



SINTEF REPORT

SINTEF ICT

Address: P.O.Box 124, Blindern
0314 Oslo NORWAY
Location: Forskningsveien 1
0373 Oslo
Telephone: +47 22 06 73 00
Fax: +47 22 06 73 50
Enterprise No.: NO 948 007 029 MVA

TITLE

Investigating Preferences in Graphical Risk Modeling

AUTHOR(S)

Ida Hogganvik and Ketil Stølen

CLIENT(S)

Norges Forskningsråd / The Research Council of Norway

REPORT NO. SINTEF A57	CLASSIFICATION Unrestricted	CLIENTS REF. NFR152839/220	
CLASS. THIS PAGE Open	ISBN	PROJECT NO. 40332800	NO. OF PAGES/APPENDICES 48/3
ELECTRONIC FILE CODE		PROJECT MANAGER (NAME, SIGN.) Ketil Stølen	CHECKED BY (NAME, SIGN.) Iselin Engan 
FILE CODE	DATE 2007-01-11	APPROVED BY (NAME, POSITION, SIGN.) Bjørn Skjellaug, Research Director 	

ABSTRACT

In a security analysis it is often helpful to draw diagrams to illustrate threat and risk scenarios. To ensure the effectiveness of such diagrams, it is essential that they are easily understood by people without training and experience in modeling and security analysis. In this report we present an empirical investigation of the risk modeling preferences among professionals and students in software engineering. The objective of the investigation was to identify the preferred way of refining an existing diagrammatic security risk modeling language without making it more difficult to understand. Our empirical investigation showed that mechanisms like size- and color coding used for conveying particular information in graphical models are less preferred by the subjects compared to textual information labels. The size or color of an element does not in general have any unique interpretation in a diagram, while textual information is more specific and self-explaining. The conclusion is that the subjects tend to prefer the representations where they get the most information without requiring them to interpret any additional graphical means.

KEYWORDS	ENGLISH	NORWEGIAN
GROUP 1	Risk modeling	Risikomodellering
GROUP 2	Empirical investigation	Empirisk undersøkelse
SELECTED BY AUTHOR	Security analysis	Sikkerhetsanalyse
	Graphical means	Grafiske virkemidler

TABLE OF CONTENTS

1	INTRODUCTION	1
2	BACKGROUND AND MOTIVATION.....	3
3	MEANS AND MECHANISMS TO EASE COMPREHENSION.....	5
3.1	REPRESENTING GRAPH NAVIGATION	5
3.1.1	<i>Set-up for “representing graph navigation”</i>	<i>6</i>
3.2	REPRESENTING VULNERABILITIES	7
3.2.1	<i>Set-up for “representing vulnerabilities”</i>	<i>8</i>
3.3	REPRESENTING RISKS	8
3.3.1	<i>Set-up for “representing risks”</i>	<i>10</i>
4	EXPERIMENT DESIGN	11
4.1	SUBJECTS	11
4.2	MATERIAL.....	11
4.3	HYPOTHESES	12
4.4	ANALYSIS METHOD	12
5	RESULTS	13
6	DISCUSSION OF MODELING PREFERENCES.....	15
6.1	REPRESENTING GRAPH NAVIGATION	15
6.2	REPRESENTING VULNERABILITIES	15
6.3	REPRESENTING RISKS	16
7	THREATS TO VALIDITY.....	17
8	CONCLUSIONS	18
8.1	CONCLUSIONS REGARDING “REPRESENTING GRAPH NAVIGATION”	18
8.2	CONCLUSIONS REGARDING “REPRESENTING VULNERABILITIES”	18
8.3	CONCLUSIONS REGARDING “REPRESENTING RISK”	19
8.4	HOW THE RESULTS IMPACTED THE CORAS LANGUAGE.....	19
	ACKNOWLEDGEMENTS	20
	REFERENCES	22
	LIST OF TABLES.....	25
	LIST OF FIGURES.....	25
	APPENDIX A – RELATED MODELING NOTATIONS.....	26
	APPENDIX B – EXPERIMENT MATERIAL.....	28
B.I	MODELING QUESTIONNAIRE.....	28
B.II	TASKS THAT WERE LEFT OUT OF THE ANALYSIS	34
	APPENDIX C – PRESTUDY	36
C.I	PRESTUDY DESIGN	36
C.II	PRESTUDY MATERIAL.....	36
C.III	PRESTUDY RESULTS AND DISCUSSION	40
C.IV	THREATS TO VALIDITY FOR THE PRESTUDY	43
C.V	CONCLUSION.....	43



1 Introduction

As security risk analysts we often face clients with problems like the following: “we’ve installed a new system, how does it affect our security?”, “the system has existed for a long time and undergone several updates and changes, but how secure is it now?” or “we like to publish more sensitive information via our intra/extranet solution, but what are the risks?” To help responding to these questions we have developed a graphical modeling language supporting the identification, communication and documentation of security threats and risk scenarios called the CORAS security risk modeling language (the CORAS language)¹.

The CORAS language with its detailed user guidelines has been applied in several large industrial field trials [11], and the major decisions regarding its underlying foundation, notation and guidelines are supported by empirical investigations [17, 18]. In this report we present the results from a study of risk modeling preferences among professionals in software engineering. The objective of the study was to find ways to refine the CORAS language to enhance comprehension. From earlier studies of security analysis concepts and terminology we know that some concepts are more difficult to understand than others [18]. These studies lead to the hypothesis that the relations that were found easy to understand would also be most suitable and appropriate to represent in a simple and straight forward manner, while the difficult concepts would need more sophisticated representations. A concept for which we searched for a more sophisticated representation was *likelihood of threat scenario paths*. The second issue in the investigation on which this report presents, was how to represent *vulnerabilities* in a more explicit manner, something that was insufficient in the original version of the CORAS language [31-33]. The last issue was whether or not we could identify a way of representing risk, and especially the *severity of risks*, to quickly give the reader an overview of the overall risk picture. To solve these issues we tested various means for representation known from the field of information visualization [45], in addition to textual information labels.

The CORAS language is intended to be used during a security analysis by the analyst, and has different purposes in each phase of the analysis. A security analysis process typically includes five phases: (1) context establishment, (2) risk identification, (3) risk estimation, (4) risk evaluation and (5) treatment identification [2]. A frequently used technique in security analysis, particularly during risk identification, is structured brainstorming. It may be understood as a structured “walk-through” of the target of analysis. The technique is not limited to a specific type of target, but can be used to assess risks towards anything from simple IT systems to large computerized process control systems or manual maintenance procedures e.g. in nuclear power plants. This does not mean that exact same technique can be applied directly to all kinds of targets; it has to be adapted to fit the target domain. The main idea of a structured brainstorming in security analysis is that a group of people with different competences and backgrounds will view the target from different perspectives and therefore identify a larger number, and possibly other, risks than individuals or a more heterogeneous group. Gathering people with a broad knowledge and experience with vulnerabilities, threats, bugs and flaws is important. The input to a brainstorming session is various kinds of target models (e.g. UML models, informal descriptions, pictures etc.). The models are assessed in a stepwise and structured manner under the guidance of the security analysis leader. The identified risks are documented by the analysis secretary. Deciding which roles that should be represented at the brainstorming session is part of tailoring the brainstorming technique towards the target in question. In some cases an expert is only participating in the part of the analysis where his or her expertise is needed. This makes it essential that we have a comprehensive and simple manner for communicating and documenting risk specific information.

¹ <http://coras.sourceforge.net>

Documenting security risks graphically has many similarities with information modeling. Information modeling is used to describe the behavior of a workflow or process and risk modeling deals with describing the “workflow” or “process” of a threat that initiates an unwanted incident. During risk analyses we have experienced the need for describing *how* incidents can happen, *who* initiates them and *what* they affect. We also need to show the relationships between assets, stakeholders, treatments, risks and other elements. Our work has been to develop a modeling language that is both sufficiently expressive to satisfy the modeler’s needs, yet simple and intuitive.

The report is structured as follows: in Sect. 2 we give the background and motivation for our investigations. In Sect. 3 we introduce various techniques for information visualization that can be used in modeling, and describe the alternatives we have explored. Sect. 4 presents the experiment design and the results are given in Sect.5. Our findings are discussed in Sect. 6 and the identified threats to validity are reported in Sect.7. Sect. 8 describes how the findings impacted the CORAS language, and Sect. 9 brings the conclusions from our work. Appendix A gives an overview of related modeling approaches. Appendix B presents the experiment material, while Appendix C describes the prestudy that was conducted before the main experiment.

2 Background and motivation

The background for this investigation is partly based on previous findings regarding the conceptual foundation and also weaknesses of the previous version of the language [31-33]. In the following we describe this in more detail. Let us first explain the UML class diagram [34] in Figure 1 that presents the conceptual foundation. The model may be understood as a kind of abstract syntax for the CORAS language, and its definitions are taken from the following international standards:

- *Information technology: Guidelines for management of IT Security* (ISO/IEC13335) [21, 22]
- *Australian/New Zealand Standard for Risk Management* (AS/NZS4360) [2]
- *Information Security Risk Management Guidelines* (HB231) [15]

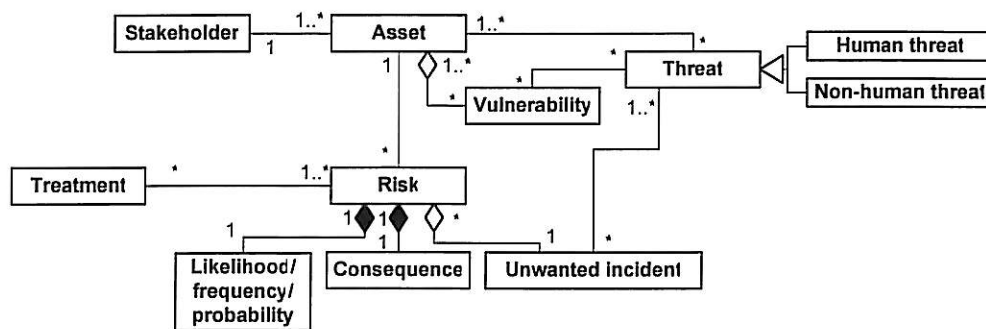


Figure 1 – The conceptual foundation

Figure 1 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 [2]. An **asset** is something to which a stakeholder directly assigns value and, hence, for which the stakeholder requires protection [15]. Assets are subject to **vulnerabilities**, which are weaknesses which can be exploited by one or more threats [22]. A **threat** is a potential cause of an unwanted incident [22]. A threat may be classified as human (with accidental origin or deliberate harmful intentions) or non-human (also called environmental) [22]. An **unwanted incident** is an event that may harm or reduce the value of assets and is something we want to prevent [21]. In [22] they use the more specialized term “information security incident” (any unexpected or unwanted event that might cause a compromise of business activities or information security.) A **risk** is the chance of something happening that will have an impact upon objectives (assets) [2]. 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 distinct 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** [2] (e.g. low, medium, high or other scales) which is based upon the estimated **likelihood** (a general description of frequency or probability [2]) for the unwanted incident to happen and its **consequence** in terms of damage to an asset. A **treatment** is the selection and implementation of appropriate options for dealing with risk [2].

A previous experiment [18] focusing on the understanding of this conceptual foundation included 57 subjects, professionals within software engineering as well as students. They responded to a questionnaire with questions in six categories, each targeting a risk related concept: *assets*, *treatments*, *frequency measures (freq.)*, *risks*, *vulnerabilities (vuln.)*, and *threats*. It was found that questions related to vulnerability and asset received more correct answers than e.g. the frequency measure category (Figure 2). This finding motivated us to investigate:

1. How should we visualize frequency measures in threat diagrams to ease graph navigation, especially the likelihood of single paths and combined paths?

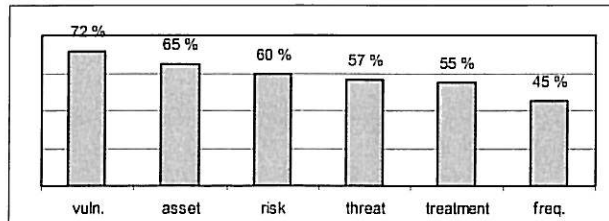


Figure 2 – Percentage of correct answers for each category (mean)

As mentioned, the first version of the CORAS language had some weaknesses, and among others it lacked a proper representation of vulnerabilities. Being able to specify vulnerabilities explicitly has been a requirement from the users of the CORAS language in field trials, but exactly how this should be done has not been clear. This made us investigate the following:

2. How should we visualize vulnerabilities? What is the preferred way among professionals within software engineering?

Both in empirical investigations [17, 18] and in field trials we have experienced that the concept “risk” belongs to our everyday vocabulary, but most people find it difficult to specify exactly what it is. The concept of risk is different from the other concepts we have studied in the sense that it is more abstract and is used to cover other, more concrete concepts. To make it less abstract we looked for the preferred way of representing it graphically. Hence, this motivated the investigation of:

3. How can we visualize risks in the models to improve the understanding of this abstract concept?

In the following we deal with each of the three issues listed above in separate subsections. The first is referred to as “representing graph navigation” (Sect.3.1, Sect.6.1 and Sect.8.1), the second as “representing vulnerabilities” (Sect.3.2, Sect.6.2 and Sect. 8.2) and the third as “representing risk” (Sect.3.3, Sect.6.3 and Sect.8.3).

3 Means and mechanisms to ease comprehension

The CORAS language is not a programming language, i.e. it is not to be executed on a computer. Nevertheless, it has been carefully designed to precisely model detailed sequences of events, and at the same time serve as a tool for capturing information from the creative brainstorming session. There exist many means and mechanism for helping us present information in a better way. Software engineering studies have shown that applying graphical means to program- and component specifications increases the understanding of the system [4], and one has successfully applied graphical means to visualize the program interdependencies [29].

A presupposition for understanding a CORAS diagram is some familiarity with the domain it describes. According to Winn [48] people that are familiar with the domain seek, encode, process and recall information in a diagram more effectively than those that are less expert. In addition to domain understanding, some knowledge of the semantics and syntax of the language is also required. Our aim was to implement means that reduced the latter to a minimum by adding helpful information in terms of symbols, textual information labels or other graphical means. 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.

In the following we motivate and present the means and mechanisms that were selected as subjects for investigation, structured according to the three main issues identified at the end of the previous section: *representing graph navigation*, *vulnerabilities* and *risk*. At the end of each subsection we present the diagram alternatives investigated in the experiment. We refer to the different tasks in the experiment using the following convention: “TE2” means Task 2 in the Experiment, “D1” means Diagram alternative 1, “TE2D1D2” means Diagram 1 compared to Diagram 2 in Experiment Task 2.

3.1 Representing graph navigation

As explained above, we refer to the process of understanding and reading CORAS diagrams as “graph navigation”. When describing the different paths via which a threat may choose to harm assets we often find it useful to draw special attention to the paths that they are more likely to follow. It was also desired by the users of the CORAS language 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 3.

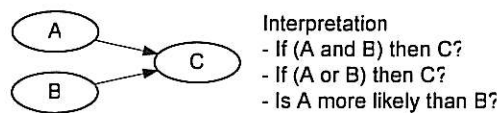


Figure 3 – Typical threat scenario situation

The first task was to find means that could help us express the likelihood of paths. The CORAS language 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 (Figure 4).

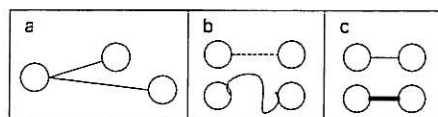


Figure 4 – Node-link configurations

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 [45]. 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 [44]. This may be a useful feature to include in the CORAS language. We chose to test thick, thin and dashed lines to illustrate more or less likely paths through the threat diagrams (Figure 6). The use of line variations was also inspired by network traffic maps which often use node-link diagrams to visualize network traffic. Typically one illustrates the load or type of link data using different line colors or varying the thickness of the lines [3]. 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 [46] 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) [20]. Fault trees let you specify dependencies between events in a tree structure using logical AND/OR operators called “gates”. One of our goals is to make the CORAS language fault tree compliant. In the CORAS language, there was no standard interpretation of two threat scenarios pointing to the same threat scenario or unwanted incident (Figure 3). 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. We decided to test the FTA-symbol for “AND” (Figure 5) which is also used in electrical circuit design [37]. We added an information label with the text “and” to the symbol to ease comprehension for people without background in FTA or circuit design. There exist a similar symbol for OR-gates, but we chose to test only one of them since they are very similar and our findings probably can cover both gate-symbols.

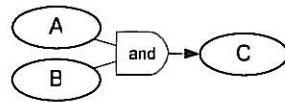


Figure 5 – The AND-gate symbol

In addition to the AND-gate symbol and dashed lines to show dependencies between combined paths, we tested a UML inspired “information note”. In UML one may attach notes to a diagram element to convey additional information about the specific element. The idea was to use the notes as an alternative to logical gates.

3.1.1 Set-up for “representing graph navigation”

The effect of two different line styles was tested in TE2 for visualizing paths that are more likely to be chosen by a threat (Figure 6):

- D1: dashed lines that are “weaker” than solid lines to illustrate less likely paths.
- D2: thick lines that are easily noticed to highlight the more likely paths in the graph.

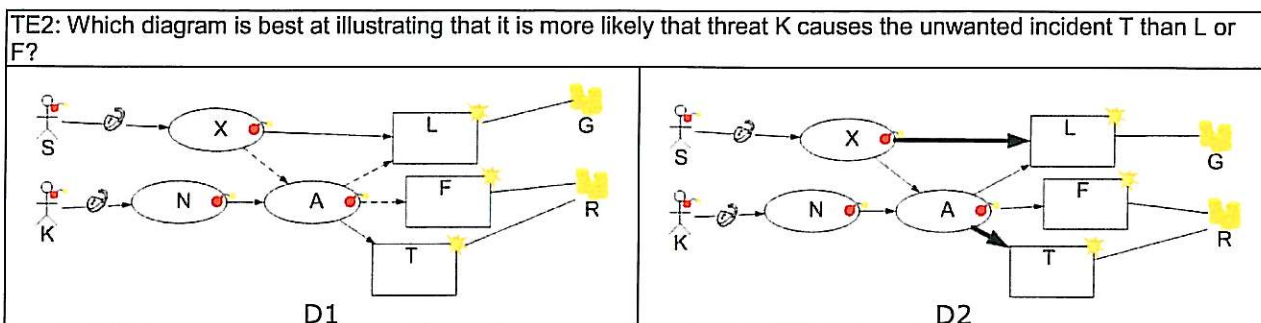


Figure 6 – The diagram alternatives for: “most likely path through a graph”

Three modeling alternatives for logical “AND” were tested in TE4 (Figure 7):

- D1: the special AND-gate symbol from fault tree notation and electrical circuit design, labeled with the text “and”, was used.
- D2: we aimed to give the two paths a special meaning by “sublightning” them, changing their appearance from a solid line to a dashed line to indicate a “special AND-dependency”.
- D3: we tested the effect of attaching UML-like notes to show the AND-dependency between the two paths.

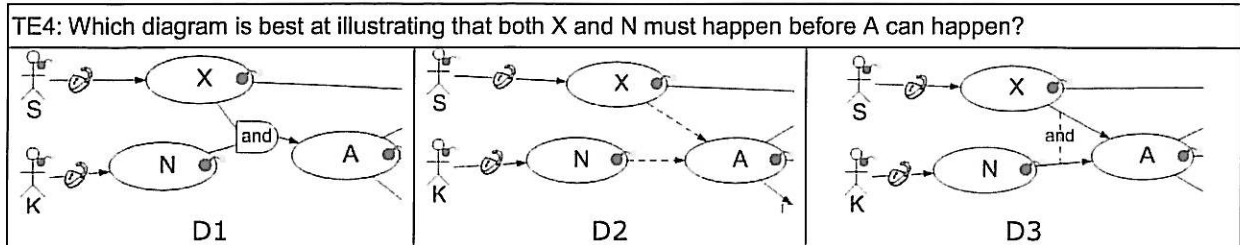


Figure 7 – The diagram alternatives for: “logical AND in graph paths”

3.2 Representing vulnerabilities

The original CORAS language 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 the CORAS language.

The way vulnerabilities was modeled in the original CORAS language 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 [28]. This is in accordance with the results from our experiment with graphical symbols where the textual stereotyping seemed to be overlooked [17]. In the same experiment we tested UML symbols vs. special symbols and found that the representation using special risk related symbols performed better than the one with conventional UML symbols. These finding suggests that carefully designed symbols are useful and that vulnerability should have a graphical symbol in addition to the text. We chose to use an *open chain lock symbol*, which is often used to symbolize a lack of security within the computer security domain. It is a simple and internationally known symbol. To achieve what [13] terms “syntactic disjointness”, the symbol is unique and it is neither 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. We compared this representation to the original CORAS language representation with respect to “shared vulnerabilities”.

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. As a consequence of this we decided to try modeling the vulnerabilities where they logically belong in the chain of events.

Winn [49, 50] 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 [51]. This is in accordance with the modeling style already used in the CORAS language. Threats always initiate the chain of events from the left and terminate at the right side of the diagram, where the assets are placed. This is also in accordance with one of the node-link rules [45] stating that “placing closed contours spatially in an ordered sequence can represent conceptual ordering of some kind” (Figure 4, 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) [7]. If the number of lines increases one may easily loose track of the path and repeated arrows help distinguishing between crossovers and connection points.

3.2.1 Set-up for “representing vulnerabilities”

Two alternative representations of shared vulnerabilities (common for two or more assets) were tested in TE5 (Figure 8):

- D1: we represented vulnerabilities with an open chain lock symbol and a vulnerability description. The vulnerabilities were located where they logically belonged in the sequence of events.
- D2: we linked the vulnerabilities closely to the assets they belonged to by specifying them as textual attributes of each asset. Since a vulnerability may yield for more than one asset, it may be listed under several assets.

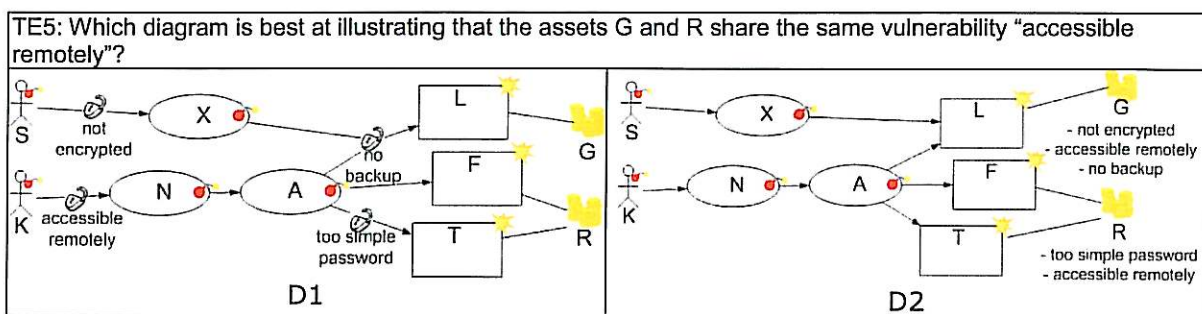


Figure 8 – The diagram alternatives for: “shared vulnerabilities”

The same alternatives were also tested in task 6 (TE6), but in the opposite order (TE6D1 = TE5D2 and TE6D2 = TE5D1).

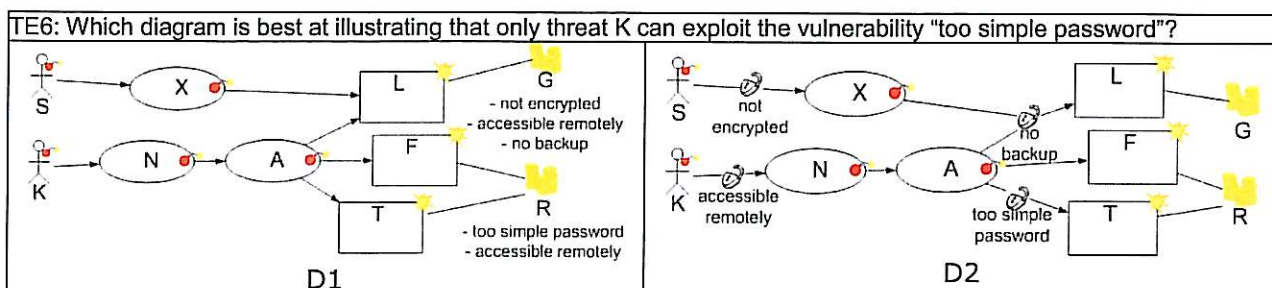


Figure 9 – The diagram alternatives for: “which vulnerabilities are exploited by which threats?”

3.3 Representing risks

Representing risk is a great challenge to its abstract nature. In the threat diagrams of the original CORAS language 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 which is especially useful during the final risk treatment phase. To visualize this issue, we investigated 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 10) and the size may be used to represent the magnitude of a concept (Figure 11). This is also inspired by the gestalt principle of breaking the pattern to attract attention. We want the reader to quickly discover the most serious unwanted incidents, since they often represent major risks. We therefore tested the effect of marking the most serious unwanted incident with a darker color (D1, Figure 13), since dark colors are more easily noticed against white background than light colors.



Figure 10 – Color

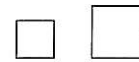


Figure 11 – Size

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 [5, 9, 10, 38, 45, 47]. In our case we only used gray and white. 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 [13]. 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 [8]. This top ranking of color has been confirmed by many other studies [25]. According to these results we 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. The color-alternative was compared to one using *size* to symbolize the most serious incident since large elements are more easily noticed than small ones (D2, Figure 13). 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 [45] (p. 182). An important aspect when using shapes is to avoid symbols that are of similar shapes. Shapes that are too similar have been found to increase search time and are therefore not recommended [42]. In our experiment material we only compared two sizes, whereas the number of size steps in a real diagram would most likely correspond to the three steps *low*, *medium* and *high risk* i.e. only one step more than in our material.

We also explored the use of textual information in the unwanted incident symbol (D3, Figure 13) and as an annotation to the risk association (D2, Figure 14). Our motivation for doing this was that studies of rapid processing of information [6, 35] have found letters and digits to be some of the best coding forms. These investigations involved studying short-time memory and how graphical means are memorized using Sternberg tasks². 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 [43], and digits, letters and words were found to represent less subjective workload than colors and shapes [43]. These findings support the modeling alternatives that use textual information labels to illustrate the seriousness of unwanted incidents and risks.

In [45] 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 12). Risk was one of the few relevant concepts for which the original CORAS language did not have a special symbol. We therefore investigated representing risks with a symbol and using size to represent its seriousness

² For a limited time the subject is shown a set of objects which has to be memorized.

(D3, Figure 14). We selected a traditional, red and white road sign that symbolizes “danger” that varies in size according to the severity of the risk.



Figure 12 – Shapes

3.3.1 Set-up for “representing risks”

We investigated which of size, color or textual information is better suited to represent the magnitude of an unwanted incident in TE3 (Figure 13):

- D1: we gave the most serious unwanted incident a darker color than the other incident.
- D2: we represented the most serious unwanted incident with a larger symbol.
- D3: we used textual information labels to specify the severity of the unwanted incidents.

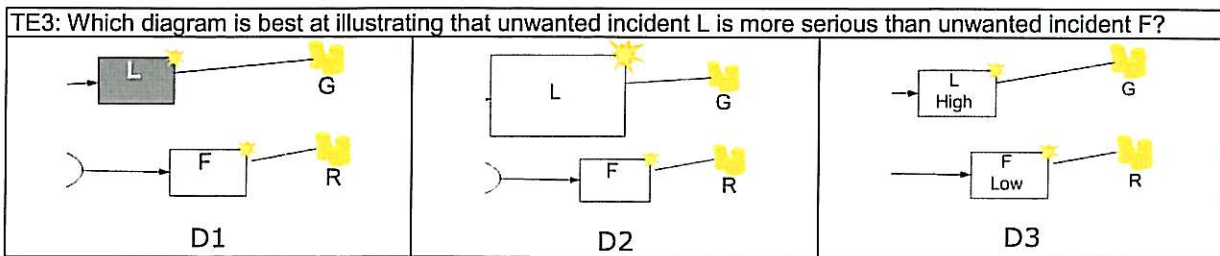


Figure 13 – The diagram alternatives for: “the magnitude of unwanted incidents”

In task 7 (TE7) we investigated the following modeling alternatives (Figure 14):

- D1: we used thick lines to illustrate the largest risks (discussed in Section 3.1).
- D2: we used textual information.
- D3: we tested a combination of symbol and size to illustrate both the concept of risk, and how to convey its seriousness.

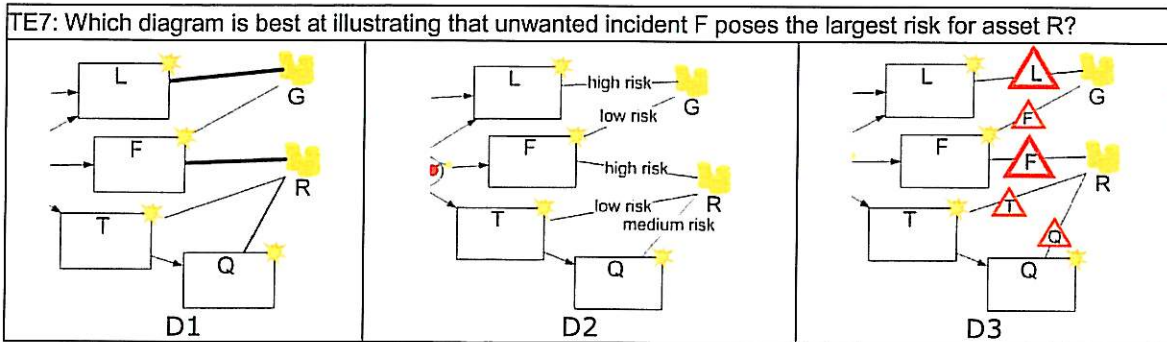


Figure 14 – The diagram alternatives for: “the magnitude of risks”

4 Experiment design

To decide upon the preferred notation we conducted an empirical study using professionals as subjects. Similar representations had been tested in a prestudy using students.

4.1 Subjects

The survey was distributed via e-mail to people in various parts of the software industry. The 33 subjects received no payment and participated voluntarily. The majority of the subject group was male between the age of 26 and 35 with 1-3 years of work experience. Work experience is shown in Figure 15. The term task experience includes the following task categories: *system design, development, testing, quality evaluation, maintenance, marketing/sale* and *security*. System security was added later in the experiment when we were interested in obtaining answers from a group of subjects that was less likely to have system development experience, but that were more experience with system security. The subjects' task experience is shown in Figure 16. The demographic data was not used for hypothesis testing, but rather to give a picture of the type of subjects that participated.

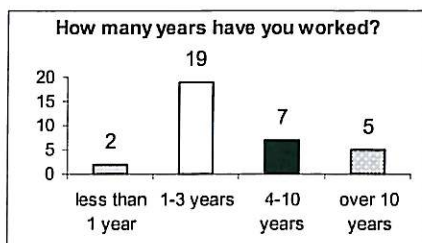


Figure 15 – The subjects' years of work experience

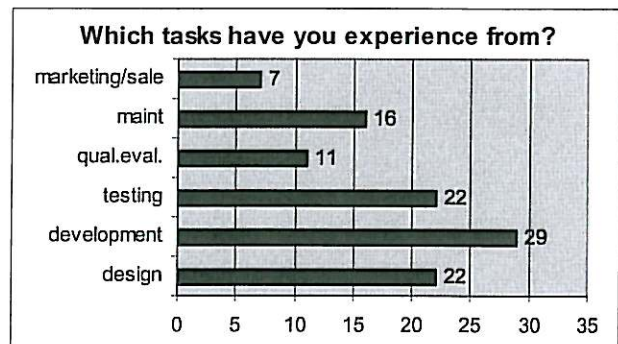


Figure 16 – The subjects' task experience

4.2 Material

The material was in the form of a questionnaire where the subjects were to prioritize between different modeling alternatives. It was similar to the prestudy using students, and a complete version is provided in Appendix B. It consisted of eight tasks and was estimated to take about 15-20 minutes to fill in. The tasks were related to variations of the basis threat diagram shown in Figure 17. The focus of each varied, but the main issues were how to simplify graph navigation, how to best represent vulnerabilities and how to highlight major risks and unwanted incidents. The explanation given in the material was as follows: *This questionnaire is about threat modeling. We present various ways of modeling the same situation, and your task is to prioritize between them and choose the diagram you think is best! This type of modeling is meant to support the risk analyst in situations where the participants in meetings have different backgrounds and expertise (technical and non-technical). The diagrams are meant to help them understand the system and each other better and faster. Compare two and two diagrams and choose the one you prefer by marking with an "X" in the table below the weight symbol, like shown in this example:*

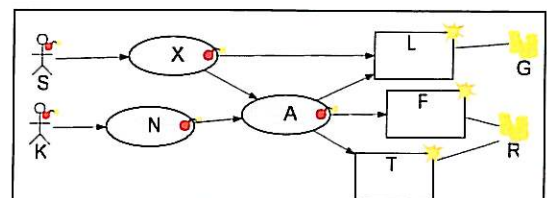


Figure 17 – The basic threat diagram

Which diagram do you prefer, diagram 1 or diagram 2?

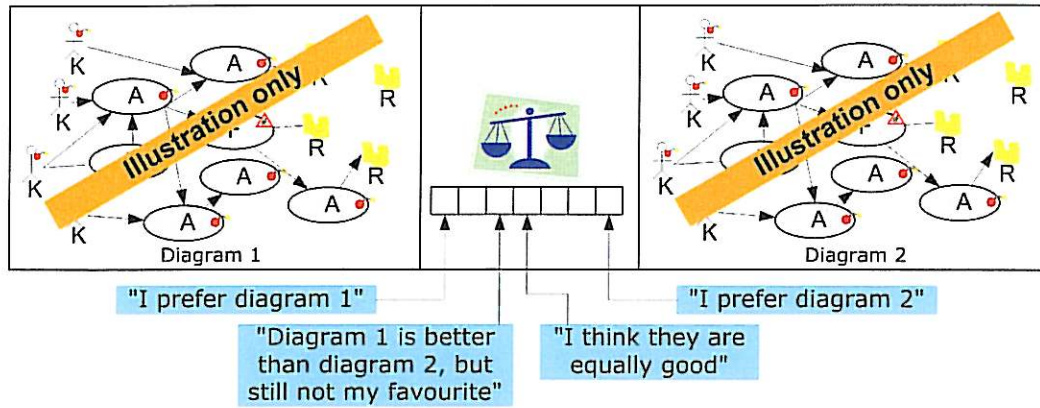


Figure 18 – Task explanation

A mark to the outmost left means “I prefer diagram 1”, the next cell means “Diagram 1 is better than diagram 2, but still not my favorite”. A mark in the middle means “I think they are equally good” while the outmost right means “I prefer diagram 2”.

4.3 Hypotheses

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.

4.4 Analysis method

The data was recoded into “0” and “1” to reflect which diagram that was preferred, an X in one of the three fields to the left 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, (Figure 19). The frequency of 1’s and 0’s could thereafter be compared. The data was not distributed normally; therefore the non-parametric Chi-Square test [39] 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.

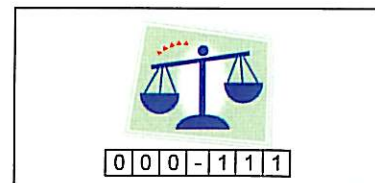


Figure 19 – Transforming data

5 Results

The results from the statistical tests are presented in the three tables below. As mentioned, the Chi-Square test is based on the assumption there will be equally many observations in each category (here: the number of “0” and “1” will be the same in each task). In the tables the difference between observed and expected observations is denoted as “Residual”. “df” denotes the degrees of freedom which is the number of variables that can vary when the sum of the variables have to add up to the observed frequency. E.g. in our case, once the frequency for the “1’s” is set, the frequency for the “0’s” is given from observed frequency minus the number of “1’s”, meaning that only the first variable is free and therefore the degrees of freedom equals 1. The outcome of the computation in a Pearson’s Chi-Square test is the Chi-Square value (abbreviated to Chi-Square), which together with the degrees of freedom (df) gives us the asymptotic significance value from the Chi-Square distribution. “Asymp. Sig.” is short for asymptotic significance (or p-value) which indicates whether the difference between two diagrams is statistically significant (in our test it must be below 0,05 to be significant). As we can see there was a significant difference between the diagram alternatives in four of the six tasks.

Table 1 shows the results from representing graph navigation: the likelihood of paths (TE2) and logical “and” (TE4).

Table 1 – Statistical results for graph navigation (TE2, TE4)

Likelihood				
		Frequencies		
		0	1	Total
TE2D1D2	Observed N	16	13	29
	Expected N	14,5	14,5	
	Residual	1,5	-1,5	
	Chi-Square	df	Asymp. Sig.	
TE2D1D2	0,310	1	0,577	

Logical “and”				
		Frequencies		
		0	1	Total
TE4D1D2	Observed N	30	2	32
	Expected N	16	16	
	Residual	14	-14	
	Chi-Square	df	Asymp. Sig.	
TE4D1D2	24,500	1	0,000	
TE4D3D1	Observed N	1	29	30
	Expected N	15	15	
	Residual	-14	14	
	Chi-Square	df	Asymp. Sig.	
TE4D3D1	26,133	1	0,000	
TE4D2D3	Observed N	2	30	32
	Expected N	16	16	
	Residual	-14	14	
	Chi-Square	df	Asymp. Sig.	
TE4D2D3	24,500	1	0,000	

The conclusion from the test of the likelihood of paths (TE2) was that neither of the modeling alternatives were preferred to the other (keep H_0). For logical “and” (TE4) we found significant differences between all modeling alternatives: D1 is preferred over both D2 and D3, and D3 is preferred over D2 (reject H_0).

Table 2 shows the results from representing vulnerabilities: shared vulnerabilities (TE5) and threats & vulnerabilities (TE6).

Table 2 – Statistical results for vulnerabilities (TE5, TE6)

Shared vulnerabilities					Threats & vulnerabilities					
		Frequencies						Frequencies		
TE5D1D2		0	1	Total	TE6D1D2		0	1	Total	
	Observed N	13	17	30		Observed N	4	25	29	
	Expected N	15	15			Expected N	14,5	14,5		
	Residual	-2	2			Residual	-10,5	10,5		
	Chi-Square	df	Asymp. Sig.		Chi-Square	df	Asymp. Sig.			
TE5D1D2	0,533	1	0,465	TE6D1D2	15,207	1	0,000			

The test of representing vulnerabilities show that neither of the diagram alternatives in shared vulnerabilities (TE5) are preferred to the other (keep H_0), while for threats & vulnerabilities (TE6) the representation in D2 is significantly preferred (reject H_0).

Table 3 shows the results from representing the magnitude of risks (TE7) and unwanted incidents (TE3).

Table 3 – Statistical results for unwanted incident/risk (TE3, TE7)

The magnitude of an unwanted incident					The magnitude of a risk					
		Frequencies						Frequencies		
TE3D1D2		0	1	Total	TE7D1D2		0	1	Total	
	Observed N	15	16	31		Observed N	9	22	31	
	Expected N	15,5	15,5			Expected N	15,5	15,5		
	Residual	-0,5	0,5			Residual	-6,5	6,5		
TE3D3D1		0	1	Total	TE7D3D1		0	1	Total	
	Observed N	23	7	30		Observed N	14	19	33	
	Expected N	15	15			Expected N	16,5	16,5		
	Residual	8	-8			Residual	-2,5	2,5		
TE3D2D3		0	1	Total	TE7D2D3		0	1	Total	
	Observed N	13	16	29		Observed N	25	6	31	
	Expected N	14,5	14,5			Expected N	15,5	15,5		
	Residual	-1,5	1,5			Residual	9,5	-9,5		
	Chi-Square	df	Asymp. Sig.		Chi-Square	df	Asymp. Sig.			
TE3D1D2	0,032	1	0,857	TE7D1D2	5,452	1	0,020			
TE3D3D1	8,533	1	0,003	TE7D3D1	0,758	1	0,384			
TE3D2D3	0,310	1	0,577	TE7D2D3	11,645	1	0,001			

From the Chi-Square test of TE3 we can conclude that the modeling alternative in D3 is preferred to that of D1, but not significantly to that of D2 (reject H_0). The result for TE7 shows that D2 is preferred to both D1 and D2 (reject H_0).

6 Discussion of modeling preferences

In the following the results for each task is discussed. The diagram alternatives are referred to as D1, D2, D3 meaning “diagram 1”, “diagram 2” etc. in the relevant figures. Whenever the results from the prestudy are relevant they will be mentioned, however since the experiment material was quite different from the one used in the prestudy we could not directly compare the results.

6.1 Representing graph navigation

The result from evaluating TE4 (Figure 6) showed that D1 was preferred to D3 which again was preferred to D2. Both alternatives with text labels were preferred to the non-text alternative. The same result was found in the prestudy using similar representations. The conclusion is that the subjects prefer a combination of a closed contour and a textual information label. Besides the possibility of being known from other notations, the and-gate shape used in D1 is clearer and more easily noticed than the other two alternatives. Using the and-symbol will also make the diagrams less cluttered compared to alternative D3.

In TE2 (Figure 7) the result showed that neither of the alternatives were preferred for this purpose. The same result was also obtained in the prestudy. It was a bit surprising that the thick line in D2 was not preferred over the thinner line in D1 since the modeling alternative is comparable to the use of thick lines in network maps to illustrate heavy traffic. A possible explanation for this is that the solidity of a line, in contrast to text, does not convey a unique interpretation. During field trials we have found it more helpful to show likelihood of paths by annotating the threat scenarios with likelihood estimates (numbers or text). The unwanted incident likelihood can then be estimated on the basis of the likelihood estimates of the threat scenarios that cause the incident. This has led to the conclusion that we should use textual information labels to show likelihood of paths, rather than relying on graphical means only.

6.2 Representing vulnerabilities

Vulnerabilities are deficiencies with respect to the analysis object that expose assets to threats. During field trials we have experienced a need for representing vulnerabilities explicitly in the diagrams. Vulnerabilities may be modeled from several perspectives: in some situations one is interested in specifying which assets that are subjects to the same vulnerabilities, while in other cases we like to see which vulnerabilities a given threat may exploit.

In TE5 (Figure 8) the experimental results showed that neither of the two representations was preferred to illustrate shared vulnerabilities. The prestudy showed a preference for D2. This task only dealt with assets and their vulnerabilities, therefore we find it surprising that D2 was not preferred since this alternative had grouped all information concerning vulnerabilities below each asset. With respect to solving a task like TE5, it would have required much more effort to use D1 where it is up to the reader to deduce which vulnerabilities are relevant for which asset.

The same two diagram alternatives were also used in TE6 (Figure 9) aiming to identify the vulnerabilities a threat may exploit. This task was new in the experiment and therefore not tested in the prestudy. The result from TE6 showed that the alternative using the chain lock symbol (D1) was significantly preferred over the alternative representation. In field trials the vulnerability symbols has proved helpful in describing threat scenarios and an excellent support in treatment identification. One of the participants in a field trial actually pointed to a vulnerability and said “*we’ve fixed this one now, so you should close the chain lock*”.

Even though the result was inconclusive for TE5, the vulnerability symbol was preferred in TE6 and has therefore been implemented in the CORAS language.

6.3 Representing risks

In security analyses we often experience that people have a problem understanding the concept “risk” due to its abstract nature. A risk is an unwanted incident that has been given specific likelihood and consequence estimates. To make the risk-concept less abstract we investigated how it can be graphically represented in threat diagrams. In risk evaluation and treatment identification we find it useful to be able to specify the magnitude of both risks and unwanted incidents.

The results for TE3 (Figure 13) showed a significant difference between the representations where D3 (textual information) was preferred to the other two alternatives. This result was also found in the prestudy. A possible explanation is that the textual information in this case had a unique interpretation, while size or color could have many interpretations. Traditionally the size of an element in software modeling has not been given a specific interpretation. The dark color was not preferred to show magnitude, but we might have obtained a different result if we had used a color signaling “danger” to a larger extent than grey. It would have been interesting to see the result if we had tested colors like red, orange and green, colors that are often used to illustrate high, medium and low risk.

It was quite surprising that the risk symbol alternative in TE7 (Figure 14, D3) received the lowest score. Also, the textual information alternative was preferred and was probably the representation that required the least effort to understand. It is important to consider the effects of giving size and color a specific semantic. Such a decision would mean that every time an element diverges from the standard size or color, the reader will expect it to mean something special, even if it was not the intention of the modeler. Also the degree of difference will play an important role. To stand out as different, the color or size has to be sufficiently different from the other elements to avoid confusion.

The overall findings from this experiment suggest that textual information in graphical models often is preferred to pure graphical means. In the already mentioned study of icons in stereotyping the original UML-based CORAS language [17] we compared a set of UML profile models to the standard UML models, both stereotyped with text labels (Figure 20). We found that the subjects receiving the special symbol version of the material completed more tasks than the other group and concluded that the text label stereotyping was not particularly significant. Comparing that conclusion to the findings in this investigation, we may draw the refined conclusion that text labels used as categorization labels may quickly be ignored by the reader because they do not convey important information needed to understand the diagram. Rather than categorizing elements with text labels, one should use text to communicate model information that the reader otherwise has to seek for in other documentation.

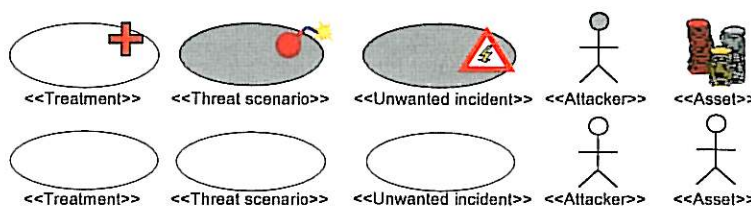


Figure 20 – Comparing stereotyping alternatives

7 Threats to validity

The main threats to validity of the empirical results of the investigation on which this report presents are described in this section. First of all, the diagrams we used were a compromise between realistic diagrams and naive notation-only tests. We cannot be sure that our results would be confirmed if we used real risk and threat diagrams. 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 will therefore validate our findings by testing them in real threat diagrams in a future field trial. With respect to text labels, it is important that the information is non-trivial and not used for categorization since they may be ignored (like the findings regarding UML stereotyping mentioned in 6.3).

The issues tested address a selection of several different security analysis modeling challenges. It would be both interesting and useful to also test other aspects and modeling alternatives, but this case we had to limit the size of the material to increase the likelihood of it being completed and handed in by the subjects.

The various modeling alternatives were tested on a population with considerable technical background. A background which is similar to that of most participants involved in security analyses and therefore considered to be a representative population.

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 it may be understood by people without a background in modeling and security analysis. 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 security expertise at a later occasion.

8 Conclusions

In a security analysis it is often helpful to draw diagrams to illustrate threat and risk scenarios. It is essential that these diagrams are easily understood by people without training and experience in modeling and security analysis. This report deals with issues related to the following question: To what extent and in what way may we improve the reading and comprehension of such diagrams by adding information with text labels or simple graphical means?

The challenge was to identify the preferred way of capturing additional information without making the language more difficult to understand. The investigation, which this report presents, focused on identifying modeling *preferences*.

Our empirical investigation showed that mechanisms like size- and color coding used to convey particular information in graphical models are less preferred by the subjects compared to textual information labels. The size or color of an element does not in general convey a unique interpretation in a model, while textual information is more specific. The subjects tend to prefer the representations where they get the most information without requiring them to interpret any additional graphical means.

8.1 Conclusions regarding “representing graph navigation”

To assess the various threats in 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 “sub lightning” (remove attention from) paths in the threat diagram one may visualize the most or least likely paths. Our study of these mechanisms in task 2 (TE2) showed that neither thick nor dashed lines were well suited for conveying this. The reason may be that different types of lines do not have a specific interpretation and therefore understanding them requires additional explanation. We did not test text labels for this purpose as they most certainly would have made the diagrams messy.

We investigated various alternatives to represent the logical and-gate from fault tree analysis and electrical circuit logic [37]. The conclusion was clear: the classical logical and-gate symbol annotated with “and” (TE4) was preferred. In other words, when joining two paths in a graph with the meaning “and” using a graphical symbol with a text label on is clearly advisable. The result for the and-gate is considered representative for the or-gate too, since these should be modeled in the same style.

To summarize: neither of the alternatives investigated were preferred, except from the gate symbol. A possible reason for this may be that the likelihood of paths between threat scenarios are less interesting than the likelihood of the actual scenarios when it comes to establishing which path a threat may choose.

8.2 Conclusions regarding “representing vulnerabilities”

In task 5 and 6 (TE5, TE6) we saw that identifying the vulnerabilities along a path between a threat and an asset is simplest done by reading the diagram from the threat on the left to the assets on the right. Looking separately at the assets and their vulnerabilities, it is preferred to have the information grouped for each asset. We conclude that since the concept of a vulnerability is closely tied to that of an asset, it is more suitable to show shared vulnerabilities in diagrams that specify assets, like in a specialized *asset diagram* (see example in Figure 22).

To summarize: if the purpose is to show the relation between assets and vulnerabilities, the context in which the vulnerabilities reside is not as important as when the threat’s preferred strategy is to be analyzed. Since we are interested in both views (threats and which vulnerabilities

they exploit, and assets and their vulnerabilities), the first should be covered in the threat diagrams and the second in the asset diagrams.

8.3 Conclusions regarding “representing risk”

An important aspect of a security analysis is to identify the risks that are above the tolerance level. It is therefore often helpful to be able to express the magnitude, or severity of both risks (TE7) and unwanted incidents (TE3) in the threat diagrams. We tested different perceptual mechanisms to illustrate magnitude and found that textual information in the diagrams is preferred to both size and color. We believe that this is because text can carry a more precise and unique interpretation than an element of varying size and color. If one decides that size has a meaning for one element, then one must consider this for the other types of elements as well and this may become too complex. For the CORAS language this means that textual information in the diagrams, possibly in combination with visual effects, is better than visual mechanisms alone. Our conclusion is that text labels should not be used as categorization labels in a diagram if they do not convey important information needed to understand the diagram. Rather than categorizing elements with text labels, one should use text to communicate diagram specific information that the reader otherwise has to seek for in other documentation.

To summarize: one should use textual information labels to indicate the severity of a risk or an unwanted incident, since the use of element size or color is too unambiguous.

8.4 How the results impacted the CORAS language

The CORAS security risk modeling language consists of several different diagram types that are used in different stages of the security analysis [16]. The result from the investigation which this report presents, had the most impact on the so-called *threat diagrams* used for risk identification and risk estimation, and the *risk diagrams*. Figure 21 shows a simple example of a threat diagram, which can be explained as follows: the target of analysis in this example is the information servers (email and file servers) in a company. The two most important assets with respect to the target are access to email and access to data files. The two threats that may affect both assets are hackers and employees, while the power supply and the electrician in combination may affect access to the file server. The hacker may exploit the old firewall solution to spread malicious code to one of the company’s computers, while employees may unintentionally introduce the same type of code. After infecting a local PC, the malicious code may spread to the email and file servers. The malicious code may cause a server crash, making the email and company files unavailable. If the power supply fails and there are no electricians available, the small stand-by-unit cannot produce sufficient power to run the file server for more than two hours. The threat scenarios and unwanted incidents have been given likelihood estimates, and the consequences of the incidents are specified on the associations to the assets.

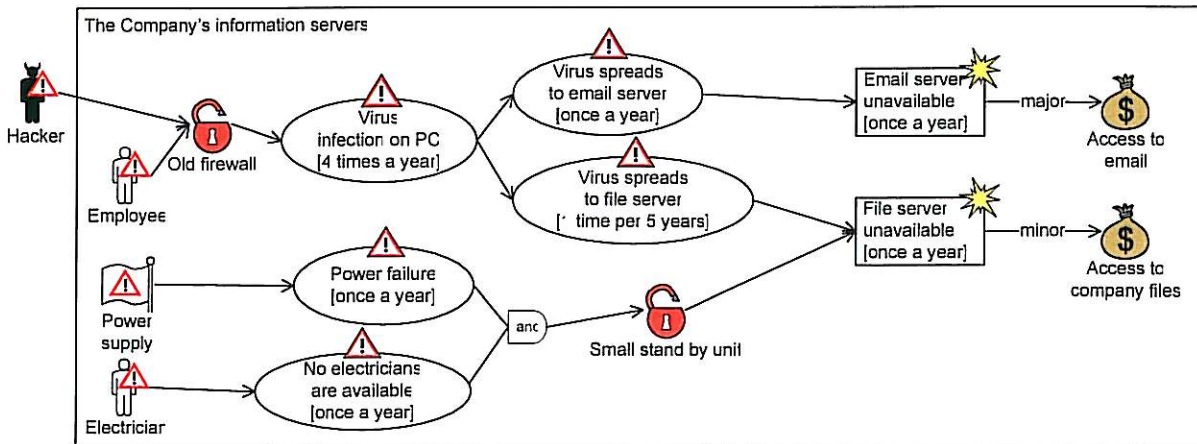


Figure 21 – Example of a threat diagram

The results from the investigations which this report presents, had the following impact:

- The logical gate symbols from FTA and circuit design were included in the language (only the and-gate is used in Figure 21, but the language provides a similar or-gate).
- The likelihood of paths is now shown by adding likelihood estimates as text labels to the threat scenarios (Figure 21).
- Vulnerabilities are now distributed in the diagram (Figure 21). 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, but as we concluded this information is better presented in a specialized asset diagram (Figure 22).
- To illustrate the severity of a risk we provide a label indicating whether the risk is acceptable or unacceptable (Figure 23). The severity of unwanted incidents is not specified explicitly, but these are implicitly shown as textual consequence- and likelihood labels in the threat diagram (Figure 21).

The asset diagram in Figure 22 specifies which vulnerabilities each asset is subject to. The risk diagram in Figure 23 illustrates which risks that are unacceptable/acceptable, which threats that may cause them and which assets that may be harmed. As we see, the employee and the hacker may cause both risks, while a power failure and the electrician may only make the file server unavailable.

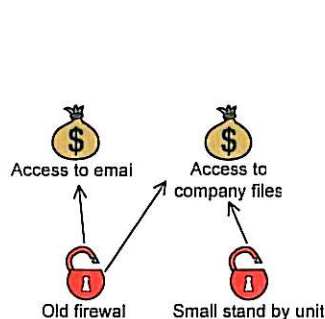


Figure 22 – Asset diagram: vulnerabilities

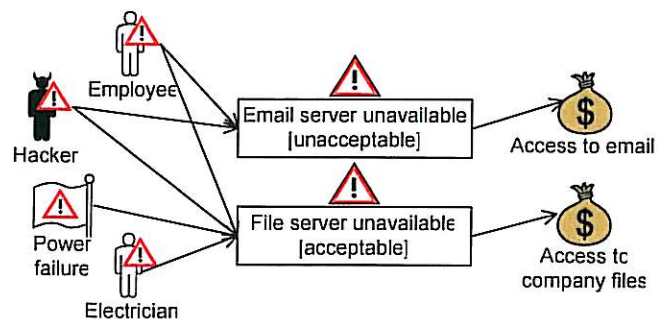


Figure 23 – Risk diagram

Acknowledgements

The research which this report presents has been funded by the Research Council of Norway project SECURIS (152839/220). The authors are very grateful to the subjects that participated voluntarily. We also thank Jan Heim for his help and guidance, and the SECURIS team: Gyrd Brændeland, Heidi Dahl, Iselin Engan, Mass Soldal Lund, Fredrik Seehusen, Bjørnar Solhaug and Fredrik Vraalsen.

References

- [1] Alexander, I., "Misuse cases: Use cases with hostile intent", *IEEE Software*, vol. 20 (1), pp. 58-66, 2003.
- [2] AS/NZS4360, Australian/New Zealand Standard for Risk Management: Standards Australia/Standards New Zealand, 2004.
- [3] 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.
- [4] 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.
- [5] 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.
- [6] Cavanagh, J. P., "Relationship between the immediate memory span and the memory search rate", *Psychological Review*, vol. 79, pp. 525-530, 1972.
- [7] Chattratchart, 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.
- [8] 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.
- [9] Christ, R. E., "Review and analysis of color coding research for visual displays", *Human Factors*, vol. 17 (6), pp. 542-570, 1975.
- [10] Cleveland, W. S. and McGill, R., "Graphical perception and graphical methods for analyzing scientific data." *Science*, vol. 229, pp. 828-833, 1985.
- [11] den Braber, F., Mildal, A.-B., Nes, J., Stølen, K., and Vraalsen, F., "Experiences from Using the CORAS Methodology to Analyze a Web Application", *Journal of Cases on Information Technology*, vol. 7 (3), pp. 110-130, 2005.
- [12] Freeman, R. E., *Strategic Management: A Stakeholder Approach*: Ballinger Publishing, 1984.
- [13] Goodman, N., *Languages of Art: An Approach to a Theory of Symbols*. Indianapolis: Hackett, 1976.
- [14] HB231, Information security risk management guidelines: Standards Australia/Standards New Zealand, 2000.
- [15] HB231, Information security risk management guidelines: Standards Australia/Standards New Zealand, 2004.
- [16] Hogganvik, I. and Stølen, K., "A Graphical Approach to Risk Identification, Motivated by Empirical Investigations", in Proc. MoDELS 2006 (LNCS 4199), pp. 574-588, 2006.
- [17] Hogganvik, I. and Stølen, K., "On the Comprehension of Security Risk Scenarios", in Proc. 13th Int. Workshop on Program Comprehension (IWPC'05), pp. 115-124, 2005.
- [18] Hogganvik, I. and Stølen, K., "Risk Analysis Terminology for IT-systems: does it match intuition?" in Proc. Int. Symposium on Empirical Software Engineering (ISESE'05), pp. 13-23, 2005.
- [19] IEC60300-3-9, Event Tree Analysis in Dependability management - Part 3: Application guide - Section 9: Risk analysis of technological systems., 1995.
- [20] IEC61025, Fault Tree Analysis (FTA), 1990.
- [21] ISO/IEC13335, Information technology - Guidelines for the management of IT Security (Part 3), 1998.
- [22] ISO/IEC13335, Information technology - Security techniques - Management of information and communications technology security (Part 1), 2004.

-
- [23] Jacobson, I., Christerson, M., Jonsson, P., and Övergaard, G., *Object-Oriented Software Engineering: A Use Case Driven Approach*: Addison-Wesley, 1992.
- [24] Jensen, F. V., *Bayesian Networks and Decision Graphs*: Springer, 2002.
- [25] 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.
- [26] Jürjens, J., *Secure Systems Development with UML*: Springer, 2005.
- [27] 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.
- [28] 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.
- [29] 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.
- [30] 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.
- [31] 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. report STF40 A03067, 2004.
- [32] Lund, M. S., Hogganvik, I., Seehusen, F., and Stølen, K., "UML profile for security assessment", SINTEF ICT, Tech. report STF40 A03066, 2003.
- [33] OMG, "UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms", Object Management Group, 2006.
- [34] OMG, "Unified Modeling Language (UML): Superstructure, version 2.0", Object Management Group, 2005.
- [35] 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.
- [36] Schneier, B., "Attack trees: Modeling security threats", *Dr. Dobbs's Journal*, vol. 24 (12), pp. 21-29, 1999.
- [37] Sedra, A. S. and Smith, K. C., *Microelectronic Circuits*: Oxford University Press, 2003.
- [38] Shneiderman, B., *Designing the User Interface*: Addison-Wesley, 1992.
- [39] Siegel, S. and Castellan, J., *Non-parametric Statistics for the Behavioural Sciences*, 2 ed: McGraw-Hill International Editions, 1988.
- [40] Sindre, G. and Opdahl, A. L., "Eliciting Security Requirements by Misuse Cases", in Proc. TOOLS-PACIFIC, pp. 120-131, 2000.
- [41] 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.
- [42] Smith, L. and Thomas, D., "Color versus shape coding in information displays", *Journal of Applied Psychology*, vol. 48 (3), pp. 137-146, 1964.
- [43] 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.
- [44] Treisman, A. and Gormican, S., "Feature analysis in early vision: Evidence from search asymmetries", *Psychological Review*, vol. 95 (1), pp. 15-48, 1988.
- [45] Ware, C., *Information Visualization: Perception for Design*, 2 ed: Elsevier, 2004.
-

-
- [46] 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.*
- [47] Wickens, C. D., *Engineering Psychology and Human Performance*, 2 ed: HarperCollins, 1992.
- [48] Winn, W., "An Account of How Readers Search for Information in Diagrams", *Contemporary Educational Psychology*, vol. 18, pp. 162-185, 1993.
- [49] Winn, W., "Perceptual strategies used with flow diagrams having normal and unanticipated formats", *Perceptual and Motor Skills*, vol. 57, pp. 751-762, 1983.
- [50] 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.
- [51] 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.
- [52] Aagedal, J. Ø., Braber, d. F., Dimitrakos, T., Gran, B. A., Raptis, D., and Stølen, K., "Model-based risk assessment to improve enterprise security", in *Proc. Enterprise Distributed Object Communication (EDOC'02)*, pp. 51-64, 2002.
-

List of Tables

Table 1 – Statistical results for graph navigation (TE2, TE4).....	13
Table 2 – Statistical results for vulnerabilities (TE5, TE6).....	14
Table 3 – Statistical results for unwanted incident/risk (TE3, TE7).....	14
Table 4 – Results for the tasks left out of the report	34
Table 5 – Results from SPSS	41
Table 6 – Task conclusions	42

List of Figures

Figure 1 – The conceptual foundation.....	3
Figure 2 – Percentage of correct answers for each category (mean).....	4
Figure 3 – Typical threat scenario situation	5
Figure 4 – Node-link configurations	5
Figure 5 – The AND-gate symbol	6
Figure 6 – The diagram alternatives for: “most likely path through a graph”	6
Figure 7 – The diagram alternatives for: “logical AND in graph paths”	7
Figure 8 – The diagram alternatives for: “shared vulnerabilities”	8
Figure 9 – The diagram alternatives for: “which vulnerabilities are exploited by which threats?” ..8	
Figure 10 – Color	9
Figure 11 – Size.....	9
Figure 12 – Shapes	10
Figure 13 – The diagram alternatives for: “the magnitude of unwanted incidents”	10
Figure 14 – The diagram alternatives for: “the magnitude of risks”	10
Figure 15 – The subjects’ years of work experience.....	11
Figure 16 – The subjects’ task experience	11
Figure 17 – The basic threat diagram.....	11
Figure 18 – Task explanation	12
Figure 19 – Transforming data.....	12
Figure 20 – Comparing stereotyping alternatives	16
Figure 21 – Example of a threat diagram	20
Figure 22 – Asset diagram: vulnerabilities.....	20
Figure 23 – Risk diagram	20
Figure 24 – Example of a Riskit graph.....	27
Figure 25 – Bayesian network.....	27
Figure 26 – Example	28
Figure 27 – Task set-up	29

Appendix A – Related modeling notations

The starting point for our work was a UML [34] profile [33] developed in the EU-funded research project CORAS [52]. As a result of our empirical investigations with the aim to satisfy the modeling needs in security analysis, the CORAS language has undergone numerous modifications and refinements since the completion of the CORAS project, and the current version is no longer a UML profile.

Misuse cases [1, 40, 41] was an important source of inspiration in the development of the above mentioned UML profile. A misuse case is a kind of UML use case [23] which characterizes functionality that the system should *not* allow. The use case notation is often employed for high level specification of systems and considered to be one of the most understandable notations in UML. A misuse case can be defined as “a completed sequence of actions which results in loss for the organization or some specific stakeholder” [41]. To obtain more detailed specifications of the system, the misuse cases can be specialized into e.g. sequence or activity diagrams.

There are a number of security oriented extensions of UML, e.g. UMLSec [26] and SecureUML [30]. These and other related languages have however all been designed to capture security properties and security aspects at a more detailed level than the CORAS language. Moreover, they do not emphasize brainstorming sessions as in our case. SecureUML: modeling access control policies in the relationship of model-driven software development. UMLsec: an extension of UML that gives the possibility to model security related features like access control, integrity and confidentiality.

Fault tree is a tree-notation used in fault tree analysis (FTA) [20]. The top node represents an unwanted incident, or failure, and the different events that may lead to the top event are modeled as branches of nodes, with the leaf node as the causing event. The probability of the top node can be calculated on the basis of the probabilities of the leaf nodes and the logical gates “and” and “or”. The CORAS threat diagrams often look a bit like fault trees, but may have more than one top node. Computing the top node probability of a fault tree requires precise quantitative input, which is rarely available for incidents caused by human errors or common cause failures. It is possible to model fault trees using the CORAS language, but it also allows qualitative likelihood values as input.

Event tree analysis (ETA) [19] focuses on illustrating the consequences of an event and the probabilities of these. It is often used as an extension of fault trees to create “cause-consequence” diagrams. Event trees can also to a large extent be simulated in our notation.

Attack trees [36] aim 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 fault trees, with the attack goal as the top node and different ways of achieving the goal as leaf nodes. Our language supports this way of modeling, but facilitates in addition the specification of the attack initiators (threats) and the harm caused by the attack (damage to assets).

The Riskit method [27] includes a risk modeling technique that makes it possible to specify factors that may influence a software development project (example in Figure 24 from [27]). It has similarities to the CORAS language, but targets project risks: “When dealing with software risks, the risks caused by the *operation* of software are often also considered. As our primary focus has been to support software development organizations in *developing* their products, we have excluded the evaluation and management of risks that might occur during the operation of software” [27] (p. 12).

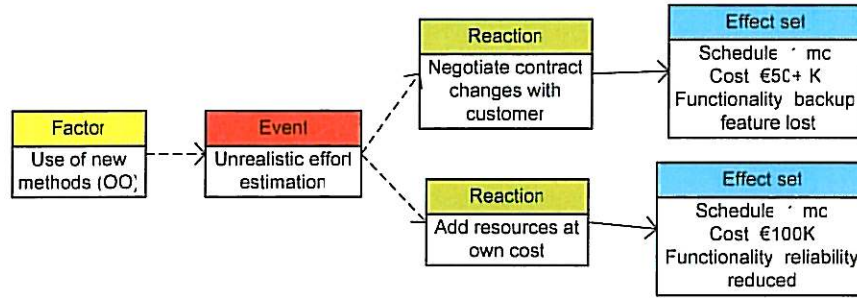


Figure 24 – Example of a Riskit graph

The Riskit modeling approach is related to the CORAS approach, but targets project risks and utilizes different concepts and definitions. Since the CORAS method targets security risks, its definitions are taken from information security related standards [2, 14, 21, 22]; Riskit, on the other hand uses its own definitions inspired by e.g. organizational strategy research [12]. As an example of the difference in concepts, Riskit lacks “threat”, possibly because it unusual to experience deliberate harmful actions towards a software development project.

A Bayesian network [24] is a directed acyclic graph that represents a probability distribution. The nodes in the network represent variables and links represent relationships between the variables (Figure 25).

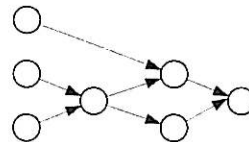


Figure 25 – Bayesian network

The root nodes are independent, while the probabilities of the intermediate nodes depend on the probabilities of its preceding nodes. A Bayesian network lets you decompose a probability distribution into a set of local distributions. These local distributions are used to compute the resulting top-node probability. A Bayesian network specifies both the qualitative structure of the case and lets you compute the quantitative probability values. Our modeling technique also provides a means to describing the behavior of a threat qualitatively. The statistical distributions that a Bayesian network requires to perform probability computations are seldom available in our types of analyses and precise statistical computations have therefore not played an important role in our work. However, if such statistics are available and all the assumptions of constructing a Bayesian network are met (statistical independence of the distributions), the CORAS language can be used to describe a Bayesian network.

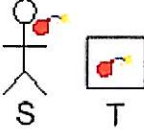
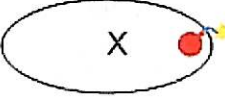

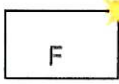

Appendix B – Experiment Material

This appendix contains the experiment material used in the main study. The material has been kept in its original format, to the extent that was possible. However, seeing that the original task set-up (illustrated in Figure 27) was space consuming it has been left out in this report. Two tasks have been discarded from the analysis: TE1 and TE8. The reasons for this are explained at the end of this appendix.

B.1 Modeling questionnaire

This questionnaire is about threat modeling. We present various ways of modeling the same situation, and your task is to prioritize between them and choose the diagram you think is best! This type of modeling is meant to support the risk analyst in situations where the participants in meetings have different background and expertise (technical and non-technical). The diagrams are meant to help them understand the system and each other better and faster.

Symbols and definitions:

Symbols			
 S T (2 symbols)	Threat: a human/non-human that intentionally or non-intentionally can cause an event that may harm an asset.		Threat scenario: a sequence of events corresponding to a threat exploiting vulnerabilities, potentially causing unwanted incidents.
 G	Asset: something of value which needs to be protected.		Unwanted incident: an undesired event that may reduce the value of an asset.
 R	Vulnerability: a weakness or a deficiency that a threat can exploit.		

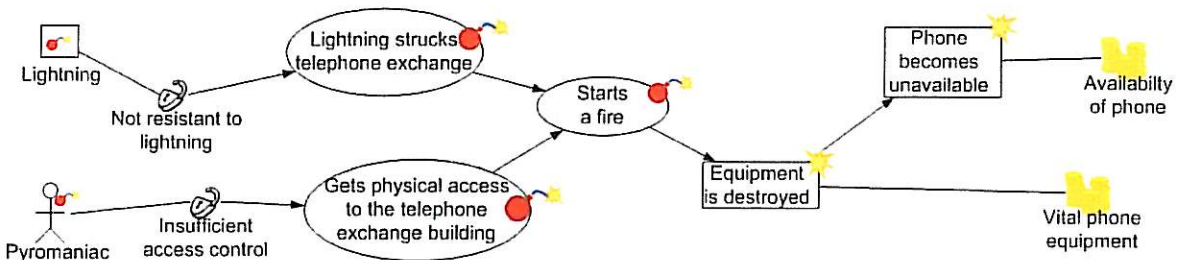


Figure 26 – Example

About you:

A) Gender: Female Male

B) Age: Under 25 years old
 26-35
 36-45
 46-55
 56-65
 Over 65

C) Current position/title:

- D) Years of work experience:**
- Less than 1 year
 - 1-3 years
 - 4-10 years
 - Over 10 years

- E) Which tasks have you experience from?**
- System design
 - System testing
 - System development
 - System quality evaluation
 - System maintenance
 - System marketing/sale
 - System security

Prioritizing different ways of modeling threat diagrams:

Compare two and two diagrams and choose the one you prefer by marking with an “X” in the table below the weight symbol, as shown in this example:

EXAMPLE: Which diagram do you prefer, diagram 1 or diagram 2?

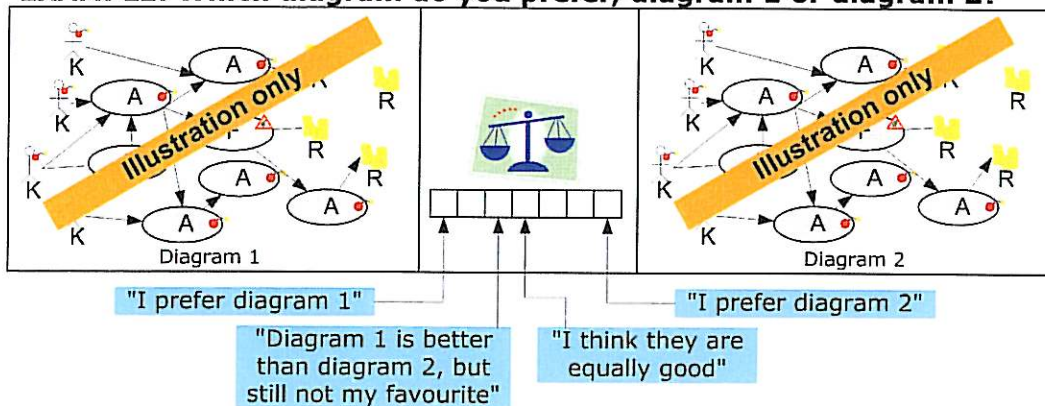


Figure 27 – Task set-up

TE1 - Which diagram is best at illustrating that threat scenario N can cause the unwanted incident F?

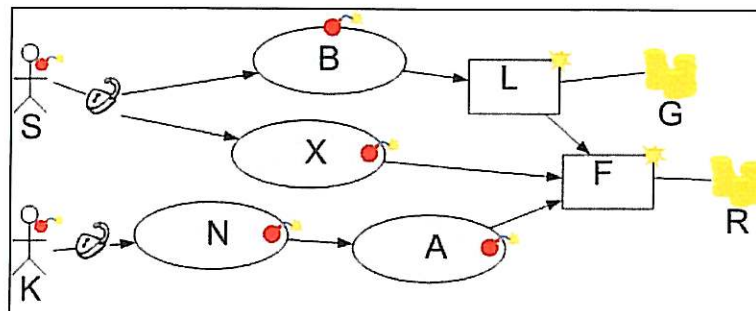


Diagram 1

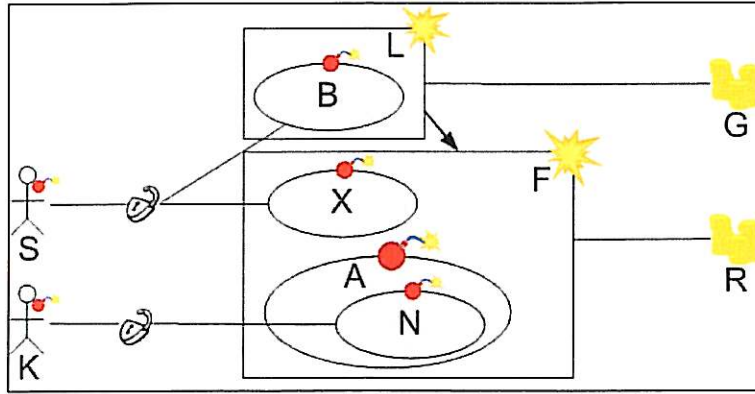


Diagram 2

TE2 - Which diagram is best at illustrating that it is more likely that threat K causes the unwanted incident T than L or F?

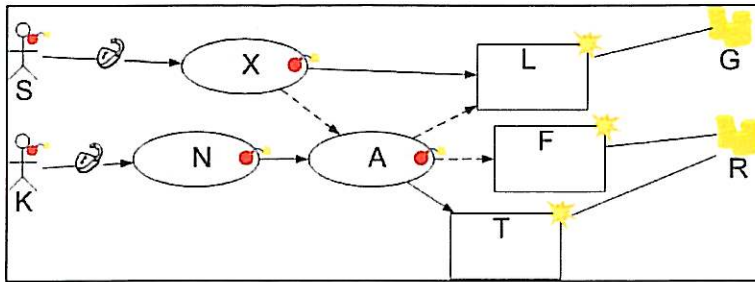


Diagram 1

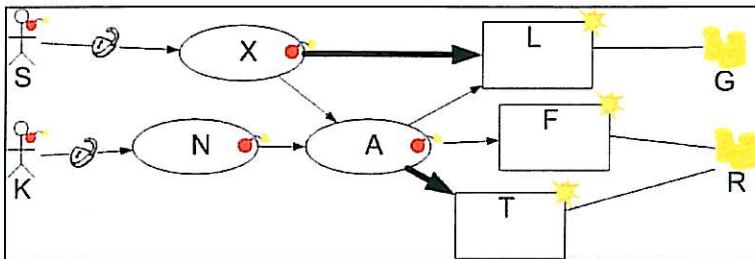


Diagram 2

TE3 - Which diagram is best at illustrating that unwanted incident L is more serious than unwanted incident F?

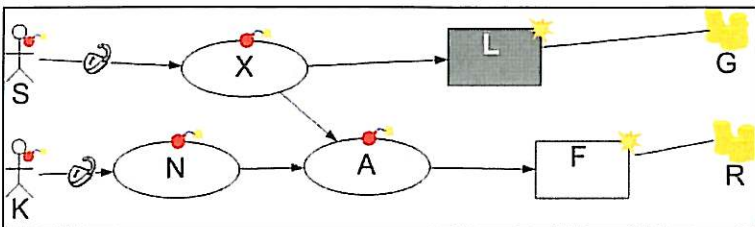


Diagram 1

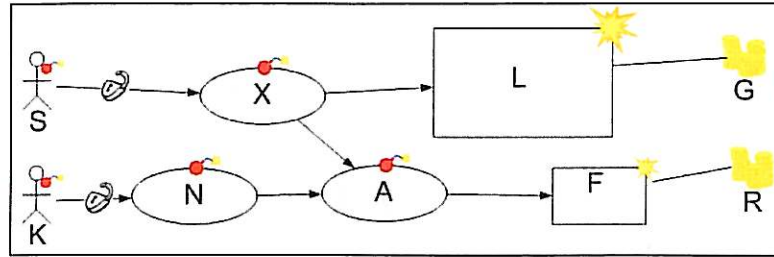


Diagram 2

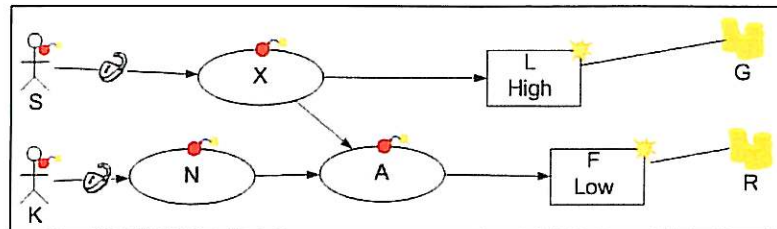


Diagram 3

TE4 - Which diagram is best at illustrating that both X and N must happen before A can happen?

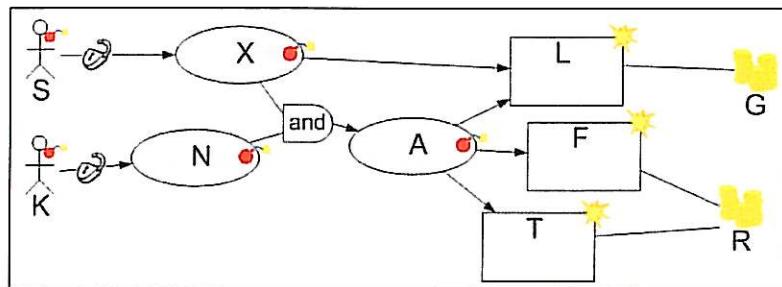


Diagram 1

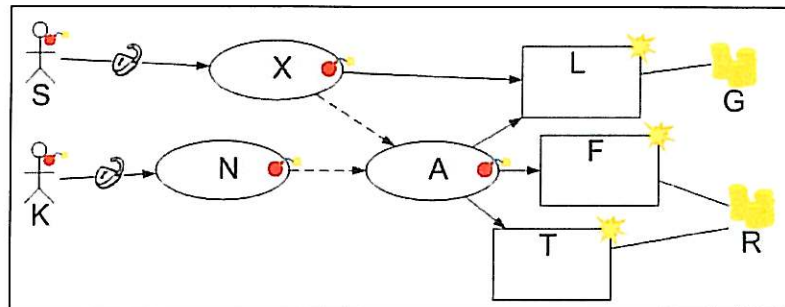


Diagram 2

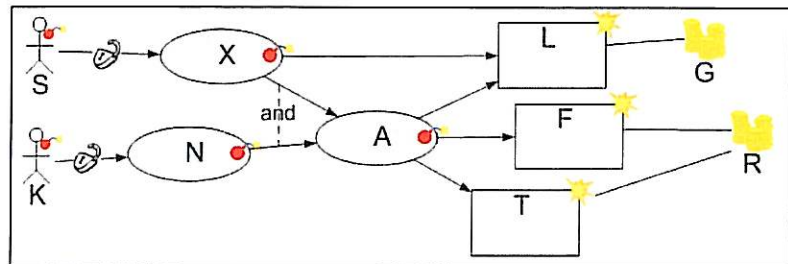


Diagram 3

TE5 - Which diagram is best at illustrating that the asset G and R share the same vulnerability “accessible remotely”?

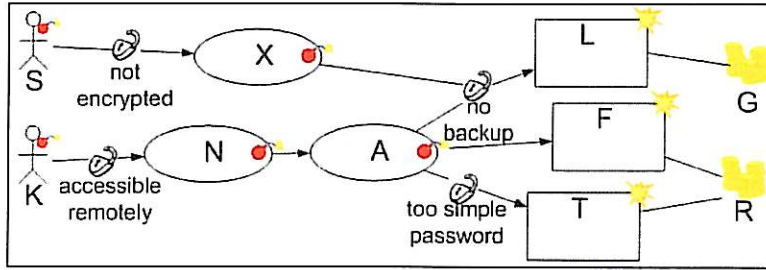


Diagram 1

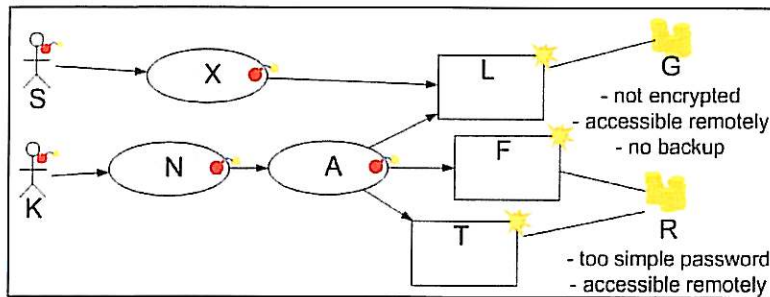


Diagram 2

TE6 - Which diagram is best at illustrating that only threat K can exploit the vulnerability “too simple password”?

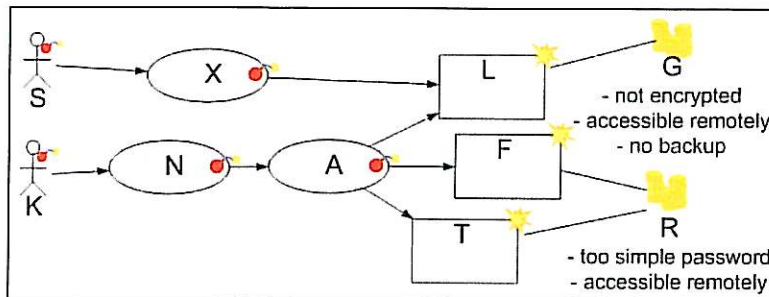


Diagram 1

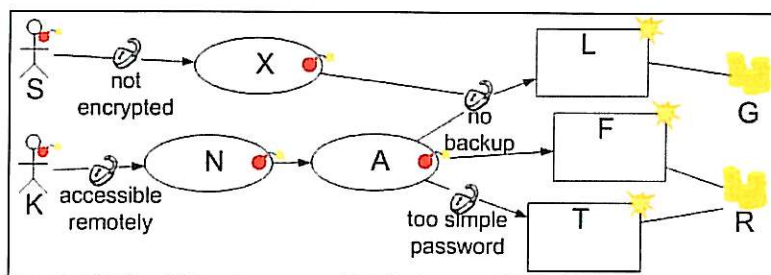


Diagram 2

TE7 - Which diagram is best at illustrating that unwanted incident F poses the largest risk for asset R?

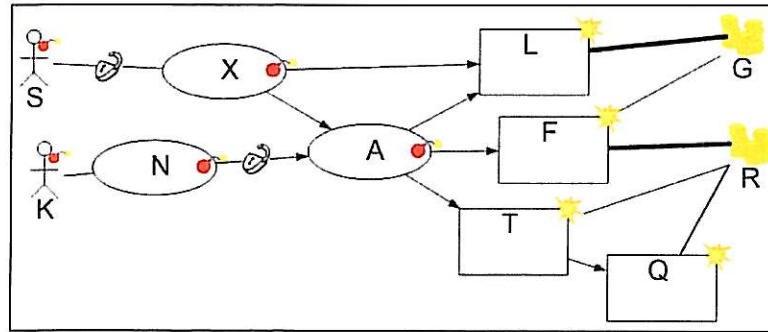


Diagram 1

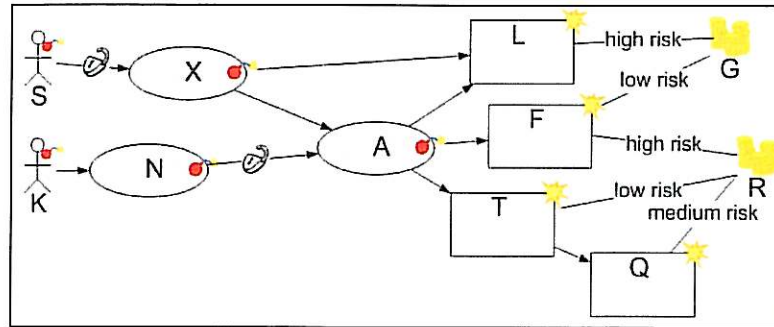


Diagram 2

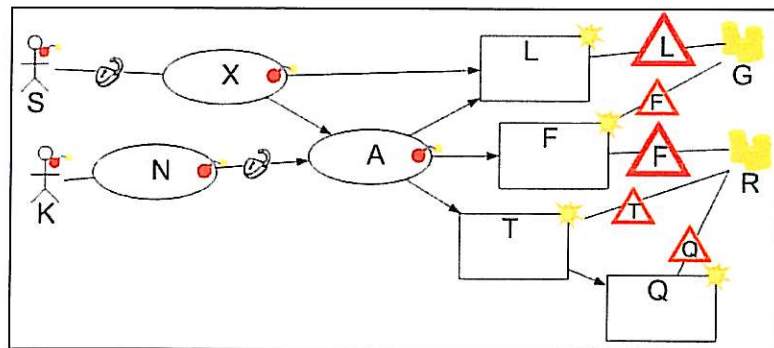


Diagram 3

TE8 - Which diagram is best at illustrating that unwanted incident F represents two *different* risks?

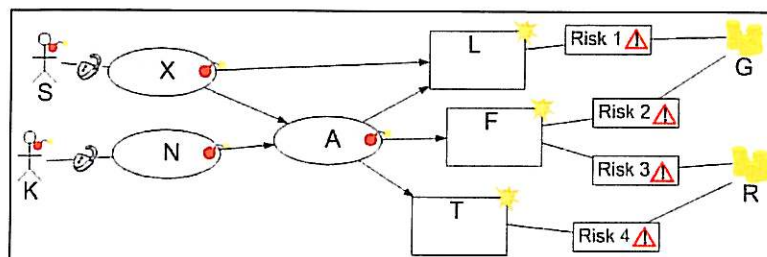


Diagram 1

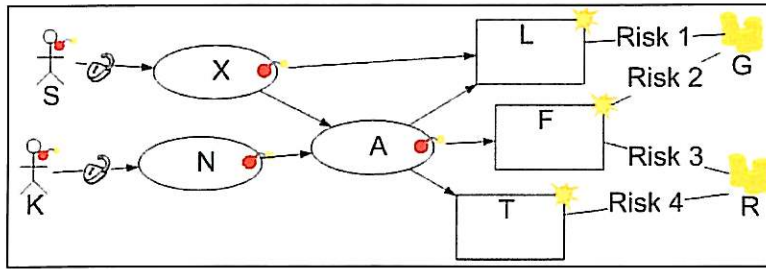


Diagram 2

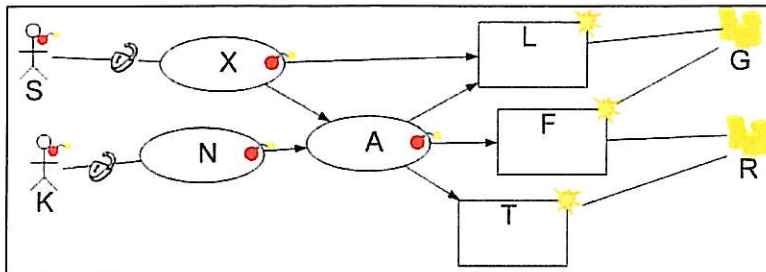


Diagram 3

B.II Tasks that were left out of the analysis

Two tasks were found either superfluous or too inconsistent to be part of the report. TE1 was similar to TP8 in the prestudy, but using the new notation. The alternative representation in Diagram 1 has later been found to be too impractical for real use. A diagram with several inclusions would imply several large boxes in a potentially cluttered and complex pattern. Since this task had only two modeling alternatives, it seemed unfair to compare a bad modeling alternative with another, because this could be an unintentional advantage.

The other modeling task that was omitted from the analysis was TE8. The intention behind the task was to investigate the difference between no text labels, text labels and boxes with text labels. By mistake, a risk icon was also included in the box, which made it more complex than the other two. Any effects, either positive or negative, with respect to this alternative could not have been said to have been caused by the use of boxes, textual labels or icons. Due to this inconsistency we decided to leave it out from this investigation.

Nevertheless, we provide the statistical results for TE1 and TE8 in the table below.

Table 4 – Results for the tasks left out of the report

TE1				
		Frequencies		
TE1D1D2		0	1	Total
	Observed N	30	3	33
	Expected N	16,5	16,5	
	Residual	13,5	-13,5	
	Chi-Square	df	Asymp. Sig.	
TE1D1D2	22,091	1	0,000	

TE8				
		Frequencies		
TE8D1D2		0	1	Total
	Observed N	17	9	26
	Expected N	13	13	
	Residual	4	-4	
TE8D3D1		0	1	Total
	Observed N	12	20	32
	Expected N	16	16	
	Residual	-4	4	
TE8D2D3		0	1	Total
	Observed N	19	12	31

	Expected N	15,5	15,5	
	Residual	3,5	-3,5	
	Chi-Square	df	Asymp. Sig.	
TE8D1D2	2,462	1	,117	
TE8D3D1	2,000	1	,157	
TE8D2D3	1,581	1	,209	

Appendix C – Prestudy

This appendix gives an overview of the prestudy, its design, material and results. Some of the material was also used in the main study, but the results cannot be compared directly since the material has been slightly modified.

C.I Prestudy design

The subjects in the prestudy were 11 master students in informatics at University of Oslo. The survey was taken as a part of one of the classes in the course “INF5150-Unassailable IT-systems”, autumn 2005 (<http://www.uio.no/studier/emner/matnat/ifi/INF5150/h05/>). The subjects had to assess several modeling alternatives and select the one they preferred the most.

C.II Prestudy material

The prestudy aimed to investigate the original UML profile notation [32] in addition to various new alternative modeling notations. The survey consisted of 7 tasks, numbered from 8-14 (task 1-7 were conventional educational tasks and therefore not part of the experiment). In the result section the following numbering convention is used: TP8D1D2 meaning “Task 8 in Prestudy, Diagram 1 compared to Diagram 2”. Each task includes a question and 2-3 diagram alternatives. The material set-up and introduction was similar to the one presented in Appendix B.

TP8 - Which diagram shows best that threat scenario N can cause the unwanted incident of incident scenario L?

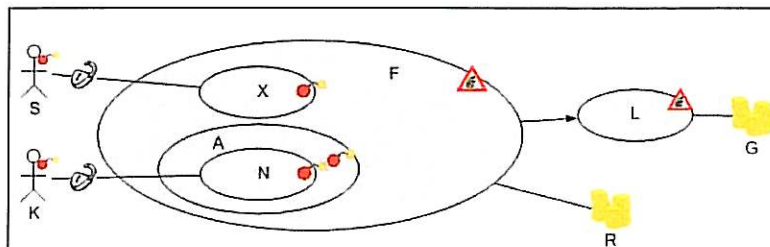


Diagram 1

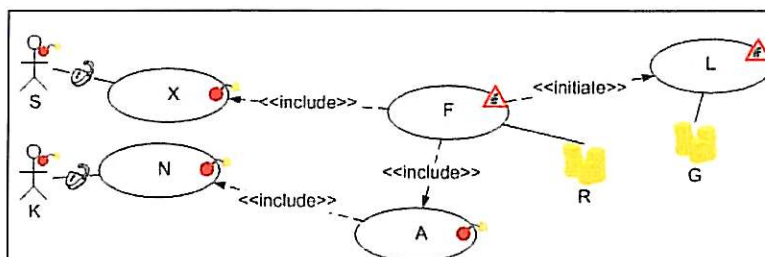


Diagram 2

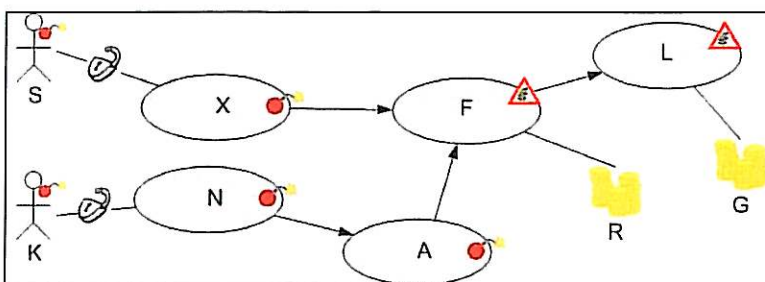


Diagram 3

TP9 - Which diagram shows best that the unwanted incident of incident scenario L is more serious than unwanted incident of F?

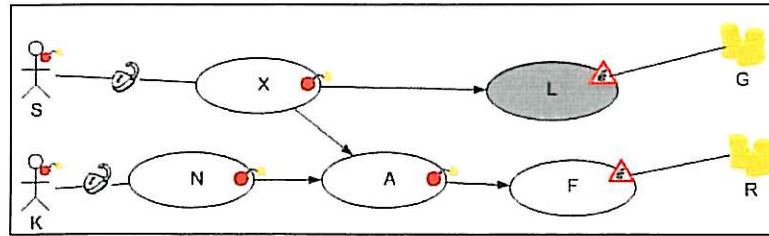


Diagram 1

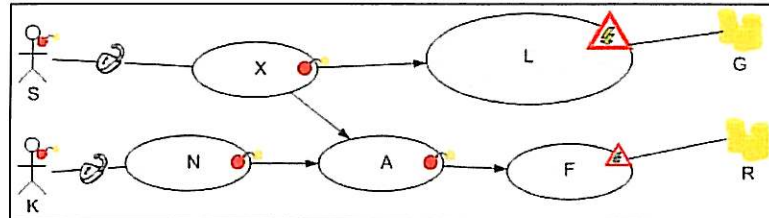


Diagram 2

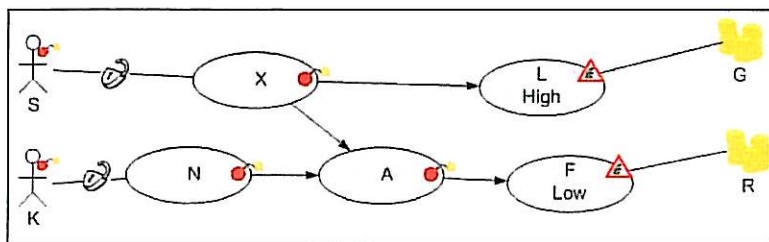


Diagram 3

TP10 - Which diagram shows best that it is more likely that T happens after A, than L or F?

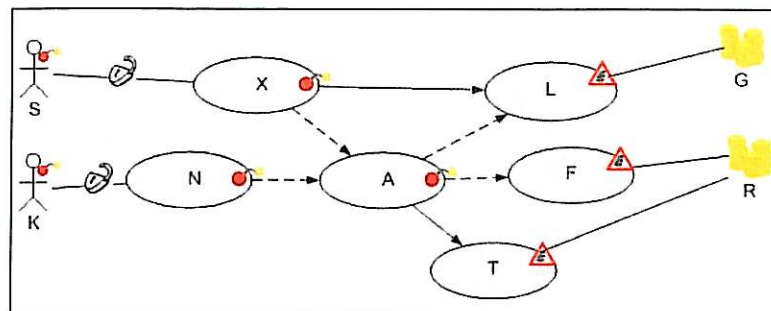


Diagram 1

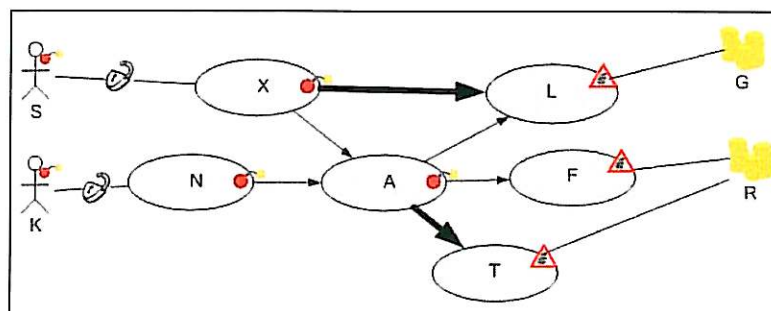


Diagram 2

TP11 - Which diagram shows best that both X and N must happen before A can happen?

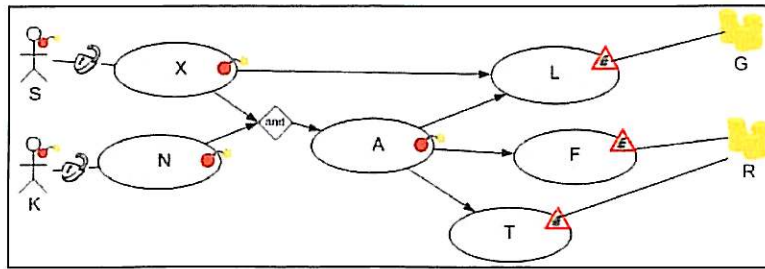


Diagram 1

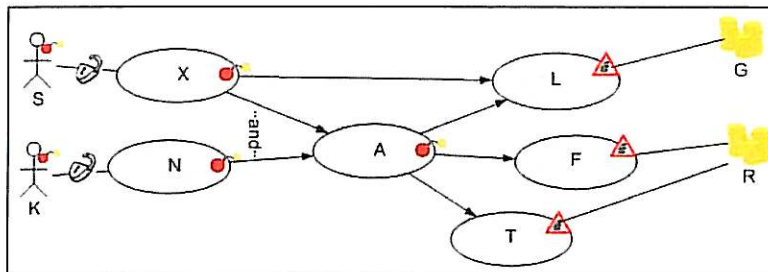


Diagram 3

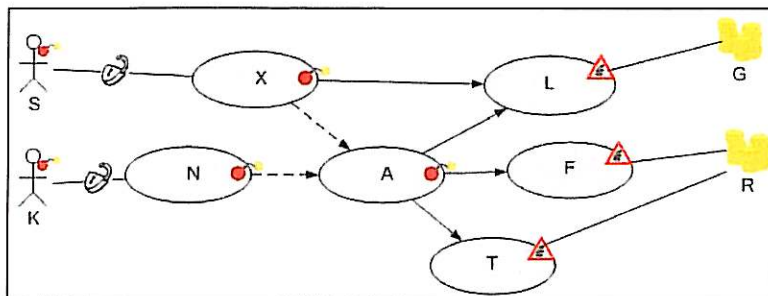


Diagram 2

TP12 - Which diagram shows best that the asset G and R share the same vulnerability?

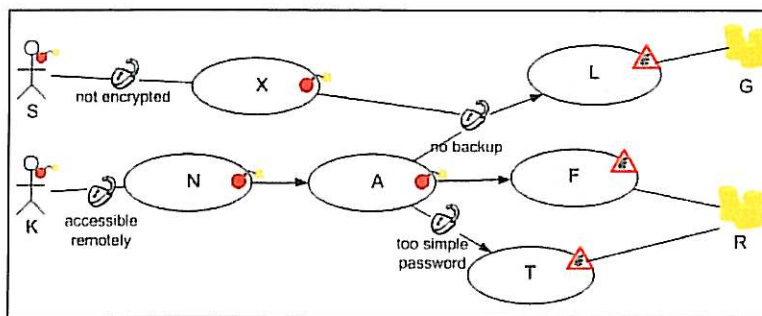


Diagram 1

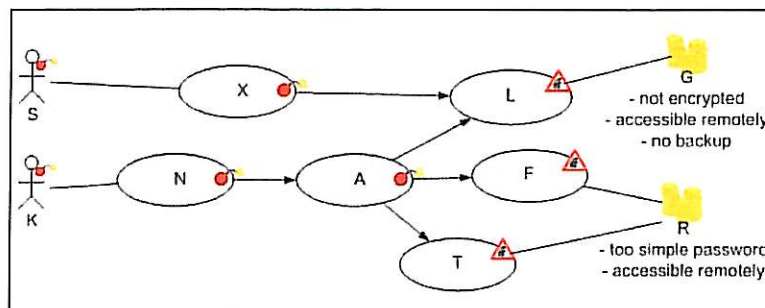


Diagram 2

TP13 - Which diagram shows best that unwanted incident F poses the largest risk for asset R?

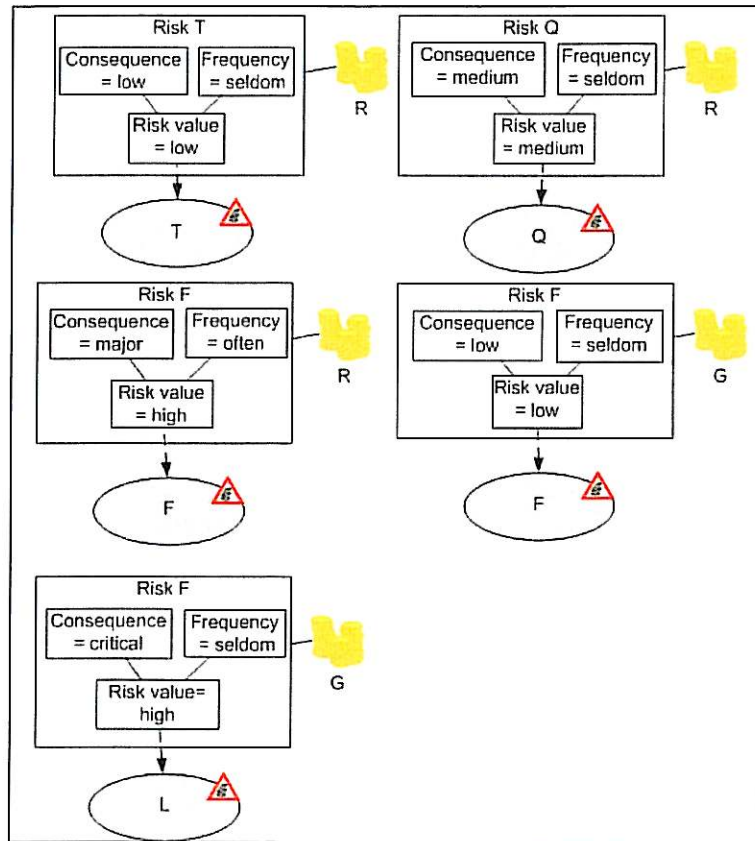


Diagram 1

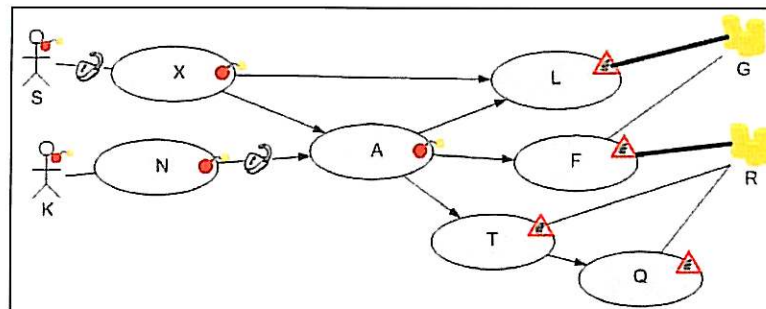


Diagram 2

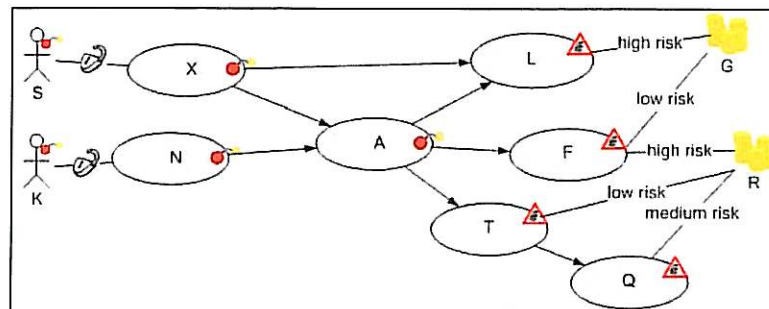


Diagram 3

TP14 - Which diagram shows best that unwanted incident of scenario F represents two different risks?

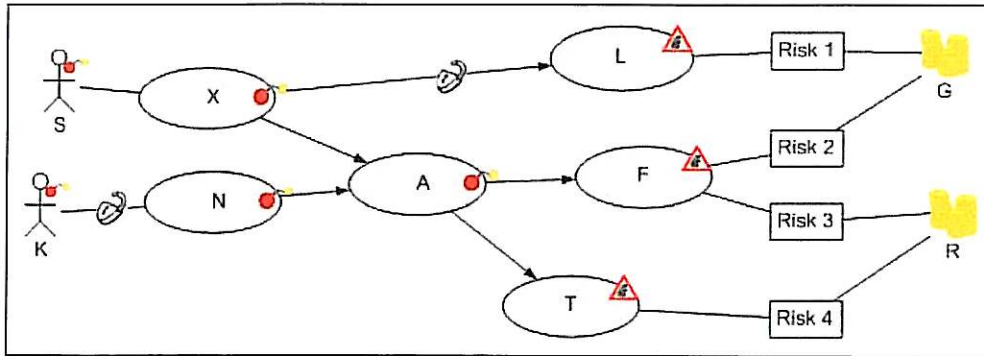


Diagram 1

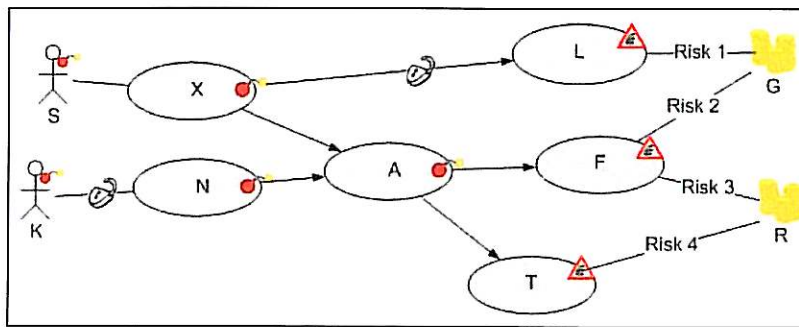


Diagram 2

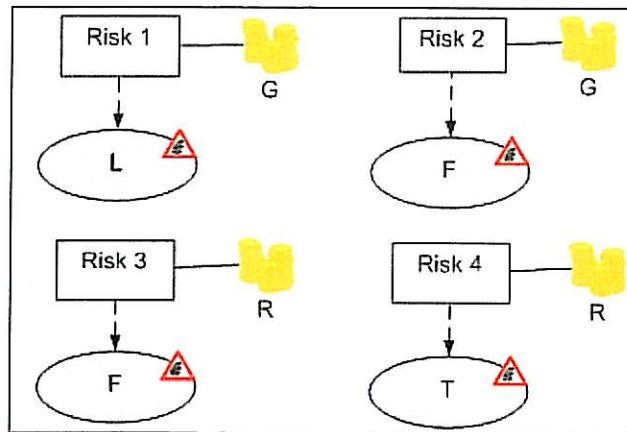


Diagram 3

C.III Prestudy results and discussion

The results from the statistical tests are presented in the tables below. The Chi-Square test is based on the assumption that there will be equally many observations in each category (here: the number of “0” and “1” will be the same in each task). In the tables the difference between observed and expected observations is denoted as “Residual”. “df” denotes the degrees of freedom which is the number of variables that can vary when the sum of the variables have to add up to the observed frequency. E.g. in our case, once the frequency for the “1’s” is set, the frequency for the “0’s” is given from the observed frequency minus the number of “1’s”, meaning that only the first variable is free and therefore the degrees of freedom equals 1. The outcome of the computation in a Pearson’s Chi-Square test is the Chi-Square value (abbreviated to Chi-Square), which together with the degrees of freedom (df) gives us the asymptotic significance value from the Chi-Square

distribution. “Asymp. Sig.” is short for asymptotic significance (or p-value) which indicates whether the difference between two diagrams is statistically significant (in our test it must be below 0,05 to be significant). As we can see there was a significant difference between the diagram alternatives in five of the seven tasks.

Table 5 – Results from SPSS

Conceptual containment				
		Frequencies		
TP8D1D2		0	1	<i>Total</i>
	Observed N	4	6	10
	Expected N	5	5	
	Residual	-1	1	
TP8D3D1		0	1	<i>Total</i>
	Observed N	8	2	10
	Expected N	5	5	
	Residual	3	-3	
TP8D2D3		0	1	<i>Total</i>
	Observed N	4	6	10
	Expected N	5	5	
	Residual	-1	1	
	Chi-Square	df	Asymp. Sig.	
TP8D1D2	0,4	1	0,527	
TP8D3D1	3,6	1	0,058	
TP8D2D3	0,4	1	0,527	
Likelihood				
		Frequencies		
TP10D1D2		0	1	<i>Total</i>
	Observed N	3	6	9
	Expected N	4,5	4,5	
	Residual	-1,5	1,5	
	Chi-Square	df	Asymp. Sig.	
TP10D1D2	1	1	0,317	
Shared vulnerabilities				
		Frequencies		
TP12D1D2		0	1	<i>Total</i>
	Observed N	1	7	8
	Expected N	4	4	
	Residual	-3	3	
	Chi-Square	df	Asymp. Sig.	

Magnitude of unwanted incident				
		Frequencies		
TP9D1D2		0	1	<i>Total</i>
	Observed N	6	1	7
	Expected N	3,5	3,5	
	Residual	2,5	-2,5	
TP9D3D1		0	1	<i>Total</i>
	Observed N	7	3	10
	Expected N	5	5	
	Residual	2	-2	
TP9D2D3		0	1	<i>Total</i>
	Observed N	0	9	9
	Expected N	4,5	4,5	
	Residual	-4,5	4,5	
	Chi-Square	df	Asymp. Sig.	
TP9D1D2	3,571	1	0,059	
TP9D3D1	1,6	1	0,206	
TP9D2D3	9	1	0,003	
Logical “and”				
		Frequencies		
TP11D1D2		0	1	<i>Total</i>
	Observed N	7	2	9
	Expected N	4,5	4,5	
	Residual	2,5	-2,5	
TP11D3D1		0	1	<i>Total</i>
	Observed N	1	7	8
	Expected N	4	4	
	Residual	-3	3	
TP11D2D3		0	1	<i>Total</i>
	Observed N	2	6	8
	Expected N	4	4	
	Residual	-2	2	
	Chi-Square	df	Asymp. Sig.	
TP11D1D2	2,778	1	0,096	
TP11D3D1	4,5	1	0,034	
TP11D2D3	2	1	0,157	

Magnitude of risk					One incident, multiple risks				
		Frequencies					Frequencies		
TP12D1D2		4,5	1	0,034					
TP13D1D2		0	1	Total	TP14D1D2		0	1	Total
	Observed N	3	6	9		Observed N	4	2	6
	Expected N	4,5	4,5			Expected N	3	3	
	Residual	-1,5	1,5			Residual	1	-1	
TP13D3D1		0	1	Total	TP14D3D1		0	1	Total
	Observed N	9	1	10		Observed N	0	10	10
	Expected N	5	5			Expected N	5	5	
	Residual	4	-4			Residual	-5	5	
TP13D2D3		0	1	Total	TP14D2D3		0	1	Total
	Observed N	2	7	9		Observed N	9	1	10
	Expected N	4,5	4,5			Expected N	5	5	
	Residual	-2,5	2,5			Residual	4	-4	
	Chi-Square	df	Asymp. Sig.		Chi-Square	df	Asymp. Sig.		
TP13D1D2	1	1	0,317	TP14D1D2	0,667	1	0,414		
TP13D3D1	6,4	1	0,011	TP14D3D1	10	1	0,002		
TP13D2D3	2,778	1	0,096	TP14D2D3	6,4	1	0,011		

The gray cells in the table below indicate that statistical significant differences were found between the modeling alternatives.

Table 6 – Task conclusions

Task topic	TaskID	Diagram	P	Hypothesis testing
Conceptual containment	TP8	TP8D1D2	0,527	Keep H0
		TP8D3D1	0,058	
		TP8D2D3	0,527	
Magnitude of unwanted incident	TP9	TP9D1D2	0,059	Reject H0. D3 (and D1) are preferred over D2.
		TP9D3D1	0,206	
		TP9D2D3	0,003	
Likelihood	TP10	TP10D1D2	0,317	Keep H0
Logical "and"	TP11	TP11D1D2	0,096	Reject H0. D1 preferred over D3
		TP11D3D1	0,034	
		TP11D2D3	0,157	
Shared vulnerabilities	TP12	TP12D1D2	0,034	Reject H0. D2 is preferred
Magnitude of risk	TP13	TP13D1D2	0,317	Reject H0. D3 is preferred over D1, but not significantly over D2.
		TP13D3D1	0,011	
		TP13D2D3	0,096	
One incident, multiple risks	TP14	TP14D1D2	0,414	Reject H0. D1 and D2 are preferred over D3.
		TP14D3D1	0,002	
		TP14D2D3	0,011	

In TP8 none of the modeling alternatives were significantly preferred. The reason may be that they all have weaknesses that will make them difficult to use in practice. The first alternative will, although conforming to so-called best practice, become impossible to use when the diagram scales due to its space consuming design. The other two diagram alternatives are based on arrows to show inclusion, but still this aspect may be difficult to see and understand for the reader.

The third diagram alternative in TP9 was preferred over the two others, which shows that text labels are preferred over both size and color to illustrate magnitude.

Two alternatives for illustrating the likelihood of different paths were tested in TP10. Neither line thickness nor line type were preferred for this purpose.

Logical and-gates are often used in traditional risk modeling. In TP11 we tested three alternatives for illustrating this. We conclude that the one resembling the original and-gate symbol the most was preferred over the two other alternatives.

When several assets share the same vulnerability it may be useful to show it explicitly. In TP12 an alternative to using the original CORAS UML profile, in which vulnerabilities are placed below each asset, was compared to one that distributed vulnerabilities throughout the graph. The original CORAS UML profile version was preferred, probably because the students were more familiar with this notation and also because this notation is easier to read than the other when one is interested in looking at the assets and their vulnerabilities only.

In TP13 we tested three alternative representations of risks with various magnitudes to find the preferred one. The third diagram alternative was preferred over the first, but not significantly over diagram 2. This means that for this task, a modeling alternative close to the CORAS UML profile was the least preferred modeling alternative. A similar result was found in TP14 where the notation close to the CORAS UML profile (diagram 3) was the least preferred among the models.

C.IV Threats to validity for the prestudy

In general, humans who get involved in a risk analysis are rarely familiar with the analysis concepts or the process used. Concepts that are intuitive to students are often also intuitive to professionals. We therefore believe that our use of students in this explorative investigation has little effect on the validity of the results.

The student received the concept definitions belonging to the original CORAS UML profile [33]. However, many of the modeling alternatives used a new type of notation. This might have been confusing since the UML profile notation was more familiar. Nevertheless, it was often the new modeling notation was preferred over the original notation, something that supports our assumption that the CORAS UML profile may be improved.

The number of subjects was limited (10-11 respondents) and we therefore draw no strong conclusions from this prestudy. Nevertheless, the findings will be used as supportive evidence when interpreting the results from the next investigation where professionals will be used as subjects.

C.V Conclusion

Although the prestudy results are based on a limited number of subjects, the results are used as an exploratory study. The findings form the basis both for the material that will be tested in the main study, as well as for a control of the study set-up.

The main findings from this prestudy were:

- The original CORAS UML notation is not always preferred (particularly for the space consuming representations).
 - Text labels seem to be preferred over more graphical means.
-

- If a symbol for a concept already exists in a related modeling notation, one should investigate this alternative before developing other modeling alternatives (i.e. for the and-gate, and consequently also the or-gate).
-