials. If one chooses not to follow the guideline, it places responsibility on the analysis team. In our opinion, the only situation where this may be an option is if the analysis team is highly competent in security analysis and thereby knows when to take short cuts.

4. In the FLO/IKT-06 trial, each step of the guideline was evaluated, providing these judgments:
   - Context establishment:
     - The modeling guidelines for asset diagram in the context establishment raised some questions and caused some confusion related to the difference between direct and indirect assets. The analysis team also pointed out the problem of having point 4 (indicate with arrows which assets may affect other assets) after point 2 (place the direct assets within the region). The team felt it more natural to decide how assets affect other assets before separating indirect assets from direct assets.
     - The modeling guidelines for target descriptions were found very helpful and the analysis team managed to produce target descriptions that were well understood by the participants, and also useful for later analysis, by following the instructions closely and using suggested languages (UML class diagrams and activity diagrams).
     - The guidelines for the approval step were also found adequate.
   - Risk identification: Although the analysis team deviated from the guideline at this step, they believed that the modeling guidelines for risk identification were helpful, but that they did not give sufficient guidance to ensure that an inexperienced analysis team was able to conduct a successful risk identification step.
   - Risk estimation: the guideline provides sufficient support for this step.
   - Risk evaluation: the guideline provides sufficient support for this step.
   - Risk treatment: the guideline provides sufficient support for this step.

Based on this feedback from the FLO/IKT-06 trial we chose to change the order and text of point 2-4 for the asset diagram. In addition, we have added a more detailed explanation of assets to the guideline. Except for the risk identification step, the statements above show that the analysis team judged the different parts of the guideline to provide sufficient support. With respect to the risk identification, we have later updated the guideline with more explanations and recommendations. As mentioned before, the analysis team in this trial consisted of inexperienced CORAS users.

5. The FLO/IKT-06 trial suggested that the guidelines should allow direct assignment of likelihood values to unwanted incidents in cases where such estimates can be just as good as estimates for threat scenarios

Since one may use the CORAS language both in a high level manner, as well as at a more detailed level, this simplification should be allowed. We have later updated the guideline to reflect this decision.

6. The importance of ranking the assets (in asset diagrams and in the approval step) was not clear to the analysis team in FLO/IKT-06 since the guideline does not use this information in the later steps.

During the field trials, the ranking of assets has shown more problematic than anticipated. Often it creates much debate and discussion without being of great importance for the rest of the analysis. A ranking of assets may be achieved indirectly from the risk acceptance levels. An asset for which only a low risk level is accepted is "ranked" as more important than an asset for which one allows a higher risk level. The guideline has therefore eased this requirement and made it optional.

7. The Statnett-07 trial pointed out difficulties in deciding the correct level of abstraction for the analysis, especially for the consequence scale.

This is related to the whole security analysis process, and not specific to the modeling language used. In our opinion, it depends on the scope of the analysis and the experience of the analysis team. Our recommendation is to consult other analysts and previous analyses if available, or study the example given in the guideline to get an impression of how other analyses have been solved.

8. In the FLO/IKT-06 trial, the analysis team requested suggestions for asset, consequence and likelihood scales.

For a non-experienced analysis team this may be very helpful, but it is not related to the graphical language in particular. In our opinion, this should be included as part of the CORAS method description.

9. The analysis team in the Hydro-06 trial suggested that one should keep proposing examples of assets in order to engage the participants.

This comment is a result of the difficulties the analysis team had with the asset identification. Instead of proposing numerous possible assets, the team should instead carefully explain the notion of assets and by this involve the participants in the discussion. When an analysis team has conducted a couple of security analyses, it will be better prepared since the assets in the various analyses often have quite similar characteristics. To help non-experienced users, the CORAS method should provide a list of assets based on experiences from field trials that one may use as a checklist. Assets like a company's brand & reputation, the availability of a service, the effectiveness of a work task and more, are examples of real, but also very general assets.

10. The analysts in the Statnett-07 trial requested tips on how to structure the brainstorming session so that most of the participants would be actively involved.

This depends on the scope of the analysis and the selection of participants. If the participants cannot contribute to each other's areas, our recommendation is to split the brainstorming session into smaller sessions focusing on issues that are more specialized. If participants feel they waste time by being present, they will become demoralized and not contribute to a fruitful discussion.

11. The FLO/IKT-06 trial requested guidance on how much time to spend on each step, and how to prioritize between the activities.

In general, no analysis is the same, and the time spent depends on the target complexity, the competence of the participants and more. The guideline provides a suggestion for the number of meetings and workshops that is expected during a CORAS analysis. It is up to the analysis team to judge this from case to case.

Conclusion: the third success criterion for the thesis work is defined as follows:

3. This thesis work should provide a modeling guideline for the modeler, that:

a. is rich and detailed with realistic examples
b. addresses non-experienced modelers
c. has recommendations based on user experiences
d. follows the security analysis process step by step

The statements extracted from the experience reports above, provide support for all of the sub criteria.

3a) we have extended the guideline with additional explanations to make the most difficult concepts more understandable (statements 2 and 3).

3b) several of the comments suggests that the guideline is suitable for non-experienced users, and after the extension with more explanations and recommendations it should have improved even more in this point (statements 1, 2, 3 and 4).

3c) the guideline is based on the field trials, and has been updated to include a number of practical recommendations based on experiences (statements 2, 3, 4, 7, 8, 9, 10 and 11).

3d) the guideline follows the security analysis process, and deviations from its instructions seem to be unwise (statements 2, 3 and 4).

9.3 Evaluation Using the Quality Framework

This section provides a summary of the quality evaluation of the CORAS language found in *Appendix H*. It includes an evaluation of how well the modeling language performs in modeling a set of "core security risk scenarios". It also presents the results and conclusions from the quality evaluation according to the language requirements described in *Chapter 6*.

9.3.1 Modeling the Core Security Risk Scenarios

The "core security risk scenarios" is a textual description of typical information gathered during a security risk analysis. The CORAS language has shown capable of modeling the relevant parts of this information. In the next paragraphs, we summarize the results from *Appendix H*.

Phase 1 – Context establishment:
The CORAS language was capable of modeling:

- The assets, including dependencies between them.
- The stakeholders.
- The indirect and direct assets.
- The risk acceptance levels of the stakeholder for each asset.
- The two stakeholders' conflicting risk acceptance level for the same asset.

This information was not modeled:

- The full description of the assets.
- The full description of the stakeholders.
- The scales of the assets, likelihood and consequence values.

When the stakeholders accept different risk values for an asset, the asset is split into two assets with reference to their stakeholder. If only one of the stakeholders value an asset, this is reflected by a specific reference to the stakeholder in the asset's name. We have chosen not to include more than the name of the assets and stakeholders in the models, because any extra information will clutter the diagram. The diagrams should be accompanied with textual explanations, which is a more appropriate place to give detailed descriptions of the elements in the diagram. Anything that are not directly necessary in order to read and understand the diagram should be avoided. The decision of not modeling the value scales was taken from a usability perspective. We believe this kind of information is more suitable for presentation in a table than in a graphical model because it only consists of static definitions.

Phase 2 – Risk identification:
The CORAS language was able to model:

- The threats, divided into human (accidental and deliberate) and non-human threats.
- The vulnerabilities, placed in a logical sequence of events with association to the threats that may exploit them.
- The threat scenarios with associations to relevant threats, vulnerabilities and unwanted incidents.
- The unwanted incidents with associations to the assets they may affect.
- A logical region symbolizing the target, for each type of threat.

This information was not modeled:

- The full description of the vulnerabilities.
- The full description of threats.
- The full description of assets.

As for the asset diagram, we decided to include only the names of the threats, vulnerabilities and assets to avoid messy and confusing diagrams. Any additional information about the elements should be stated in the accompanying textual description, which typically is required for the final analysis report.

Phase 3 – Risk estimation:
The CORAS language was able to model:

- The consequence of each unwanted incident towards the assets.
- The likelihood of each unwanted incident.

This information was not modeled:
- None.

It required very little effort to model the information from this phase using the CORAS language. In practice, the modeling consisted of adding textual information about likelihoods and consequences to the threat diagrams made in the previous phase. This makes the notation very suitable for modeling "on-the-fly" during risk estimation.

Phase 4 – Risk evaluation:
The CORAS language was able to express:
- The risks including their risk value.
- A separation of acceptable and unacceptable risks according to the risk acceptance levels of each stakeholder.

This information was not modeled:
- None.

The risk evaluation phase basically takes the information gathered in the risk identification and risk estimation and compares it to the risk acceptance levels from the first phase. The CORAS language had no problems modeling this information. The language provides a set of overview diagrams that can be very useful when presenting the findings of the analysis. The risk diagrams are created by "hiding" the vulnerabilities and threat scenarios between a threat and an unwanted incident in the threat diagrams. The unwanted incident symbol is replaced with a risk symbol with the appropriate risk value. Depending on the modelers choice of tools, this can be done either manually (like in Microsoft Visio) or semi-automatically in the CORAS tool. The risk evaluation diagram is a variation of the risk diagram where acceptable risks are grayed out. The latter diagram type is not necessary in order to model the core security risk scenarios.

Phase 5 – Treatment identification:
The CORAS language was able to express:
- The treatments, pointing towards the element in the threat diagram where they should be applied.
- An overview of the treatments that may be applied for each risk.
- An example of the risk diagram after applying a specific treatment.

This information was not modeled:
- The treatments' effect on the unacceptable risks.
- The estimated change in risk value each treatment may have on each of the unacceptable risks.

The decision of not modeling treatment effects in the diagrams is based on experience with the UML profile where this was a part of the notation. The treatment effect labels had a tendency of making the diagrams quite unreadable since they often conflicted with existing elements and relations. The CORAS language has the option for adding this kind of information to the treatment arrows, but we do not recommend using it, unless for very simple diagrams. The treatment overview diagram is an extra diagram type, suitable for presentation purposes in reports etc., but not required in order to model the core security risk scenarios. The CORAS language has no customized diagram for modeling the expected effects of the treatments, but one may illustrate this using a risk diagram. In our opinion, it not necessary to spend time on modeling this information since it is more efficient to present it in a table like in the core security risk scenarios.

9.3.2 Evaluating Against the Quality Requirements

The table below summarizes the scores for each category of quality requirements in the quality framework (see Sect. 6.2). It is important to keep in mind that the number of requirements within each category and their level of details vary. We have not weighted the categories to adjust for this, and therefore the scores provide an *indication* of weak and strong aspects of the CORAS language. The highest possible average score for each topic within the categories is 3.

Table 26 – Evaluation scores

Category		Must-requirements			Should-requirements		
		Score	# req.	Avg.	Score	# req.	Avg.
Domain appropriateness:	- modeling perspectives	6	2	3		-	-
	- concepts	41,8	14	3		-	-
	- symbols	30	10	3	9	3*	3
Participant language knowledge appropriateness		9	3	3		-	-
Knowledge externalizability appropriateness		8	3	2,7	1	1	1
Comprehensibility appropriateness:	- internal representation	18	7*	2,3		-	-
	- external representation	17	6*	2,8	8	3	2,7
Technical actor interpretation appropriateness		3	1	3	3	1	3
Organizational appropriateness		8	3	2,7		-	-

"# req." denotes the number of requirements
"-" means no requirements of this kind
* requirements were omitted, see *Appendix H* for details

As we can see from the scores, the CORAS language is judged to have maximum appropriateness factor for being used during security analyses (domain appropriateness). This means that it has all the required modeling perspectives, and includes the necessary concepts and symbols related to the security analysis domain.

The CORAS language is also perceived as easy to learn and understand based on the users' familiarity with other modeling languages (participant language knowledge appropriateness). Its resemblance with use cases and fault trees makes it easy to understand for new users, even though they are not familiar with the notation.

Due to its already existing tool support and its formal, textual syntax, it has a high appropriateness factor for computerized tools (technical actor interpretation appropriateness).

There is however some aspects that still can be further improved, mainly related to comprehensibility appropriateness, the language's ability to express all security related scenarios (knowledge externalizability appropriateness) and its appropriateness for various types of organizations (organizational appropriateness).

The comprehensibility appropriateness of the internal representation of the language (its concepts and relationships between these) received the lowest score. As the investigations of security analysis terminology has shown, there are terms that are more difficult than other terms, for example risk and frequency measures (See details in *Appendices A* and *B*).

On the other hand, these findings are not found to the same extent in the realistic settings of the field trials. Based on this we have reasons to believe that the artificial setting of the questionnaires influences the results, and that the concepts in fact are better understood than the initial investigations indicate. The language have been found to deviate from the requirement of using general rather than specialized concepts since threat is specialized into human or non-human threats, and human threats are either categorized as having deliberate origin or accidental (i.e. sabotage or blunder). This decision was made from a usability point of view, since the threat concept was found too general to comprehend for novices to security analysis and it was necessary to bring it down to a more pragmatic level. This is an example of where one of the general requirements in the quality evaluation requirement conflicts with requirements that should ensure understandability.

The criticism regarding the comprehensibility appropriateness of the external representation of the CORAS language is valid for almost all modeling notations (UML included) and is related to drawing aesthetically pleasing models. When the CORAS diagrams grow in size they can easily become cluttered and difficult to read. One may avoid this by dividing the diagrams into smaller and more manageable pieces of information. Crossing lines may also be a problem, for example from treatment to treated entity, unwanted incident to assets or from threat to threat scenarios. This means that the modeler relies on experience to know when the maximum level of tolerable complexity in a diagram is reached, and a splitting is necessary.

The main difficulty of achieving full score in knowledge externalizability appropriateness is related to the brainstorming setting and its free and creative nature. Since it is highly recommended to document the findings from the brainstorming immediately, preferably by modeling it "on-the-fly", it sets high demands to the analysis secretary. The secretary has to actively model new information, change and update existing models during the entire brainstorming session. The language receives top score for all the must requirements in this section, but not entirely the one that says modeling should be easy to conduct as part of the work task. The quality requirement is defined from a general modeling language point of view, and does not take into account the special modeling situation in a security analysis. This means that it may have to be interpreted in a slightly different manner, and assuming the modeler has been provided sufficiently tool support, the requirement may in fact be sufficiently fulfilled.

One of the requirements for organizational appropriateness is that the language must help the organization to achieve its goals, which in a security analysis setting means controlling its security risks. In order to validate this requirement one should monitor the organization before and after the introduction of the CORAS language, but since this is beyond the scope of our work, the requirement is considered sufficiently supported, but not fully supported. Such an evaluation may be a subject of future work. We have however found it to ease the explanation of security risks within the organization, and this is an important aspect of the process of controlling risks.

The low score on knowledge externalizability appropriateness for the should-requirements is related to expressing event trees. Should-requirements are not that important to the language quality, but more of the type "nice-to-have". Although event tree is an important technique within security analysis, it requires additional constructs beyond the current syntax of the CORAS language. The intention or meaning of event trees can however be expressed, but then in a form that does not look like traditional event tree notation.

9.4 Threats to the Validity of the Evaluations

This section summarizes the main weaknesses of the evaluations of the CORAS language.

9.4.1 The Validity of the Design Investigation

In this section, we describe the main threats to validity of the investigation of graphical preferences from Sect. 9.1 (details in *Appendix G*). First, the diagrams we used were less detailed and complex than real diagrams, but more than just examples of notation. Real risk and threat diagrams contain more descriptive text, and this additional text might make the text labels less visible. This weakness is difficult to avoid, and we should therefore validate our findings by testing them in real threat diagrams in a future field trial.

In this design investigation, we included several alternative representations addressing a selection of security analysis modeling challenges. It would be both interesting and useful to test also other aspects and modeling alternatives, but we had to limit the size of the material to increase the likelihood of it being completed and returned by the subjects.

The various modeling alternatives were tested on a population with considerable technical background. A background that is similar to that of most participants involved in security analyses and therefore considered a representative population. Nevertheless, it would have been useful to confirm our findings by testing them on a group with less technical and more management related background.

Since the experiment material was distributed by email we could not control how the subjects chose to fill in the questionnaire (helping facilities, time spent etc.). We do not believe this affected the overall validity of the results since the subjects were asked to choose the alternatives they *preferred,* i.e. a completely subjective measure.

The subjects received no other introduction to security analysis than the short background provided with the material. We do not know whether the results would have been different if the subjects had been familiar with security analysis. One of the success factors of the CORAS language is however that people without a background in modeling and security analysis may understand it. Therefore, we believe that it was a good idea to first test the language on this type of population, and rather test it on subjects with more competence on security risk analysis at a later occasion.

9.4.2 Strengths and Weaknesses Related to the Field Trials

The SECURIS field trials took place in realistic settings and addressed real security issues within the organization of the client. The representatives of the client contributed to the diagrams by providing information during the brainstorming sessions, but leadership and modeling was the responsibility of the analysis team.

A clear weakness of the field trials is how we have collected empirical data. The evaluation reports are based on the analysts' opinions of how well the method, language and modeling tool worked (except for the Hydro-05 and Hydro-06 trials where the participants responded to questionnaires). From the Statnett-05 trial the client and the observers from the Norwegian Defense and Research Establishment (FFI) have summarized their experiences with the CORAS approach [9, 10]. This means that the evaluation involved a kind of "self-evaluation" of our own method and language. To make this self-evaluation of the CORAS language as independent as possible, the author of the thesis did not participate in the analysis team for the last three field trials. The limited number of hours available often made the analysis team choose between completing the analysis tasks and spending the client's time on evaluation. When the aim of the field trials was to test CORAS in a realistic setting and deliver complete analysis results to the client (as if they had hired in a consultancy firm), the extent to which the client was involved in the evaluation part had to be reduced.

One of the points where we lack empirical support is related to how well the analysis documentation in form of CORAS models are understood by decision makers and others that receive the final analysis report. Within the scope of the SECURIS field trials there was no room for additional instigations beyond the actual security analyses, but this is subject to further work.

9.4.3 Threats to the Validity of the Quality Framework Evaluation

The quality framework evaluation consists of two parts, the core security risk scenarios and the quality requirement evaluation.

The core security risk scenarios represent a typical security risk analysis based on our experience from several industrial field trials [22-24, 58]. The scenarios give a step-by-step example of the information that is expected to be identified during the analysis and how it can be modeled. This can be used to compare different analysis and modeling strategies like a "benchmarking test". The scenarios are influenced by analyses within various industrial sectors that all had different targets and scopes. They have been carefully defined to make them as general as possible, but also include the appropriate level of details. Later field trials have shown the need for AND-gates, but the scenarios do not include an example of this. If one chooses to revise the scenarios, this is suggested for inclusion. The core security scenarios can to some extent be modeled in different styles, depending on the preferences of the modeler. Deciding how well a modeling language expresses the scenarios is partly a subjective measure. One the one side, the modeling language should be able to express the main parts of the information gathered at each step in the analysis, but on the other side one has to ensure that the amount of details does not affect the readability of the diagrams.

The quality framework SEQUAL has been developed for modeling languages in general. It has therefore been necessary to add several requirements that are specific to security risk analysis. One may claim that the set of extra requirements is incomplete or contains superfluous requirements, but at the time it was defined it seemed complete and well justified. Nevertheless, after two applications of the framework we have found some requirements to be overlapping, some are a bit irrelevant and some need a particular interpretation when applied to security analysis modeling languages (see *Appendix H*).

The quality evaluation framework has not been evaluated or tested by others than the author. Moreover, the second application of the framework was to evaluate a language made by the author. This is not sufficient to claim its validity as a general framework suitable for use by people unfamiliar with CORAS, SEQUAL and more. To support its validity, it should be tested by applying it to other security modeling languages, or at least use it for an independent evaluation of the CORAS language to see whether the results in *Appendix H* are confirmed.

10 Discussion

This chapter consists of two parts. In the first part, Sect.10.1, we evaluate and discuss to what extent we have reached our research objectives. In the second part, Sect. 10.2, we outline alternative research directives and identify some interesting issues for further research.

10.1 Fulfilling the Success Criteria

In *Chapter 2*, we characterized our research objectives in terms of success criteria. In the following, we investigate to what extent they have been fulfilled.

10.1.1 The First Success Criterion

1. The work of this thesis should include a conceptual foundation for security analysis, that:
   a. specifies the main security analysis concepts and their relationships
   b. uses a terminology that is easy to understand
   c. uses a terminology that is in accordance with international standards
   d. is described in a simple and understandable manner
   e. provides a solid, underlying basis for a security risk modeling approach

The sub criteria of the success criterion are supported as follows:

1a) The main security analysis concepts and their relationships are defined in the conceptual model in *Chapter 5*. See *Appendix A* and *Appendix B* for the full documentation.

1b) Many security and risk related concepts are used informally in English and other natural languages. To avoid misunderstandings we have conducted empirical investigations of how common people understand these concepts. We summarize the results of these in *Chapter 5* and present the full details in *Appendix A* and *Appendix B*. These investigations have helped ensure that the conceptual foundation captures security analysis concepts in accordance with common understanding. The experience reports from five industrial field trials [20, 22, 40, 58, 132] in which the CORAS language and conceptual foundation have been used, indicate very few problems related to the concepts and relations between them (see also *Chapter 9*).

1c) The underlying foundation supports the CORAS security analysis method which follows the Australian/New Zealand Standard for Risk Management [6]. This standard has later been incorporated in the ISO17799/ISO27002 standard [82]. Terms specific to information security complies with the ISO/IEC13335 standard [76].

1d) The conceptual model in *Chapter 5* is described using a simple UML 2.0 class diagram, and is accompanied with an explanatory example.

1e) The conceptual foundation for the CORAS language may function as a basis for security risk modeling approaches that use concepts from the international standards that the definitions in our foundation are taken from.

10.1.2 The Second Success Criterion

2. The work of this thesis should specify a language for describing security risks, that:
    - a. is suitable for use in structured brainstorming sessions
    - b. is easily understandable for the participants in the brainstorming, including those who receive the analysis results afterwards
    - c. has a precise syntax, meaning its design should be based on:
        - c.i best practice within information visualization
        - c.ii experiences with realistic security risk scenarios
        - c.iii users' preferences
        - c.iv existing risk modeling techniques
    - d. has a structured semantics that translates arbitrary diagrams into English
    - e. supports and documents the different steps in the security analysis process

The sub criteria of the success criterion are supported as follows:

2a) The CORAS method is based on extensive use of structured brainstorming. The CORAS language has been designed for use in workshops where the diagrams are modeled "on-the-fly". Since the language may be used in high level as well as more detailed analyses, the diagrams may be extended with more information as the analysis progresses. In the industrial field trials [20, 22, 40, 58, 132] we have often received positive feedback with respect to the suitability of the CORAS language in brainstorming sessions. For instance, the client in the Statnett-05 field trial stated that the language was of high value when conducting threat and vulnerability assessments of complex ICT systems [9]. More evidence from the field trial evaluations supporting the fulfillment of this sub criterion is found in Sect. 9.2.2.

2b) The language has been judged easy to understand by participants in several field trials. Statements supporting this are provided in Sect. 9.2.2. According to one of the participants in the Hydro-05 trial, the language is considered to be a good way of visualizing threat scenarios and very suitable for presentation. We lack empirical data on how the CORAS models are understood by decision makers and others that read the final report, and this is suggested for further work.

2c) The graphical and textual syntaxes of the CORAS language are summarized in *Chapter 8*, and defined in detail in *Appendix D.*

2c.i) The empirical investigation summarized in Sect. 9.1 tested several different syntax alternatives adhering to best practice within information visualization, to identify the most suitable way of capturing relevant aspects syntactically. We built heavily on the results form these in the selection of concrete syntax. A professional, graphical designer has designed the icons with the aim to match the concepts they symbolize in an intuitive manner. For instance, we emphasized that the icons should be easily discriminated, both in color and in black/white print.

2c.ii) The syntax has evolved in order to handle the security analysis scenarios we have encountered in the field trials. The core security risk scenarios defined in *Chapter 6* are inspired by experiences from the field trials [22-24, 58], and as argued in *Appendix H* the CORAS language is well suited to capture these. Based on this, and experiences from the field trials [20, 22, 40, 58, 132] we claim that the CORAS language can handle realistic security risk scenarios.

2c.iii) During the entire development of the CORAS language, we have had a strong focus on user preferences. The underlying foundation, on which the language builds, was defined to meet the users' intuitive understanding of the terminology (see investigations in Sect. 5.2). The decision on whether to use special risk related graphical icons was based on empirical investigations involving users (see investigation in Sect. 5.3). The final decisions regarding the concrete syntax were based on the results from the empirical investigation described in Sect. 9.1 which also focused on users.

2c.iv) The development has been inspired of well-known risk analysis techniques like fault trees and event trees (often called cause-consequence diagrams when put together). Since we identified the need for tree-like graphs with multiple top events (not allowed in fault trees) we could not adapt the fault tree notation completely. The intention of event trees may also be described. If one needs to express fault trees we have included the characteristic AND/OR gates from this notation. The inclusion of these operators is a result from the empirical investigation described in Sect. 9.1.

2d) The semantics of the CORAS language facilitates a schematic translation of any diagram into a paragraph in English (Sect. 8.1.3 and 8.1.4, full details in *Appendix D*). The semantics translates a diagram in two steps: (1) from the graphical syntax to the corresponding textual syntax, (2) from the textual syntax into corresponding sentences in English. The semantics is structured in the sense that we may translate each vertex and each relation separately. The meaning of a diagram as a whole is the set of sentences one gets from applying the semantic mapping to its vertices and relations.

2e) The modeling language offers customized diagrams for each step of the security analysis. Which kind of diagram that is suited for each step is described in the modeling guideline in *Chapter 8*, including detailed modeling instructions. Compared to traditional documentation techniques, which are mainly based on the use of tables that are filled in manually, the CORAS language is based on the vision that the users should concentrate on the graphical modeling of threat scenarios and let a tool extract the relevant information and automatically fill in tables. Field trial evaluation reports support the CORAS language's appropriateness for documenting the process in this manner (*Chapter 9*).

10.1.3 The Third Success Criterion

3. This thesis work should provide a modeling guideline for the modeler, that:
    a. is rich and detailed with realistic examples
    b. addresses non-experienced modelers
    c. has recommendations based on user experiences
    d. follows the security analysis process step by step

The sub criteria of the success criterion are supported as follows:

3a) The guideline provides an example-driven introduction to the CORAS method and CORAS language. The example addresses a security analysis of the same kind as those we have experience from in the field trials. In this example, we follow the analysis process from the start until the end. The guideline has been extended and revised several times based on the experience reports [20, 22, 40, 58, 132], and practical recommendations has been included (Sect. 8.2.6).

3b) We have designed the guideline to address non-experienced modelers and statements found in the experience reports indicate that it really does (see Sect. 9.2.3). More recently, we have updated the guideline with an additional section (Sect. 8.2.6) containing recommendations, tips and hints based on experiences from practical use.

3c) The development of the guideline has been driven by experiences from the industrial field trials. From the experience reports we have gathered recommendations, tips and hints and included them into the guideline (see Sect. 9.2 and Sect. 8.2.6).

3d) The guideline in *Chapter 8* is structured according to the steps in a security analysis process. It describes the purpose of, and the expected outcome from each step, including detailed modeling instructions. The guideline also suggests which roles that should be present from the client's side at each meeting or workshop.

10.1.4 The Fourth Success Criterion

4. This thesis work should provide a framework for evaluating the quality of security risk modeling approaches, that:
    a. covers quality requirements related to modeling languages in general
    b. includes quality requirements special to security risk analysis
    c. includes a modeling test case representing core security risk scenarios

The quality evaluation framework defined in *Chapter 6* was first applied to evaluate the UML profile [124] and then to evaluate the CORAS language (*Appendix H*). The evaluations have provided a good overview of the strengths and weaknesses of both languages. The sub criteria of this success criterion are supported as follows:

4a) To ensure the inclusion of quality requirements relevant to modeling languages we have used the quality framework SEQUAL [100-104] as a basis when defining the quality requirements (see *Chapter 6*). SEQUAL provides a set of general requirements that any modeling language should fulfill.

4b) In addition to the quality requirements from SEQUAL, we have added requirements aiming to capture the specific needs of the security analysis domain. These are related to security analysis concepts and the relevant modeling needs identified through the industrial field trials.

4c) The quality evaluation framework also includes a realistic modeling test case (the core security risk scenarios) to ensure its practical usefulness. The core security risk scenarios are inspired of the analysis topics addressed in the DNV-04 [23], NetCom-04 [24], Statnett-05 [58] and Hydro-05 [22] field trials. These analyses are of course confidential, but they provided good indications regarding the complexity of the information gathered in a security analysis. They also increased our awareness of the challenges posed to a security risk modeling language. By embedding these experiences into general security related problems, relevant to any modern business organization, we ensured the relevance of the core security risk scenarios. The core security risk scenarios are used as a benchmarking test with respect to the expressive power of the language, its modeling ease, elegance and more.

10.2 Things We Could Have Done Differently and Future Work

The research method we have used and the design we ended up with requires many choices on our behalf. In this section, we account for various alternative strategies and solutions and explain why we did not select them. We distinguish between strategies and solutions related to the design of our artifacts and strategies for evaluation. We also indicate some issues for future research.

10.2.1 Strategies and Solutions Related to Design

We believe one should adhere to international standards when defining risk related concepts for inclusion in the language. However, this does not ensure that these concepts match intuition. In our work, we particular experienced this kind of problem in relation to the concept asset. We believe it would have been fruitful to work more on the explanations related to asset identification; in particular develop more detailed guidelines for how to conduct the asset identification step of the analysis. The guidelines should focus on describing how to identify the "correct" assets, how to ensure that they are kept at the right level of abstraction, and how to define suitable consequence scales. We know this is one of the most difficult parts of the CORAS method, and it should be subject to further work.

Deciding upon, and designing the graphical icons for the language depends on two main tasks, first selecting a concept that represents the term, and second designing a graphical icon that represents the concept. For some of the terms we had problems in finding alternative concepts that could represent the term (e.g. vulnerability was difficult), whereas other terms were more straightforward (e.g. asset may be represented as monetary values). A professional designer made 1-3 alternative graphical icons for each concept, and we selected the one we felt represented the term best (for example for the asset we selected the moneybag icon over the coin and the gold barrel). Due to limited financial resources, it was not possible to have the icons tested on a broader test group in for instance an empirical study. It is quite expensive to hire a graphical designer, and the project could only fund one design iteration for the graphical icons. We are not aware of other concepts and icons that may represent the terms better, but cannot exclude that such exist.

A concept that may be included in future versions of CORAS (in the conceptual foundation, modeling language and guideline) is the notion of "barriers". Barriers represent existing security mechanisms in the target that have been implemented to reduce the likelihood of threat scenarios and unwanted incidents (e.g. a firewall is a barrier). Barriers may also be implemented to reduce the consequences of potential unwanted incidents. The concept of barriers can be very useful in the process of estimating risks. If the target has several strong barriers, the likelihood of unwanted incidents is reduced, whereas weak or non-existent barriers means that unwanted incidents are more likely to happen. One may also judge the strengths of the barriers by classifying them into different categories according to type. An "organizational barrier" in the form of a written security policy is weaker than a physically separated network, which represents a "physical barrier". A "symbolic barrier", in the form of a pop-up message that warns the user when he/she connects to an unsecured wireless network, is stronger than a policy against using unsecured networks (organizational barrier), but weaker than a physical barrier that makes it impossible to access unsecured networks.

The current version of the CORAS approach provides little support for combining results from different analyses (for instance, analyses of two components of the same system). This kind of modularity might for example reduce the effort related to keeping an analysis documentation updated, since only the part where a change has been made needs to be re-analyzed. This has not been within the scope of our work, but ongoing research [21] is investigating this problem, and this may in the future result in new versions of the CORAS approach.

To facilitate more advanced tool-based support, a semantics based on translating to paragraphs in English, may not be sufficiently precise. A more precise semantics is necessary, particularly with respect to developing automatic analysis functions:

- From field trials, we know that there is a need for automatic computation of likelihood values of unwanted incidents.
- In treatment identification there is a need for functions that help choosing the optimal combination of treatments. These should be optimal with respect to risk level, costs, efficiency and more. This will require a more detailed specification of each treatment than the current version of the CORAS approach offers.
- The risk evaluation diagrams should be generated automatically based on the risk diagrams and the risk acceptance levels. This will also allow for an investigation of how an adjustment of the risk acceptance levels will affect the overall risk picture.

The development of a sufficiently precise semantics for such purposes has however, not been within the scope of our work, but is a subject for further work.

10.2.2 Strategies for Evaluation

The project context and setting of the industrial field trials required us to conduct security analysis according to the CORAS method in a professional manner, resembling commercial analyses. This was a prerequisite for involvement by many of the industrial partners. This of course, ensured that the CORAS method was tested in realistic settings, but unfortunately it also restricted our possibilities to explore different modeling and analysis strategies. It would for instance been interesting to analyze the same system twice for the same client, one using the CORAS language and one using their own approach to see the differences. There are of course a number of practical and economical considerations related to this kind of experiments. Amongst others, the client would have to provide two sets of participants with similar competence (often very difficult in practice) and the time they spend on the second analysis might have to be compensated financially.

To improve the guideline further, it would be interesting to test it on users without previous background in CORAS and get them to report their experiences. Within our research project, there have not been resources to carry out this kind of test.

One of the points where we lack empirical support is related to how well the documentation in the form of models in the security analysis reports are understood by decision makers and others who are supposed to make use of the results and recommendations. This has much to do with the confidentiality aspects of the analyses. One way we may test this is to have the client pointing out a trusted person within its organization, who have no previous knowledge of the analysis, and have him/her to read the analysis report. The analysis team could then interview this person to see whether the diagrams were understood. It would also be interesting to revisit the client after some time, to see how the findings of the analysis have had consequences for the organization. This has however not been possible within the setting of our work, but may be considered in the future.

The structured semantics has only been tested by the developers. An interesting experiment would be to have one student group translating a set of diagrams into sentences in English, and then have a different group modeling these textual descriptions back to diagrams. By doing this, one could explore whether some information is lost in the translation. If so, it means that there is some information in the diagrams that is not captured by the semantics. This could for instance be the order or the placement of elements in the diagram.

References

[1] Aerospace Recommended Practice (ARP) 4761: Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment: Society of Automotive Engineers Inc, 1996.

[2] AIChE, "Guidelines for Chemical Process Quantitative Risk Analysis", American Institute of Chemical Engineers, Center for Chemical Process Safety, 1989.

[3] Alberts, C., J. ; and Dorofee, A. J., "OCTAVE Criteria Version 2.0", Tech. report CMU/SEI-2001-TR-016. ESC-TR-2001-016, 2001.

[4] Alghathbar, K. and Wijesejera, D., "authUML: a three-phased framwork to analyze access control specifications in use cases", in Proc. ACM Workshop on Formal methods in security engineering (FMSE'03), pp. 77-86, 2003.

[5] Appukkutty, K., Ammar, H. H., and Goseva-Popstojanova, K., "Software Requirement Risk Assessment using UML (Short paper)", in Proc. ACS/IEEE International Conference on Computer Systems and Applications, pp. 112-115, 2005.

[6] AS/NZS4360, Australian/New Zealand Standard for Risk Management: Standards Australia/Standards New Zealand, 2004.

[7] AS/NZS4360, Australian/New Zealand Standard for Risk Management: Standards Australia/Standards New Zealand, 1999.

[8] Barber, B. and Davey, J., "The Use of the CCTA Risk Analysis and Management Methodology CRAMM in Health Information Systems", in Proc. MEDINFO'92, pp. 1589-1593, 1992.

[9] "BAS5 - Statnett-case ved bruk av CORAS-metoden: ROS-analyse av IKT-systemene ved RS' kontrollsenter" (document summarizing the client's experiences), Statnett, 2006-03-06.

[10] "BAS5: ROS og caseanalyser " (power point presentation that summarizes the experiences). FFI - Forsvarets forskningsinstitutt, 2006-03-06.

[11] Basin, D., Doser, J., and Lodderstedt, T., "Model Driven Security for Process-Oriented Systems", in Proc. 8th ACM symposium on Access control models and technologies (SACMAT'03), pp. 100-110, 2003.

[12] Basin, D., Doser, J., and Lodderstedt, T., "Model Driven Security: from UML Models to Access Control Infrastructures", *ACM Transactions on Software Engineering and Methodology*, vol. 15 (1), pp. 39-91, 2006.

[13] Baskerville, R. L., "Investigating Information Systems with Action Research", *Communications of the Association for Information Systems*, vol. 2 (19), 1999.

[14] Becker, R. A., Eick, S. G., and Wilks, A. R., "Visualizing Network Data", *IEEE Transactions on Visualization and Computer Graphics*, vol. 1 (1), pp. 16-21, 1995.

[15] Bell, R., "Introduction to IEC 61508", in Proc. 10th Australian Workshop on Safety Critical Systems and Software (SCS'05), Conferences in Research and Practice in Information Technology, (55), pp. 3-12, 2005.

[16] Bistarelli, S., Fioravanti, F., and Peretti, P., "Defense tree for economic evaluations of security investment", in Proc. 1st Int. Conference on Availability, Reliability and Security (ARES'06), pp. 416-423, 2006.

[17] Bouti , A. and Kadi , A. D., "A state-of-the-art review of FMEA/FMECA", *International Journal of Reliability, Quality and Safety Engineering*, vol. 1, pp. 515-543, 1994.

[18] Bratthall, L. and Wohlin, C., "Is it Possible to Decorate Graphical Software Design and Architecture Models with Qualitative Information? - An Experiment", *IEEE Transactions on Software Engineering*, vol. 28 (12), pp. 1181-1193, 2002.

[19] Britton, C. and Jones, S., "The Untrained Eye: How Languages for Software Specification Support Understanding in Untrained Users", *Human Computer Interaction*, vol. 14 (1&2), pp. 191-244, 1999.

[20] Brændeland, G. and Dahl, H., "Evaluation of the method and tool used during the 10th SECURIS field trial" (Statnett). Memo, SINTEF ICT, 2007.

[21] Brændeland, G., Dahl, H., Engan, I., and Stølen, K., Using Dependent CORAS Diagrams to Analyse Mutual Dependency, in *Proc. of CRITIS'07 (to be published)*, 2007.

[22] Brændeland, G., Hogganvik, I., and Engan, I., "Evaluation of the methodology and tool used during the 7th SECURIS field trial" (Hydro CSI). Tech.rep., STF90 A0631, SINTEF ICT, 2005.

[23] Brændeland, G., Hogganvik, I., and Seehusen, F., "Evaluation of risk analysis methodology and tools used during the 4th SECURIS field trial" (DNV). Tech.rep., STF90 F04081, SINTEF ICT, 2004.

[24] Brændeland, G., Hogganvik, I., and Seehusen, F., "Evaluation of the methodology and tool used during the 5th SECURIS field trial" (NetCom). Tech.rep., STF90 F05400, SINTEF ICT, 2005.

[25] BS7799-1, Information Technology - Code of practice for information security management (since 2000 adopted by ISO/IEC17799) British Standards Institute (BSI), 1995.

[26] BS7799-2, Information Security Management Systems - Specification with guidance for use (replaced by ISO27001): British Standards Institute (BSI), 1999.

[27] Cahill, M.-C. and Carter, R. C. J., "Color code size for searching displays of different density", *Human Factors*, vol. 18 (3), pp. 273-280, 1976.

[28] Cavanagh, J. P., "Relationship between the immediate memory span and the memory search rate", *Psychological Review*, vol. 79, pp. 525-530, 1972.

[29] Chattratichart, J. and Kuljis, J., "An Assessment of Visual Representations for the 'Flow of Control'", in Proc. 12th Workshop of the Psychology of Programming Interest Group (PPIG'00), pp. 45-48, 2000.

[30] Christ, R. E., Research for evaluating visual display codes: an emphasis on colour coding., in *Information Design: The design and evaluation of signs and printed material*, R. Easterby and H. Zwaga, Eds.: John Wiley and Sons Ltd, 1984, pp. 209-228.

[31] Christ, R. E., "Review and analysis of color coding research for visual displays", *Human Factors*, vol. 17 (6), pp. 542-570, 1975.

[32] Cleveland, W. S. and McGill, R., "Graphical perception and graphical methods for analyzing scientific data." *Science*, vol. 229, pp. 828-833, 1985.

[33] "Common Criteria for Information Technology Security Evaluation v3.1", CCMB-2006-09-001, 2006.

[34] Cortellessa, V., Goseva-Popstojanova, K., Appukkutty, K., Guedem, A., Hassan, A. E., Elnaggar, R., Abdelmoez, W., and Ammar, H. H., "Model-based performance risk analysis", *IEEE Transactions on Software Engineering*, vol. 31 (1), pp. 3-20, 2005.

[35] CSI/FBI, "COMPUTER CRIME AND SECURITY SURVEY", 2005.

[36] Dahl, H. E. I., Hogganvik, I., and Stølen, K., "Structured Semantics for the CORAS Security Risk Modelling Language." in Proc. 2nd Int. Workshop on Interoperability Solutions on Trust, Security, Policies and QoS for Enhanced Systems (IS-TSPQ'07), pp. 79-92, 2007.

[37] Davison, R., Martinsons, M. G., and Kock, N., "Principles of canonical action research", *Information Systems Journal*, vol. 14 (1), pp. 65-86, 2004.

[38] Demarco, T. and Plauger, P. J., *Structured Analysis and Systems Specification.* : Prentice-Hall, 1979.

[39] Doan, T., Demurjian, S., Ting, T. C., and Ketterl, A., "MAC and UML for Secure Software Design", in Proc. ACM Workshop on Formal Methods in Security Engineering (FMSE'04), pp. 75-85, 2004.

[40] Engan, I., Lund, M. S., and Torsbakken, S., "Evaluation of the method and tool used during the 8th SECURIS field trial" (Hydro ICT). Memo, SINTEF ICT, 2007.

[41] Fenton, N., Krause, P., and Neil, M., "Software Measurement: Uncertainty and Causal Modeling", *IEEE Software*, vol. 19 (4), pp. 116-122, 2002.

[42] Fenton, N. and Neil, M., Combining evidence in Risk Analysis using Bayesian Networks: Agena White Paper, W0704/01, v01.01, 2004.

[43] Fenton, N. and Ohlsson, N., "Quantitative Analysis of Faults and Failures in a Complex Software System", *IEEE Transactions on Software Engineering*, vol. 26 (8), pp. 797-814, 2000.

[44] Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., and Chandramouli, R., "Proposed NIST standard for role-based acess control", *ACM Transactions on Information System Security (TISSEC)*, vol. 4 (3), pp. 224-274, 2001.

[45] Freeman, R. E., *Strategic Management: A Stakeholder Approach*: Ballinger Publishing, 1984.

[46] Gane, C. and Sarson, T., *Structured Systems Analysis: Tools and Techniques*: Mcdonnell Douglas Infomation, 1977.

[47] Genrich, H. J., "Predicate/transition nets", in *Advances in Petri Nets* vol. LNCS 254-255, W. Brauer, W. Resig, and G. Rozenberg, Eds.: Springer-Verlag, 1987, pp. 207-247.

[48] Georg, G., France, R., and Ray, I., "An Aspect-Based Approach to Modeling Security Concerns", in Proc. CSDUML'02, pp. 107-120, 2002.

[49] Goodman, N., *Languages of Art: An Approach to a Theory of Symbols*. Indianapolis: Hackett, 1976.

[50] Goseva-Popstojanova, K., Hassan, A. E., Guedem, A., Abdelmoez, W., Nassar, D. E. M., Ammar, H. H., and Mili, A., "Architectural-level risk analysis using UML", *IEEE Transactions on Software Engineering*, vol. 29 (10), pp. 946-960, 2003.

[51] Green, T., "Cognitive dimensions of notations", in Proc. Human Computer Interaction (HCI'89), People and Computers V, pp. 443-459, 1989.

[52] Green, T., "Describing information artefacts with cognitive dimensions and structure maps", in Proc. Human Computer Interaction (HCI'91), People and Computers VI, pp. 297-317, 1991.

[53] Hassan, A. E., Goseva-Popstojanova, K., and Ammar, H. H., "UML Based Severity Analysis Methodology", in Proc. RAMS'05, pp. 158-164, 2005.

[54] HB231, Information security risk management guidelines: Standards Australia/Standards New Zealand, 2000.

[55] HB231, Information security risk management guidelines: Standards Australia/Standards New Zealand, 2004.

[56] Hevner, A. R., March, S. T., Park, J., and Ram, S., "Design Science in Information Systems Research", *MIS Quarterly*, vol. 28 (1), pp. 75-105, 2004.

[57] Hogganvik, I., Lund, M. S., and Stølen, K., "Quality Evaluation of the CORAS UML profile", SINTEF ICT, Tech.rep. SINTEF A2199, 2007.

[58] Hogganvik, I., Seehusen, F., and Solhaug, B., "Evaluations of the methodology and tool used during the 6th SECURIS field trial" (Statnett). Tech.rep., STF90 A06032, SINTEF ICT, 2005.

[59] Howard, M. and LeBlanc, D., *Writing Secure Code*, 2 ed: Microsoft Press, 2003.

[60] Howard, M. and Lipner, S., *The Security Development Lifecycle*: Microsoft Press, 2006.

[61] Howard, R. A., *Dynamic Probabilistic Systems: Volume 1: Markov models*: John Wiley & Sons, 1971.

[62] Humprey, A., "SWOT - Strengths, Weaknesses, Opportunities, Threats" Standford University, 1960-1970.

[63] IEC16085, Information technology - Software life cycle processes - Risk management, 2004.

[64] IEC60300-3-9, Dependability management - Part 3: Application guide - Section 9: Risk analysis of technological systems - Event Tree Analysis (ETA), 1995.

[65] IEC60812, Analysis techniques for system reliability - Procedures for failure mode and effect analysis (FMEA and FMECA), 1985.

[66] IEC61025, Fault Tree Analysis (FTA), 1990.

[67] IEC61078, Analysis techniques for dependability - Reliability block diagram method, 1991.

[68] IEC61165, Application of Markov techniques, 1995.

[69] IEC61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related (E/E/PE) Systems: IEC, 1998-2000.

[70] IEC61882, Hazard and operability studies (HAZOP studies) - Application guide, 2001.

[71] ISO9001, Quality management systems – Requirements, 2000.

[72] ISO14001, Environmental management systems – Requirements with guidance for use, 2004.

[73] ISO15408, Information technology – Security techniques – Evaluation criteria for IT security (Common Criteria), 2005.

[74] ISO21827, Information technology – Systems Security Engineering – Capability Maturity Model, 2002.

[75] ISO27001, Information technology – Security techniques – Information security management systems – Requirements, 2005.

[76] ISO/IEC13335-1, Information technology – Security techniques – Management of information and communications technology security - Part 1:Concepts and models for information and communications technology security management, 2004.

[77] ISO/IEC13335-3, Information technology – Guidelines for the management of IT Security – Part 3: Techniques for the management of IT Security, 1998.

[78] ISO/IEC13335-4, Information technology – Guidelines for the management of IT Security – Part 4: Selection of safeguards, 2000.

[79] ISO/IEC13335-5, Information technology – Guidelines for the management of IT Security – Part 5: Management guidance on network security, 2001.

[80] ISO/IEC14516, Information technology – Security techniques – Guidelines for the use and management of Trusted Third Party services, 2002.

[81] ISO/IEC14977, Information technology – Syntactic metalanguage – Extended Backus-Naur Form (EBNF), 1996.

[82] ISO/IEC17799, Information technology – Security techniques – Code of practice for information security management (ISO27002), 2005.

[83] ISO/IEC18028, Information technology – Security techniques – IT network security, 2005-2006.

[84] ISO/IEC20000, Information technology – Service management (Part 1 and 2), 2005.

[85] ISO/IECTR18044, Information technology – Security techniques – Information security incident management, 2004.

[86] ITU-T X.800: Security architecture for open system interconnection for CCITT applications. (Technically aligned with ISO 7498-2), 1991.

[87] Johnson, W. G., *MORT Safety Assurance Systems*: Marcel Dekker, 1980.

[88] Jubis, R. M. T., "Coding effects on performance in a process control task with uniparameter and multiparameter displays", *Human Factors*, vol. 32 (3), pp. 287-297, 1990.

[89] Jürjens, J., *Secure Systems Development with UML*: Springer, 2005.

[90] Jürjens, J., "Sound Methods and Effective Tools for Model-based Security Engineering with UML", in Proc. Int. Conference on Software Engineering (ICSE'05), pp. 322-331, 2005.

[91] Jürjens, J., "UMLsec: Extending UML for secure systems development", in Proc. UML 2002 - The Unified Modeling Language, (LNCS 2460), pp. 412-425, 2002.

[92] Jøsang, A., Bradley, D., and Knapskog, S. J., "Belief-Based Risk Analysis", in Proc. Australasian Information Security Workshop 2004 (AISW'04), (32), pp. 63-68, 2004.

[93] Kavianian, H. R., Rao, J. K., and Brown, G. V., "Application of Hazard Evaluation Techniques to the Design of Potentially Hazardous Industrial Chemical Processes", U.S. National Institute for Occupational Safety and Health, 1992.

[94] Kemeny, J. G. and Snell, J. L., *Finite Markov chains*: Springer-Verlag, 1976.

[95] Kemmerer, R. A., "Cybersecurity", in Proc. Int. Conference on Software Engineering (ICSE'02), pp. 705-717, 2002.

[96] Kjellén, U., Tinmannsvik, R. K., Ulleberg, T., Olsen, P. E., and Saxvik, B., *SMORT - Sikkerhetsanalyse av industriell organisasjon (Norwegian only)*: Yrkeslitteratur, 1987.

[97] Kontio, J., Software Engineering Risk Management: A Method, Improvement Framework, and Empirical Evaluation, in *Dept. of Computer Science and Engineering*: Helsinki University of Technology, 2001.

[98] Krause, P. and Clark, D., *Representing Uncertain Knowledge: An Artificial Intelligence Approach*: Intellect Press, 1993.

[99] Krogstie, J., Conceptual Modeling for Computerized Information Systems Support in Organizations, in *Faculty of Electrical Engineering and Computer Science*: The Norwegian Institute of Technology, The University of Trondheim, 1995.

[100] Krogstie, J., "Evaluating UML Using a Generic Quality Framework", in *UML and the Unified Process*, L. Favre, Ed.: IRM Press, 2003, pp. 1-22.

[101] Krogstie, J., "Using a semiotic framework to evaluate UML for the development of models of high quality", in *Unified Modeling Language: Systems analysis, design, and development issues*, K. Siau and T. Halpin, Eds.: Idea Group Publishing, 2001, pp. 89-106.

[102] Krogstie, J. and Arnesen, S. d. F., "Assessing Enterprise Modeling Languages Using a Generic Quality Framework", in *Information Modeling Methods and Methodologies*: Idea Group, 2005, pp. 63-79.

[103] Krogstie, J., Lindland, O. I., and Sindre, G., "Defining quality aspects for conceptual models", in Proc. IFIP8.1 Working Conference on Information Systems Concepts (ISCO3), Towards a Consolidation of Views, pp. 216-231, 1995.

[104] Krogstie, J. and Sølvberg, A., *Information Systems Engineering: Conceptual Modeling in a Quality Perspective*: Kompendiumforlaget, 2003.

[105] Kuzniarz, L., Staron, M., and Wohlin, C., "An Empirical Study on Using Stereotypes to Improve Understanding of UML Models", in Proc. 12th Int. Workshop on Program Comprehension (IWPC'04), pp. 14-23, 2004.

[106] Larkin, J. H. and Simon, H. A., "Why a Diagram Is (Sometimes) Worth Ten Thousand Words", *Cognitive Science*, vol. 11, pp. 65-99, 1987.

[107] Lees, F. P. and Mannan, S., *Lees' loss prevention in the process industries : hazard identification, assessment and control*, vol. 1, 3rd ed: Elsevier Butterworth-Heinemann, 2005.

[108] Lindley, D. V., *Introduction to Probability and Statistics from a Bayesian Viewpoint*: Cambridge University Press, 1965.

[109] Linos, P. K., Aubet, P., Dumas, L., Helleboid, Y., Lejeune, D., and Tulula, P., "Visualizing program dependencies: An experimental study", *Software Practice and Experience*, vol. 24 (4), pp. 387-403, 1994.

[110] Lodderstedt, T., Basin, D., and Doser, J., "SecureUML: A UML-Based Modeling Language for Model-Driven Security", in Proc. UML'02, LNCS, (2460), pp. 426-441, 2002.

[111] Lund, M. S., den Braber, F., Stølen, K., and Vraalsen, F., "A UML profile for the identification and analysis of security risks during structured brainstorming." SINTEF ICT, Tech.rep. STF40 A03067, 2004.

[112] Lund, M. S., Hogganvik, I., Seehusen, F., and Stølen, K., "UML profile for security assessment", SINTEF ICT, Technical report STF40 A03066, 2003.

[113] March, S. T. and Smith, G. F., "Design and natural science research on information technology", *Design Support Systems, Elsevier Science*, vol. 15, pp. 251-266, 1995.

[114] Martin, N., "Simple Technique for Illustrating Risk (STIR)", SANS Institute, 2002.

[115] McDermott, J., "Abuse-case-based assurance arguments", in Proc. 17th Computer Security Applications Conference (ACSAC'O1), pp. 366-374, 2001.

[116] McDermott, J., "Visual Security Protocol Modeling", in Proc. New Security Paradigms Workshop (NSPW'05), pp. 97-109, 2005.

[117] McDermott, J. and Fox, C., "Using abuse case models for security requirements analysis", in Proc. 15th Computer Security Applications Conference (ACSAC'99), pp. 55-66, 1999.

[118] McGrath, J. E., *Groups: interaction and performance*: Prentice-Hall, 1984.

[119] MEHARI (Méthode Harmonisée d'Analyse de Risques Informatiques): CLUSIF (Club de la Sécurité de l'Information Français), 2004.

[120] Microsoft, "The Security Risk Management Guideline", Microsoft Solutions for Security and Compliance, Microsoft Security Centre of Excellence, 2006.

[121] Myagmar, S., Lee, A. J., and Yurcik, W., "Threat Modeling as a Basis for Security Requirements", in Proc. Symposium on Requirements Engineering for Information Security (SREIS'05), 2005.

[122] Nielsen, D. S., *The Cause/Consequence Diagram Method as a Basis for Quantitative Accident Analysis*: Danish Atomic Energy Commission, RISO-M-1374, 1971.

[123] NIST, "Risk Management Guide for Information Technology Systems", U.S. National Institute of Standards and Technology (NIST), NIST Special Publication SP800-30, 2002.

[124] OMG, "UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms", Object Management Group, 2006.

[125] OMG, "Unified Modeling Language (UML): Superstructure, version 2.0", Object Management Group, 2005.

[126] Pauli, J. and Xu, D., "Threat-driven architectural design of secure information systems", in Proc. 7th International Conference on Enterprise Information Systems (ICEIS'05), pp. 136-143, 2005.

[127] Poulsen, K., Sluggish movement on power grid cyber security, in *SecurityFocus* http://www.securityfocus.com/news/9328. 2004-08-13.

[128] PricewaterhouseCoopers, "The Information Security Breaches Survey 2006", UK Department of Trade and Industry, 2006.

[129] Rausand, M. and Høyland, A., *System reliability Theory: Models, Statistical Methods, and Applications*, 2 ed: Wiley, 2004.

[130] Ray, I., Li, N., France, R., and Kim, D.-K., "Using UML To Visualize Role-Based Access Control Constraints", in Proc. SACMAT'04, pp. 115-124, 2004.

[131] Redmill, F., Chudleigh, M., and Catmur, J., *HAZOP and Software HAZOP*: Wiley, 1999.

[132] Refsdal, A. and Solhaug, B., "Evaluations of the methodology and tool used during the 9th SECURIS field trial" (FLO/IKT). Tech.rep., SINTEF A1532, SINTEF ICT, 2007.

[133] *Risk and Reliability - An Introductory Text*, 5 ed: Risk & Reliability Associates (R2A), 2005.

[134] Rumbaugh, J., Jacobson, I., and Booch, G., *The Unified Modeling Language Reference Manual*: Addison Wesley Longman, Inc., 1998.

[135] Sampson, G., *Writing Systems*: Hutchinson, 1985.

[136] Savage, L. J., *The Foundations of Statistical Inference*: J. Wiley, 1962.

[137] Schneider, W. and Shiffren, R. M., "Controlled and automatic human information processing I: Detection, search, and attention." *Psychological Review*, vol. 84, pp. 1-66, 1977.

[138] Schneier, B., "Attack trees: Modeling security threats", *Dr. Dobb's Journal*, vol. 24 (12), pp. 21-29, 1999.

[139] Schneier, B., *Secrets & Lies: Digital Security in a Networked World*: John Wiley & Sons, 2000.

[140] Sedra, A. S. and Smith, K. C., *Microelectronic Circuits*: Oxford University Press, 2003.

[141] Seehusen, F. and Stølen, K., "Graphical specification of dynamic network structure." in Proc. 7th Int. Conference on Enterprise Information Systems (ICEIS'05), (3), pp. 203-209, 2005.

[142] Shneiderman, B., *Designing the User Interface*: Addison-Wesley, 1992.

[143] Siegel, S. and Castellan, J., *Non-parametric Statistics for the Behavioural Sciences*, 2 ed: McGraw-Hill International Editions, 1988.

[144] Simon, H. A., *The Sciences of the Artificial*, 3 ed: MIT Press, 1996.

[145] Sindre, G. and Opdahl, A. L., "Capturing Security Requirements through Misuse Cases", in Proc. 14th Norwegian Informatics Conference (NIK'2001), pp. 219-230, 2001.

[146] Sindre, G. and Opdahl, A. L., "Eliciting Security Requirements by Misuse Cases", in Proc. TOOLS-PACIFIC, pp. 120-131, 2000.

[147] Sindre, G. and Opdahl, A. L., "Templates for Misuse Case Description", in Proc. Workshop of Requirements Engineering: Foundation of Software Quality (REFSQ'01), pp. 125-136, 2001.

[148] Smith, L. and Thomas, D., "Color versus shape coding in information displays", *Journal of Applied Psychology*, vol. 48 (3), pp. 137-146, 1964.

[149] Song, H. and Leangsuksun, C., "A Framework for Cluster Availability Specification and Evaluation", in Proc. 43th ACM Southeast Conference (Student poster in "Architecture and distributed system session"), pp. 202-203, 2005.

[150] Spiegelhalter, D. J., "Probabilistic reasoning in predictive expert systems", in Proc. 2nd Annual Conference on Uncertainty in Artificial Intelligence (UAI'86), pp. 47-67, 1986.

[151] Stølen, K. and Solheim, I., "Technology Research Explained", SINTEF ICT, Tech.rep. A313, 2006.

[152] Stålhane, T. and Wedde, K. J., "Practical experience with the application of HazOp to a software intensive system", in Proc. Joint ESCOM and ENCRESS Conference, pp. 271-281, 1998.

[153] Swiderski, F. and Snyder, W., *Threat Modeling*: Microsoft Press, 2004.

[154] Tamassia, R., Di Battista, G., and Batini, C., "Automatic Graph Drawing and Readability of Diagrams." *IEEE Transactions on Systems, Man, and Cybernetics.*, vol. 18 (1), pp. 61-79, 1988.

[155] Tan, K. C., Effects of Stimulus Class on Short -Term Memory Workload in Complex Information Displays, in *Department of Industrial Engineering and Operations Research*: Virginia Technical University, 1990.

[156] Treisman, A. and Gormican, S., "Feature analysis in early vision: Evidence from search asymmetries", *Psychological Review*, vol. 95 (1), pp. 15-48, 1988.

[157] Vetterling, M., Wimmel, G., and Wisspeintner, A., "Secure Systems Development Based on the Common Criteria: The PaIME Project", in Proc. 10th ACM SIGSOFT symposium on Foundations of software engineering (SIGSOFT'02/FSE-10), pp. 129-138, 2002.

[158] Vorster, A. and Labuschagne, L., "A Framework for Comparing Different Information Security Risk Analysis Methodologies", in Proc. SAICSIT'05, ACM International Conference Proceeding Series, (150), pp. 95-103, 2005.

[159] Walpole, R. E., Myers, R. H., and Myers, S. L., *Probability and Statistics for Engineers and Scientists*, 6th ed: Prentice Hall International, 1998.

[160] Ware, C., *Information Visualization: Perception for Design*, 2nd ed: Elsevier, 2004.

[161] Wertheimer, M., *Laws of Organization in Peceptual Forms [English Translation of: "Untersuchungen zur Lehre von der Gestalt", II, Psychologische Forschung 4 (1923), pp. 301 –350.] In Willis D. Ellis (ed.): A Source Book of Gestalt Psychology*: Routledge & Kegan Paul, 1923.

[162] West, S. and Andrews, A. D., "OCTAVE-Best Practices Comparative Analysis", Prepared for U.S. Army Medical Research and Materiel Command, ATI IPT Technical Report 03-4, 2003.

[163] Wickens, C. D., *Engineering Psychology and Human Performance*, 2 ed: HarperCollins, 1992.

[164] Winn, W., "An Account of How Readers Search for Information in Diagrams", *Contemporary Educational Psychology*, vol. 18, pp. 162-185, 1993.

[165] Winn, W., "Perceptual strategies used with flow diagrams having normal and unanticipated formats", *Perceptual and Motor Skills*, vol. 57, pp. 751-762, 1983.

[166] Winn, W., "The role of diagrammatic representation in learning sequences, identification, and classification as a function of verbal and spatial ability", *Journal of Research in Science Teaching*, vol. 19, pp. 79-89, 1982.

[167] Winn, W. and Solomon, C., "The effect of the rhetorical structure of diagrams on the interpretation of simple sentences", *University of Washington, unpublished manuscript*, 1991.

[168] Winograd, T. and Flores, F., *Understanding Computers and Cognition*: Addison-Wesley Professional, 1987.

[169] Xu, D. and Nygard, K., "A Threat-Driven Approach to Modeling aand Verifying Secure Software", in Proc. Int. Conference on Automated Software Engineering (ASE'05), pp. 342-346, 2005.

Table of Figures

List of Tables