Abstract

The misuse case methodology is an approach for eliciting security requirements in software development projects. Misuse cases are an extension of the well-known use case methodology, and use the same basic types of diagrams and documentation forms. This thesis presents a study of the introduction and application of the misuse case methodology in a development team in the computer software industry. A qualitative research approach, with workshops and interviews has been applied to determine the usability and effectiveness of misuse cases. In addition, the practioners' perception of the method has been investigated, as this is believed to be an important factor for the adoption of the method in the team's engineering process. The results show that the method was easy to learn, easy to use and gave a good result compared to the time and resources used.

Contents

1 Introduction

The requirements engineering phase of a software development project consists of elicitation of requirements, analysis and specification. Defining what the customer want and need. Requirements are classified as being either functional or non-functional. The IEEE standard glossary of software engineering terminology [14] defines functional requirements as "a requirement that specifies a function that a system or system component must be able to perform". Non-functional requirements are the more subjective properties of a system, such as usability, interoperability, reliability and security. This thesis is concerned with the latter, the role of security in requirements engineering.

Traditionally, there has been a lack of focus on security-related issues in the requirement gathering phase [7]. It is argued [24] that traditional methods and standards for security requirements engineering are too heavyweight and not easily understood by software developers. To elicit functional requirements, the use case methodology is commonly in use. Since use cases have limited support for eliciting security requirements, a method called *misuse cases* [23] has been developed. The misuse case approach is based on the well-known use case methodology, and uses many of the same tools and procedures. This thesis applies and evaluates the *misuse case* methodology on a real-world software project.

1.1 Use cases

In software design, a technique known as *use cases* [15] has gained increasing popularity since it was first proposed by I. Jacobson in 1986. Use cases have their role in the requirements modeling phase of the design and provide a way to describe, but also explore, functional requirements of a system.

A use case is a structured way to address how a user or another system interacts with a system. When modeling with use cases, it is common to start with a use case diagram that depicts the actors, use case(s) and interaction(s). The diagram is only to give a brief overview of the use case, but will give both the author and the reader an instant feeling how the user relates to the system. An example of the diagram form is show in figure 1.

The most important part of the use case is the textual description. The description is written in a pre-defined, and partly standardized, form. This form usually contains an identification

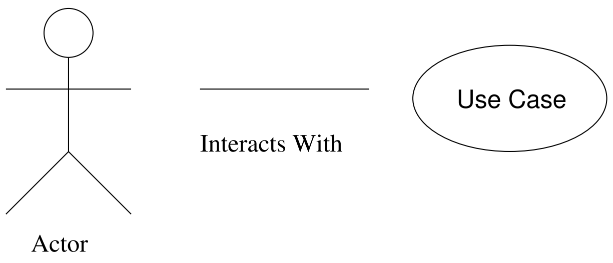

Figure 1: Use case diagram

section, a step-wise descriptive section, information on applicability of the use case, assumptions, preconditions, exception handling and more.

When used correctly, the use case will improve communication between the stakeholders and the software designers, and give an assurance that the design really solves the task.

An example of a use case diagram including 4 different use cases is shown in figure 2.

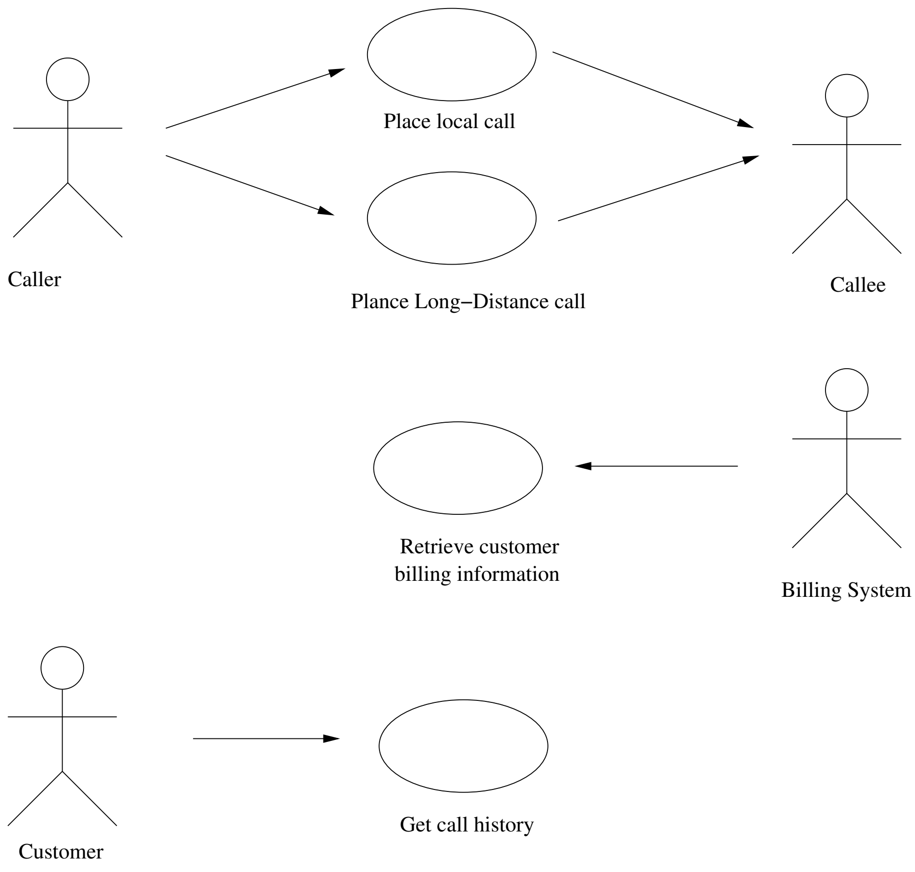

Figure 2: Example of a use case diagram. Adapted from Bittner and Spence [4]

For a text-book treatise on the use case methodology, *Use Case Modeling* by Bittner and Spence [4] can be used.

1.2 Misuse cases

Use cases are primarily suited to explore and document functional requirements. Non-functional requirements are less tangible than their functional counterparts, and it seems like they're more difficult to specify. The more prominent areas of non-functional requirements are: Usability, efficiency, scalability, reliability and security. These requirements are often measurable, but only after the product is completed. It's more difficult to define goals, and to describe how these goals are to be met in the design process. As such, the non-functional requirements are often viewed secondary to the functional requirements.

In this work, the focus has been on the security aspect of the non-functional requirements. It is seen that security concerns are something that is rarely addressed until very late in the development cycle. More often than not, it seems to be introduced as an after-thought, or after specific customer requirements.

By many practitioners, security is seen as a separate profession. It has its own theoretical foundation, use its own techniques and procedures, non which are seen as easily adoptable by software designers.

A technique, known as *misuse cases* [23] has been developed to overcome this short-fall. A misuse case is the negative form of a use case, and focuses on threats to a system rather than its functionality. With its similarities to the better known, and widely used, use case technique, the misuse case method could be a helpful tool to address the problem where security issues are not taken into account among software architects and developers.

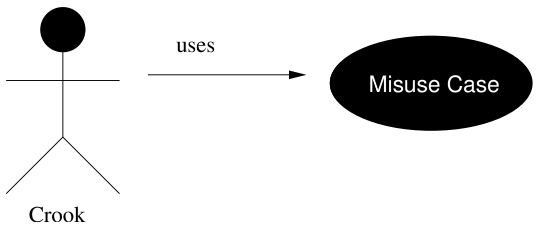

Figure 3: The simplest misuse case

As for the use case, the misuse case method consists of both a graphical and a textual representation. The graphical representation (figure 3) is used to give a quick overview of the misuse case. Elements used are the *misactor*, or “crook”, the misuse case itself, and interaction arrows between the misactor and the misuse case. The difference in the diagram form is that the misactor and the misuse case has inverted colors, to distinguish them from the use case. The textual representation has an identification section and a descriptive section, similar to the form used by the use case. An example of a misuse case form is shown in table 1.2, showing some of the fields used. The complete misuse case form, as used in this study, can be found in Appendix A.

Misuse case ID	SA-006
Case Name	Brute-force password attack
Summary	The misuser gains access to the system by trying large sets of passwords
Author	N.N
Date	2005-04-11
Basic path	bp-1 The crook installs the client connection driver bp-2 In a loop, the crook attempts to connect to the server, using one of the documented pre-defined users, using guessed passwords
Alternative paths	ap-1 In step 2, the crook may create multiple connection threads to reduce the time required
Mitigation points	mp-1 at action 2, there is a connection driver timer that delays the attempts. This will not prevent the misuse, but delay it considerably.
Triggers	Always TRUE. This can happen at any time.
Preconditions	None
Assumptions	[..]
Mitigation guarantee	[..]
Related business rules	[..]
Potential misuser profile	Programmer. Knows the published programming interface
Stakeholders and threats	st-1 Possible loss of data, st-2 possible disclosure of data, st-3 possible alteration of data. May disrupt business and affect customer relations

Table 1: Example misuse case form

Misuse cases and use cases can easily be combined into one. Figure 4 shows how the diagram form may look. In the textual description, a misuse case can either be embedded in a use case, or it can be a stand-alone form.

When misuse cases are embedded in a use case, a new field called “Threats” are added to the use case form. A use case may have more than one threat. In this work, only the stand-alone form is considered.

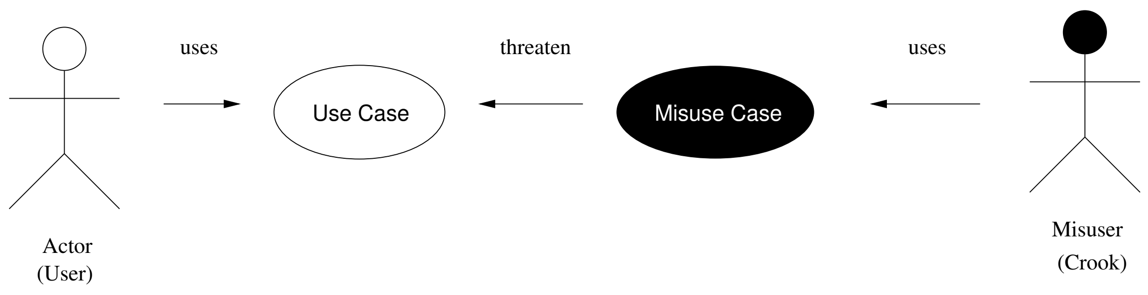

Figure 4: Use and misuse case

A five-step approach for determining and specifying security requirements with misuse cases has been proposed by Sindre and Opdahl [24]:

1. Identify critical assets
2. Define security goals
3. Identify threats
4. Identify and analyze risks
5. Define security requirements

An asset is something of value to the owner or stakeholder. What is perceived as an asset can differ from case to case. Examples are system processes, sub-systems and information items. For threat analysis the misuse case diagrams and forms discussed above are utilized. The identification of possible misactors and their possible harm to the system is used as input to the fourth step where each threat is analyzed based on the potential harm to the stakeholders. A cost/benefit analysis might be necessary to decide which threats should be addressed in the security requirements.

A related modeling technique is the *security use case*[11]. While misuse cases are used to explore and document security threats, the security use cases are used to specify and analyze security requirements. Misuse cases and security use cases can be seen as complementary techniques, where the security use case is used to mitigate the threat documented by the misuse case.

Misuse cases has been discussed in the literature, most notably in [23] [24] [10]. A textbook introduction can be found in [2].

1.3 Other concepts

Misuse cases are used to describe threats to a system's security, where a *threat* is the possibility for a breach in the system's security. Software security is often defined as the union of seven areas, *authentication*, *authorization*, *availability*, *integrity*, *confidentiality*, *auditing*, and *non-repudiation*.

Auditing: Recording of system activity to verify correct behavior of the system and its users.

Authentication: Verification of the identity of users.

Authorization: Controlling access to assets and resources.

Availability: The system is available for use, and responds timely, in spite of malicious attacks.

Confidentiality: Securing that only authorized personnel and systems have access to assets and resources.

Integrity: Preservation of data, so only necessary and consistent changes are allowed.

Non-repudiation: Ensuring that the identity of a user is irrefutably verified. Ensuring that a user can't deny a previous action.

1.4 Project motivation

To verify that the misuse case method is usable as a tool for practioners in software development, the method has to be applied in real-world settings. Applying the method in an industrial setting, observing the use by practioners and assessing the results gained, can certify that the method is useful and is a contribution to the software engineering field.

Actual use of the misuse case approach has so far been limited. This thesis will apply the method to an industry project with the aim to answer:

- Is the misuse case method easy to learn? Adaption of a method can be dependent on how much effort is needed to learn the method, i.e how much training is necessary before a practitioner can successfully use the method.

- Is the method perceived to be easy to use? A method that is perceived to be easy to use will probably be more popular among the users. If the method is perceived to have a high overhead, such as unnecessary method steps and verbose forms, it is believed that the method will be more likely to be rejected by the practioners.
- Is the method effective? A method will be deemed as effective if it produces relevant output within given time and resource constraints.

1.5 Setting of the study

This study is performed within one software product development group in a large multi-national company (> 30.000 employees). The development group has emerged from work done by research groups, both at the university and independent research institutions. It has still close links to academia, and it is fair to say that the focus of work in many ways resembles more research activities than "big business". The group consists of 34 software developers, with a gender composition showing 3 females and 31 males. The age composition is shown in figure 5.

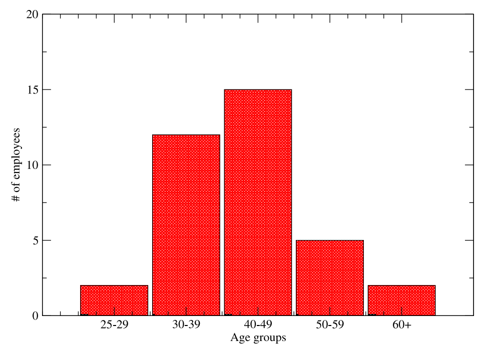

Figure 5: Age composition of development group

The group has development and sustaining engineering responsibility for two products. Each of the products is in the same size and complexity range, having between 1.5 and 2.2 MSLOC (million source lines of code). This is in the same order as a modern operating system kernel. For comparison, the Linux 2.4 kernel has approx 2.4 MSLOC and the XFree86 window system 1.8MSLOC [28].

In the approx. 10 years the group has existed, getting a product out to the market has been the prime focus. Functionality, efficiency and stability have been the major concerns for the product. Other non-functional areas, such as security, has not been high on the agenda. All threat countermeasures, except simple user authentication (username/password), have been applied retrospectively. Neither a formal nor an informal method of security threat analysis has been used. In some cases, customer feedback and corporate-wide guidelines have resulted in retrofitting certain security mechanisms. In short, little or no effort has been made to determine security requirements up-front in this organization.

Market analysis has shown that the customers and potential customers of the product line of this development group has an increasing focus on secure software. This has raised interest in security issues in the company at large, resulting in requirements to each product group to perform some kind of security assessment of their product line. The present requirements are informal and vague, and no methodology is currently suggested nor required for the determination and validation of the requirements.

1.6 Structure of this thesis

Chapter 1 is an introduction to the concepts of use cases and misuse cases. A motivation for the work is given.

Chapter 2 is a treatise of the research methods used to conduct this study.

Chapter 3 is a description of the first case study.

Chapter 4 is a description of the second case study.

Chapter 5 is a discussion on the findings of the case studies

Chapter 6 gives the conclusions and suggestions for further work

2 Methods

The most common way to classify research methods is among the quantitative methods and the qualitative methods. The choice of approach is guided by what we want to study and our ability to control the setting. The quantitative approach is developed in the natural sciences and several methods are in use, but key factors are to study a large representative selection of a population, application of statistical methods to the data, and falsifiable hypothesis. The qualitative methods are more aimed at studying people, people's interaction with each-other, and social and cultural phenomena.

The aim of this study has been to introduce a new methodology to a software development team and evaluate its use by the team. Determining if the methodology works, and more importantly, why it works, can be seen as the principal goal. To be able to do this assessment, the methodology has been put to use on a specific project initiated by the company.

It is inherently difficult to use quantitative methods when you evaluate methodologies for software design and development. The need for a controlled environment with many practitioners, and the need to redo the experiment with changed parameters is impractical, and in most cases impossible. In this work only a limited number of practitioners were available, and the study had to be completed within tight time and resource constraints, so a qualitative method, using a case study with observation and interviews were considered to be more likely to succeed.

Robson[21] further classifies qualitative research in being either post-positivistic, constructivistic, or critical. The constructivistic approach is also known as interpretive, and can be identified by the notion that:

> People, unlike the objects of the natural world, are conscious purposive actors who have ideas about their world and attach meaning to what is going on around them. In particular, their behavior depends crucially on these ideas and meanings.

To understand the behavior of humans, these insights must be taken into account, and their actions must be interpreted based on the socio-historical context in which these humans live and work.

Klein and Myers [16] propose a set of seven principles for evaluating interpretive field research. These principles has been used as a guideline in this work.

The first principle, known as the fundamental principle of the hermeneutic circle suggests that you can't understand a system or a process by looking at only its whole or only its details. Understanding comes from a systematic iteration where you combine the investigation of the process' particulars, how each part of the process work and is perceived by the participants, with the generals, how the process looks from the outside. In this study, the first principle is applied by examining the observations from workshops and meetings, the produced written material, all of which constitute the parts of the process, with the interview data and the result product, which represents the whole.

Klein and Myers' second principle, the principle of contextualization, states the importance of understanding the context in which research data is gathered. We need to gain an understanding of the participants' background and work culture to adequately understand their response. Since the team members and the observer have worked together previously on several projects, a common background is shared. Sharing a common context should be helpful in the interpretation of the observations.

The third principle, the principle of interaction between the researcher and the subjects, says that when conducting observational qualitative research, the relationship between the observer and the participants must be determined and clear.

Constable et al [6] classifies three observational roles, the neutral observer, the observing-participant, and the participant-observer. The participant-observer is usually an outsider who aim to come in and be a team member while still keeping the outsiders view and having a different contextual background. The observing-participant is someone who is already affiliated with the team and has connections with the participants. The neutral observer is someone who do not interfere with the discussions and the actions of the participant, a "fly on the wall".

During this work, there was no difference in the role of the observer and the other team members. The observer was as much a team member and participant as the others. In the classification by Constable et al, the role of the observer has been that of an observing-participant. The observing-participant must always question his objectivity and his influence on the other participants and the setting. The balance between active participation and observation can be problematic to achieve, and great care were taken to understand how this affects the behavior of the other participants and the results.

The fourth principle, the principle of abstraction and generalization, requires us to make an effort to abstract and generalize the findings in a study. Since the study is limited in scope, involving two cases and few practitioners, effort must be taken to find the pieces that can be used to explain general concepts.

The researcher carries with him a background, an intellectual history, that forms preconceptions about the study and the participants. The fifth principle, the principle of dialogical reasoning, addresses this and requires the researcher to "confront his or her preconceptions that guided the original research with the data that emerge through the research process" ([16] pg. 76).

The sixth principle, the principle of multiple interpretations, require the researcher to try to find conflicts or different versions in the view of the examined process. Since this study was limited to two cases, using a small number of participants, it was difficult to find contradictions.

The principle of suspicion suggests that what you hear and see from the participant should not be taken at face value. The researcher must always look behind the mask and understand why something is said or done. Just as important is what is left out. Knowingly or unknowingly, the participants may leave out parts that are difficult to talk about or details they perceive as not being relevant, although they are.

2.1 Workshop session

The suggested process of eliciting security requirements with misuse cases started with an identification of the critical assets and defining security goals for each of these. A workshop was chosen as the arena for this exploratory phase.

A semi-structured agenda was considered the best suited approach as the complex nature of interconnections between different sub-system in a large software system are best covered by an open brainstorm-like discussion. It is, of course, important that every asset is explored systematically, but on the other hand the open discussion and interaction between the participants could uncover details not easily seen by each single participant on his own.

The misuse case form was introduced in the third step of the process, the identification of threats. An introduction to the form was given at this time. The initial plan was to describe some of the threats in full session and the remaining by each participant on his own. It was, however, seen that two participants working together was a more efficient approach, so this was chosen instead of each working alone.

2.2 Interview techniques

After the workshop and misuse case documents was finished, interview sessions was conducted with each of the participants. The goal of these interviews was to gain further insight in how the method was perceived and to get feedback on improvements.

Robson [21] classifies three types of research interviews, the fully structured interview, the semi-structured interview and the unstructured interview. The fully structured interview has a set of pre-determined questions, and resembles a survey questionnaire commonly used in quantitative research. Even if the answers are free-form, this interview style can give you quantifiable results which is useful in a quantitative study. However, the pre-determined questions are restrictive as they prohibit new questions based on the response on a previous question.

The semi-structured interview is based on some predetermined questions, with the possibility of extending the questions based on the response. It is common to have a set of wide questions giving the interview a structure, and ensuring that all predetermined areas are covered. In between these, the interview takes more the form of a conversation where the response can be used to form the next question in a sort of a feedback loop.

The unstructured interview is non-standardized and open-ended. The entire interview has the form of a conversation. Robson suggests that this technique is not something the unexperienced researcher should use as it can be difficult to drive the interview in a productive direction.

In this study there was an interest in a deeper understanding of how the participants perceived the use of the misuse case methodology. The number of participants were small, and a semi-structured approach was thus seen as the best choice for the interview part of this study.

2.3 Preparations

The engineering staff for the projects documented in case 1 and 2 were primarily using StarOffice^{TM} and $\LaTeX$ for document preparations. It was seen as important to provide templates for misuse case descriptions for both these environments to ease introduction, and to save time (appendix A). In addition, a helper document with description of each field in the template (appendix B), and an example misuse case description was provided as a help.

A conference room equipped with whiteboard, flip-charts and computer with projector was allocated for the “all-hands” sessions. For the interview sessions, the office of the interviewee was preferred as this was believed to ease any tension and provide a quiet, undisturbed, athmosphere.

3 Case 1 - Misuse cases in the requirements gathering phase

A new product idea was launched in the company to fill a niche in the market of financial and telecommunications sector, with particular focus on medium and large size enterprises. The product would have a distributed and replicated design using multiple inter-connected nodes, and having high efficiency, reliability and scalability as key factors. A simplified block diagram of the system is shown in figure 6 for illustrative purposes. Details on the product idea and architecture is withheld of confidentiality reasons. To be considered by many of the potential customers in the targeted market segment, it was believed that at some level a secure design would have to be documented. As such, this project was chosen as a target for a misuse case based analysis.

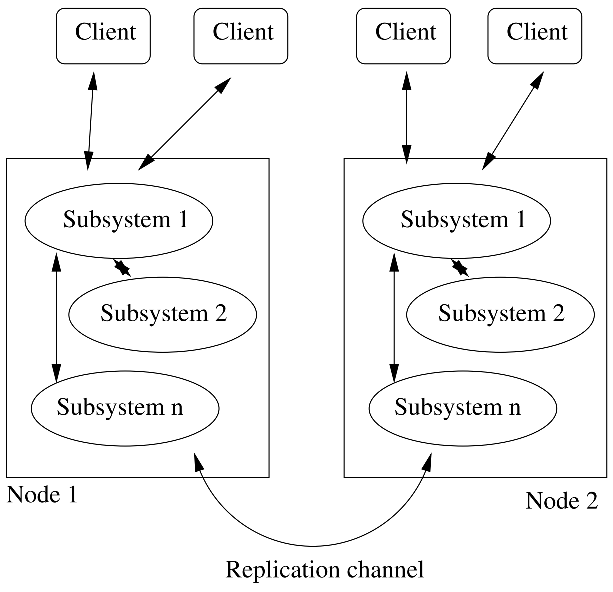

Figure 6: System block diagram (simplified)

3.1 Workshop

The workshop was set with 10 participants. The invitation to join was open, but it was ensured that expertise with all the functional areas of the product were present. Most of the senior architects were present, together with engineering staff and a representative from the QE/QA department. All the participants had at least a cursory knowledge of use case modeling, and 3 had extensive experience using it as part of a design process.

The session started with an introduction where the mandate was presented - to explore and document potential security threats to the product. The participants were informed that the workshop would serve an additional purpose, being a part of this study. One person was appointed as a note-taker to document the findings. His notes, in addition to private observation notes, constitute the documentation from this workshop.

The architect team presented the preliminary design, and a set of six high-level system components that constituted a logical division of responsibility in the system were identified. These components were felt as the most logical *assets* for the purpose of the analysis. Each of the components were examined in turn, identifying the communication channels between the components and between each component and the outside. Having a clear view on the interfaces between the sub-systems, between the system and the user space, and between the system and the operating platform was perceived as helpful for better being able to identify both the goals and the threats.

Definition of the security *goals* were done in an iterative process which included an initial threat assessment. The consensus in the meeting was that by looking at potential threat scenarios the goals was easier to identify and name. However, in retrospect the use of a standardized typology for classification of goals could have resulted in a faster process, and given a higher confidence in the goal coverage, Examples of the ad-hoc defined goals identified by the meeting are “stop unauthorized client access”, “ensure non-repudiability of data entries”, and “hinder eavesdropping”.

A smaller group was seen as more appropriate for working out the details of the threat analysis. Exploring and documenting the misuse cases was thus decided to be postponed to a later meeting. The findings were to be submitted to other members of the workshop for review and quality assurance. The workshop ended after two hours of work, with its objective complete.

3.2 The capricious nature of Big Business

A week after the initial workshop was conducted, and before the group that should work out the misuse case details started to work, the project was put on hold. Changes in the business requirements for the product had mandated an architectural redesign process. Since security analysis is highly dependent on the system architecture, any further work with exploration of the security threats had to be postponed. The plan was to let the architect group work for 4-6 weeks, and then reconvene to continue. This estimate proved to be overly optimistic and a date for the restart of the project was postponed further and further.

After a period of about six months, with no opportunity to continue the work, the architecture group finished their redesign. A request for resources to start up the requirements gathering process was accepted by the management, and re-staffing was started. An allocation for 3-4 engineers to complete the security analysis was granted, and a staff reallocation process was initiated.

By the time the reorganization was complete, a change in plans was announced from a higher level of management. The resource situation mandated that the team was given other priorities, and local management had no choice but to cancel the entire project. All work on the project, including this study, was stopped.

4 Case 2 - Evaluation and retrofitting of security features

Since the first case had to be abandoned before all parts of the misuse case methodology could be applied, a new case study was initiated. The company have a product which had been a niche product for about 8 years, but was now being rolled out to a larger number of customers. An increasing number of potential customers had signaled that software security was a high priority. Some customers would also require a documented evaluation of security in their software base. A new major release of the product was in planning, and a security evaluation was a part of the project plan. This work was suggested as to fill a part of the process, evaluating security threats and suggesting changes to the product.

The mandate of the evaluation process, from the company's point of view, was to capture, explore and document the security risks facing the product. The documentation was then to be used both as an input to the evaluation, but also to suggest changes to the product. A misuse case approach was deemed appropriate for the first part (capture, explore, document), as this resembles the process you're undergoing during initial requirements gathering.

The product in question is a distributed enterprise-class system with customers primarily in the financial and telecommunications sectors. Core architectural characteristics are distribution over multiple hosts, high demand for uptime, and fast fail-over of services in case of failure. Further details on the product itself are withheld of confidentiality reasons.

Due to resource constraints, a smaller number of people were assigned to do this work than in Case 1. A group of 3 mid-level engineers from the software development department were asked to participate. Although the number of people was less than in the first case, the team had broad product expertise so all functional areas of the product were covered. Additional persons from the development group were also available in case they were needed.

4.1 Workshop

The workshop was conducted over a period of two days, where the team undertook the work of determining what the assets were, which security goals were important, and what the risk for each of the assets were.

The first session started by drawing a system diagram on the whiteboard, identifying each component and the basic interaction between these components. The product consists of two or more nodes (see figure 7 interconnected at several levels, leaving multiple entry-points into the system.

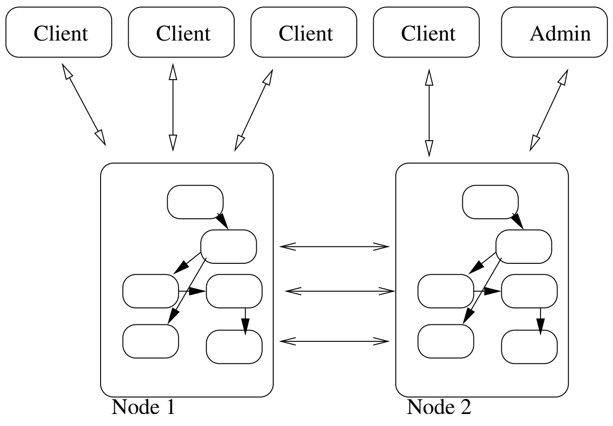

Figure 7: System diagram

Each node consists of a set of system processes communicating both with each-other, and with corresponding processes on the neighbor node(s) (see figure 8). The participants chose these system processes to be the *assets* for the purpose of the misuse case evaluation.

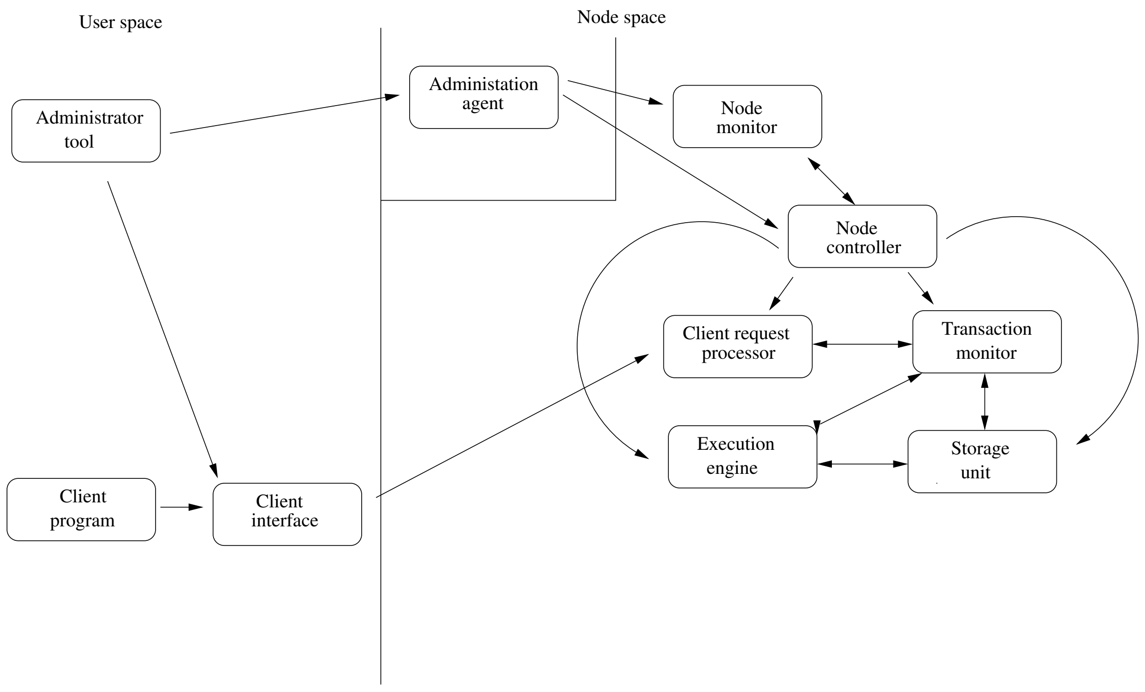

Figure 8: Single node diagram

After the assets were identified, a set of security goals were found for each of the assets. In [24], it is suggested that a standard typology, as the Common Criteria [5], is used as a base for the goal determination. Instead of using a standard typology, an ad-hoc method was chosen for this case, as it was concluded that it would require too much time to familiarize the participants with the Common Criteria. The knowledge the participants had with the product and the typical customer operating environment was deemed as good enough to get a sufficiently good list of goals. Some of the goals are shown in table 2.

Id	**Description**
AUDT01	Ensure that all management operations are logged
AUTH01	Ensure that only authorized users get access
COMM01	Prevent eavesdropping
COMM02	Prevent communication tampering
COMM03	Prevent denial-of-service attack
RES01	Ensure that disk resources are not exhausted
RES02	Ensure that memory resources are not exhausted

Table 2: Examples of ad-hoc determined security goals

4.2 Misuse cases

An introduction to the misuse case description forms was given to the participants in an all-hands meeting. This introduction took use cases as a starting point and developed the ideas of negative use cases based on the participants previous experience with use cases.

Two misuse cases were described, as part of the learning process, with the whole team present. By doing this, any questions concerning the form and the process could be addressed and discussed. An aspect of this study was to gain insight in how easy it would be for the participants to use the method on their own without direct guidance, so the rest of the misuse cases were described by the participants in pairs.

The finished misuse case descriptions were used as input to the larger requirements process for the new product release. The intention was to use them to drive specific requirements for changes to the product, and as support documentation for a security manual.

4.3 Interviews

Two weeks after the completion of workshop and the misuse case descriptions, the participants were interviewed. The purpose of the interview sessions was the get a better understanding of the participants perception of the misuse case methodology, and the way it had been applied in this project. Each of the interviews were done separately, in the participant's own office. None of the interviews were taped, so instead extensive note-taking was used for documentation.

Each of the interviews lasted between 35 and 45 minutes, and included topics as previous experience with use case modeling, knowledge of security threat analysis, reactions to the workshop and misuse case exploration, suitability of the method, and suggestions for improvement.

5 Results and Discussion

When evaluating a method for use in a software development process, there are several important aspects to take into consideration. Does the new method fill a need in such a way that it is useful for the team that applies it? Is it easy to use? What amount of training is required to successfully apply it? Does the method give good results compared to the time and resources investing in using it, e.g cost efficiency, and does it perform well compared to similar methods?

This evaluation of the misuse case method took place during five sessions. Three of the sessions were workshops, each lasting between one and two hours. Two were meetings where misuse case descriptions were worked out, by a single participant alone, or pairs of participants working together. In addition, post-task interview sessions where held with each of the participants, each interview lasting between 35 and 45 minutes.

5.1 Current state of affairs

The interviews gave clear evidence that security requirements have not been taken seriously in previous projects, and security evaluation has not been an issue neither during requirements gathering nor during the design phase. Neither a formal nor an informal security analysis methodology has been described and mandated in the organization's engineering process documents.

Some quotes were recorded in the interview sessions that could shed some lights on the reasons why this has been omitted:

> We didn't know what the customer security requirements will be

> Security wasn't deemed important in that product (on a previous project)

> We needed to shorten the time-to-market

These quotes indicate both a resource problem and a cultural problem. There seems to be an assumption that the customers are functionality-driven, and not interested in security. Security in a system isn't easily quantifiable, and might thus not be a selling-point except for very specific groups of customers such as defense, financial institutions and telecommunication operators.

Leaving security analysis out of the project plan give room for other activities perceived to be more important for the potential customer.

Further, it was clear from the interviews that security analysis of software had not been discussed at any length in the university classes the participants took. When asked specific questions on knowledge of a set of particular methodologies, the Common Criteria[5], $i*$ [17] and ISO17799/BS7799 [12], none of these were recognized by any of the participants. A check with three [19][3][15] of the text books used in various software engineering classes at the Norwegian University of Technology and Science, where all the participants have graduated from, showed that functional requirements gathering was covered with multiple chapters in each book, but non-functionals were hardly covered at all. None of the books go through any methods for eliciting security requirements. This suggests that lack of coverage during education can be a reason to why this part of requirements gathering is neglected in the industry.

Which of the possible reasons, time constraints, lack of a method or little knowledge in the field of security requirements is the more prominent might have become clear by extending the interviews to also include product managers and project owners. It was not possible to arrange such interview sessions within the scope of this project, but this could be interesting as a follow-up.

5.2 Initial reactions

It could be expected that introducing a new methodology would meet some resistance. This turned out not to be the case with this team. None of the participants showed any reluctance, and a positive attitude was observed during the sessions. It could be that this is because the participants was curious on a new method, or simply that they wanted to show courtesy. Another interpretation is that they felt a security requirements process was long overdue.

Both observational data and interview response indicated that the problem area was well understood by the participants, in the sense that they saw that the current situation was a problem. Quality issues, in a wide sense, had been an important issue in the company as a whole, in particular the last two years, and official company information hints that there had been an increase in feedback from unsatisfied customers. A decrease in product quality had raised the costs on support, bug fixing and the publishing of patch-releases to a level where it was deemed necessary to do something. Increased focus on quality factors like reliability, security and availability had thus become a mantra to the entire company, even though it hadn't yet caused any change in the local engineering process.

Having a defined method available as part of the engineering toolbox would probably be helpful, independent of which method this is. When the staff has experience using such a method it would be easier to incorporate it into the engineering process. It can also be more easily entered into the project plans, with time and resources allotted.

5.3 Workshop

Arranging a workshop to initiate the process was felt by all the participants to be a good idea. Getting the team together in an open, brain storming like, dialogue enabled them to quickly reach a common understanding of the problem and establish a common nomenclature.

There are several important aspects concerning how to compose team a for a workshop. You need to strike a balance between having too few to cover the functional areas of the product, and too many where you risk a loss of focus.

The first workshop (Case 1) had representatives from different stakeholder groups, product management, architects, engineering and QA. A complex group composition could help in letting the workshop see new problems and new solutions. In his study of misuse cases for trade-off analysis, Alexander found that the broad participation lifted the discussion out from a deadlock, and "the workshop moved into a free space in which it could think creatively" ([1] pg. 67).

This positive exchange of ideas was not achieved in this study's Case 1. The most probable cause for this is the difference in seniority between the participants. The social hierarchy differences became prominent as the more senior architects dominated the discussion, and this made it difficult to have an open dialog. As a consequence, a lot of time was spent on discussing subtle architectural details with minimal impact on the result. The more junior participants were completely passified and did not contribute to the discussion at all, even though they also represented stakeholder groups. The chairperson could have kept a more stringent control of the discussion, giving time to each participant. Such an intervention could, however, have dampened the discussion even more, and been counter-productive.

In the second case, a smaller number of people participated. In contrast to Case 1, the participants were from a more equal level in the organization. This turned out to be very effective in the sense that each sub-task was completed in relatively short time and with few diversions into details. Each of the participants took part in the discussion, and no single person or group dominated. However, there is a risk that not all the relevant threats were found, and that a larger participation could have uncovered more.

There is no fixed number of participants that will suit every project. While this study had best results with a small group, this is not believed to be a universal truth. Combined knowledge to cover all the functional areas of the product has to be involved, and you need to ensure that focus is maintained and that no sub-group takes control of the workshop.

5.4 Goal determination

An unsuccessful attempt was made to apply the typology of the Common Criteria [5] in the goal determination phase. The Common Criteria defines 11 functional classes, 65 families and 135 components of security requirements, so it is extensive and require training to be able to use. Once mastered, it could be very useful for establishing a common language for goal descriptions, and provide identification for misuse case templates.

There was no available time for training and familiarization with the Common Criteria, thus instead of applying the CC, the goals were determined in an ad-hoc fashion using the participants' knowledge of the system and the typical customer configuration. While these goals reflect important characteristics of the system, it is believed that using a formally defined typology would increase the quality of the goal finding process and ensure better goal coverage.

An after-analysis of the records from the workshop show that the goals match members of the Common Criteria family of security requirements, but the CC states fifty additional requirement families that were not found by the ad-hoc process. Whether this resulted in the omission of relevant threats in the misuse case description phase is not known, but this is probable. It is therefore believed that the use of the Common Criteria, or a similar classification scheme, could be very useful in the goal determination phase.

5.5 Misuse case identification

The threat identification phase of the the second case study showed that many threat scenarios were common to more than one subsystem. This is to be expected from generic threats, such as denial-of-service attacks, but even more specialized threat scenarios showed up in several subsystems. Accounting for this, a total of twelve unique misuse cases were identified and deemed important enough to be explored and described.

The analysis of the completed forms gave an important insight, as it turned out that the four of the threat scenarios could be traced back to a single design flaw. By working out the detailed step-by-step basic path description, the participants were able to discover an underlying problem which none of them had seen before.

It is believed that this new insight was gained because the the participants had to work through the scenarios systematically. Studying the scenario from a different angle-of-view, from the problem side instead of the solution side, is probably what unveiled the root cause.

5.6 Form suitability

The misuse case form used in this study is extensive, and in most cases you will probably use only a subset of the fields. As one of the participants stated in the interview:

> The form was a bit "overkill", but it probably constitutes a least common denominator.

The participant could not point at any one of the fields and say that it was useless. It was felt that all could be useful in different settings.

All the misuse case forms where post-processed to get an understanding of which fields were used. Figure 3 shows the usage frequency of the fields. As would be expected, the identifying header fields were present in every form along with the step-by-step description ("basic path") and a list of the preconditions that apply.

Four field were never used, *extension points*, *related business rules*, *technology and data variations*, and *terminology and explanations*. Extension points was felt as being potentially useful, but not used in this case. Business rules is a term the participants associated with commerce systems, where each installation has a set of rules that control the behavior of that particular system. The system examined in this study has no direct equivalent of such a rule set which may explain why this field was not used. Terminology and data variations did not find any use, either because the threat scenario was present on all platforms, or because all terms was felt as being obvious and a part of the common knowledge in the group. If the misuse cases were to be interchanged between different groups, this field could be more useful as the socio-technical context in which each group operate differs.

Always	**50-99%**	**1-49%**	**Never**
Case ID	Alternate path	Mitigation guarantee	Extension points
Name	Mitigation points		Rel. business rules
Author	Triggers		
Date	Assumptions		Technology
Summary	Pot. misuser prof.		Terminology
Basic path	Stakeh. and risks		
Preconditions			

Table 3: Frequency of used fields

5.7 Productivity

Productivity can be difficult to ascertain. Different definitions give raise to different ways to measure productivity. In this work productivity was chosen to be defined as “how many relevant misuse cases can be produced by a number of people in a given time”. Every project works within time and resource constraints so a method that gives good results with a small resource consumption will be attractive.

For Case 1, a ten minute overview of the method was given. The different steps of the method were briefly discussed to give the participants a context to work in. An in-depth explanation of the misuse case diagrams and forms was not given since the project was abandoned before this step was reached.

In Case 2, the same short overview of the method was given. Before each step of the method, a more detailed description of the purpose and procedure was given. A more detailed description of the misuse case form was given in an extra fifteen minutes. This introduction contained a demonstration of the misuse case diagram, the misuse case form, identifying the different fields and how to use them in the project. Active use of the form was assumed to be the superior way to learn how to use it, so two cases were described collectively as a practice. As the method is similar to the use case methodology, already established in the group, and since many of the same practices are shared, the team could get started easily . An intuitive understanding was reached easily, so good productivity could be reached in a short time. However, in a setting where the participants are not familiar with the use case methodology a longer introduction and training period might be needed, and more time and exercise will be required to become productive.

A quantitative comparison of the time and resource efficiency between use cases and misuse cases could have been an interesting topic. However, no empirical research on the effectivity of use cases was found, as was also the result of a review done by Dobing and Parsons [9].

5.8 Misuse case recycling

Sindre, Firesmith and Opdahl [22] have proposed an approach for reuse of misuse cases and security use cases. In addition to the potential saving of time and resources, they suggest that reusing misuse cases has other advantages:

> Advantages of the proposed approach include building and managing security knowledge through the shared repository, assuring the quality of security work by reuse, avoiding over-specification and premature design decisions by reuse at the generic level and focusing on security early in the requirements stage of development.

In this study it was found that the misuse cases described pointed at threats that are present in many products. This generic trait of the misuse cases points to an interesting opportunity for recycling of misuse cases. If misuse case descriptions can be reused from one project to another with only minor adjustments it is likely that time and resources can be saved. If there is such a generic trait, a library of misuse cases could be evolved to serve as templates for later projects.

A prototype for a software tool for maintaining a misuse case repository has been developed and described by Munkelien [18]. So far, there exist little empirical research on the use of such a tool.

5.9 Misuse cases for re-engineering

While the misuse case approach was originally designed to explore and document security requirements in the early phases of a software project, the major part of this study has looked at its application on an existing system. In a new project we are concerned with potential threats to a system, trying to counter them before they arise. For an existing system we are concerned with existing and known threats in addition to potential threats.

The examination of an existing product, using the misuse case approach, gave new and important insight. As notices in chapter 4, several design flaws were uncovered. Some of these flaws were known to the participants in advance, but the methodical exploration gave new insight in the potential consequences of the design flaws. This show that the misuse case methodology is useful for use in front of revisions of a system, and not only for the initial design of a system.

5.10 User experience

How the practitioner perceives a method is of importance for the likelihood that the method is used, but also for the quality of the outcome. Positive feelings of a method make it more probable that the method will be used again, and commit the practitioner to an optimal use of it. On the other hand, a reluctance to use the method can be the result if the practitioner has a negative perception, and a sub-optimal use of the method may be the result. Several factors affect the perception, such as how useful the practitioner feels the method is, and how time and resource effective it is felt to be.

As mentioned earlier, in section 5.2, a positive attitude was observed among the participants in this study. During the entire life-span of Case 2, all the participants were active and contributed to the process. The interviews point to the conclusion that the method was felt as useful, and it was perceived as contributing positively to an important, but previously neglected, part of the engineering process. The participants all felt that they had been productive or very productive during the sessions. They felt that the time invested in learning and applying the method gave a good result, so the application of the method was felt as a positive experience.

When asked if they were going to use this method again, on a later project, a more diffident answer was given. Use of a new method in a project requires acceptance and understanding from the project management, and even if they felt successful this first time, it didn't ensure that the method would become a part of the common toolbox. Educating all levels of the engineering organization, including management, about security problems and the method seems to be instrumental to gain acceptance.

The fact that this seems to have been a positive experience for the team make it more probable that the misuse case method will be utilized again. At least as long as there is a perception that it fills a void in the development process and the management agrees to the necessity of utilizing a method to increase security.

5.11 Related studies

So far, there has been few empirical studies on industrial use of misuse case. Alexander did a study in a railway company using misuse cases for trade-off analysis [1]. His findings were that misuse cases helped the group to better focus on the nature of the inter-disciplinary conflicts. Having a workshop, utilizing the combined knowledge of participants from the various stakeholder groups, they were able to get out of a deadlock situation and find solutions not previously seen.

In the same manner as the study by Alexander, this study also found that the workshop revealed knowledge not previously known by any participant alone. In this context, the combined knowledge of the workshop exceeded the sum of each of the participants' knowledge.

The CORAS project has adapted misuse cases for threat scenarios [8] [13]. The technical papers describe how misuse cases fit into the CORAS model for risk assessment of security-critical systems. Unfortunately, there is little discussion of experiences with the misuse case method in the published papers from the field trials [26][25][20]. In one field trial [27], misuse case diagrams are used to facilitate the risk identification process, but there is no discussion on the suitability of the method or the user experiences. Neither is the misuse case diagram identified as such.. It is difficult to interpret the lack of acknowledgment to the misuse case method by the CORAS project. it might indicate that it contributed little to the project or that it became so integrated or effortless that the authors did not deem it necessary to discuss the utilization of the method.

6 Conclusions and further work

This thesis has presented two cases where the misuse case methodology was applied on industry projects. The first case, that unfortunately had to be terminated before completion, attempted to apply misuse cases on a new development project. The second case applied misuse cases in the requirements gathering phase for a new version of an existing product.

This thesis argues that the misuse case method is easy to learn. The methodology was learned by instruction and by using the method. The instructions were brief and most of the learning took place by doing. The similarities between use cases, a method familiar to the participants, and misuse cases seems to have reduced the time required to familiarize with the method. The quality of the misuse cases will probably increase with more experience, but the participants were able to work unassisted with sufficient proficiency after less than an hour of instruction.

Using a workshop to initiate the process was perceived as useful by the participants. The workshop provided the participants with a common understanding of the task. The shared interaction of all the participants gave important contributions to the asset identification, goal determination and initial threat analysis.

Developing the misuse cases in pairs was perceived to be effective. Although the misuse case form was initially felt as being more complex than needed, the form was perceived as being intuitive and easy to use.

The misuse case methodology was designed for requirements gathering in new systems. This study shows that misuse cases are also usable for exploring and describing security threats in an existing software systems. Further, it is shown that the misuse case methodology clearly filled a need within the studied organization, and that it was well received by the practioners participating in this study.

This thesis uncovered reasons why security considerations were not taken into account in previous projects. The lack of knowledge of a method combined with a strong focus on functionality was seen as the major factors.

Further work is needed to explore the use of a standard typology for goal determination. This study suggests that the use of a standard typology could increase the confidence in the results and potentially increase the quality of the work.

Testing the product is an important aspect of software development. One way to develop functional tests for a system is to employ use cases. The common features of use cases and misuse cases suggests that an interesting topic for further study would be how misuse cases could be used as a basis for security test cases.

References

[1] Ian Alexander. Initial industrial experience of misuse cases in trade-off analysis. In *IEEE Joint International Conference on Requirements Engineering (RE'02)*, pages 61–68, Essen, Germany, 2002. IEEE.

[2] Ian F. Alexander and Neil Maiden. *Scenarios, Stories, Use Cases*. Wiley, 2004.

[3] Erling S. Andersen. *Systemutvikling*. NKI-forlaget, 2 edition, 1994.

[4] Kurt Bittner and Ian Spence. *Use Case Modeling*. Addison-Wesley, 2003.

[5] CCIMB. *Common Criteria for Information Technology Security Evaluation*. CCIMB, 1999.

[6] Rolly Constable, Marla Cowell, Sarita Zornek Crawford, David Golden, Jake Hartvigsen, Kathryn Morgan, Anne Mudgett, Kris Parrish, Laura Thomas, Erika Yolanda Thompson, Rosie Turner, and Mike Palmquist. Ethnography, observational research, and narrative inquiry. writing@csu. 2005. Retrieved 2005-04-12 from http://writing.colostate.edu/references/research/observe/.

[7] Robert Crook, Darrel Ince, Luncheng Lin, and Bashar Nuseibeh. Security requirements engineering: When anti-requirements hit the fan. In *IEEE Joint International Conference on Requirements Engineering (RE'02)*, pages 203–205. IEEE, September 2002.

[8] Folker den Braber, Theo Dimitrakos, Bjørn Axel Gran, Ketil Stølen, and Jan Øyvind Aagedal. Model-based risk management using UML and UP. In *Information Resources Management Association International Conference (IRMA'2002)*, pages 925–927, 2002.

[9] Brian Dobing and Jeffrey Parsons. Understanding the role of use cases in UML: A review and research agenda. *Journal of Database Management*, 11(4):28–36, 2000.

[10] Donald Firesmith. Engineering security requirements. *Journal of Object Technology*, 2(1):53–68, January-February 2003.

[11] Donald Firesmith. Security use cases. *Journal of Object Technology*, 2(3):53–64, May-June 2003.

[12] International Organization for Standardization. *ISO/IEC 17799 - Information technology - Code of practice for information security management*. ISO, 2000.

[13] Siv-Hilde Houmb, Folker den Braber, Mass Soldal Lund, and Ketil Stølen. Towards a UML profile for model-based risk assessment. In *Proc. UML 2002 Satellite Workshop on Critical Systems Development with UML*, pages 79–91. Munich University of Technology, 2002.

[14] IEEE. *IEEE Std. 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology*. IEEE, 1990.

[15] Ivar Jacobson, Magnus Christerson, Patrik Jonsson, and Gunnar Övergaard. *Object-Oriented Software Engineering, A Use Case Driven Approach*. Addison-Wesley, 1992.

[16] Heinz K. Klein and Michael Myers. A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS Quarterly*, 23:67–94, March 1999.

[17] L. Liu, E. Yu, and J. Mylopoulos. Security and privacy requirement analysis within a social setting. In *International Conference on Requirements Engineering (RE'03)*, pages 151–161, Monterey, California, September 2003. IEEE.

[18] Kine M. Munkelien. Verktøy for gjenbruk av misbrukstilfeller og sikkerhetstilfeller. Master's thesis, Norwegian University of Science and Technology, 2004.

[19] Roger S. Pressman. *Software engineering, A Practioner's Approach*. McGraw-Hill, 5 edition, 2001.

[20] Dimitris Raptis, Theo Dimitrakos, Bjørn Axel Gran, and Ketil Stølen. The CORAS approach for model-based risk analysis applied to the e-commerce domain. In *Communication and Multimedia Security (CMS-2002)*, pages 169–181, 2002.

[21] Colin Robson. *Real World Research*. Blackwell Publishing, 2 edition, 2002.

[22] Guttorm Sindre, Donald G. Firesmith, and Andreas L. Opdahl. A reuse-based approach to determining security requirements. In *9th International Workshop on Requirements Engineering: Foundation for Software Quality*, Portland, OR, USA, May 2003. ICSE'03.

[23] Guttorm Sindre and Andreas L. Opdahl. Eliciting security requirements by misuse cases. In *Proc. TOOLS Pacific 2000*, pages 120–131, November 2000.

[24] Guttorm Sindre and Andreas L. Opdahl. Determination and specification of security requirements during early RE. *Requirements Engineering Journal*, 2005.

[25] Yannis Stamatiou, Eva Skipenes, Eva Henriksen, Nikos Stathiakis, Adamantios Sikianakis, Eliana Charalambous, Nikos Antonakis, Ketil Stølen, Folker den Braber, Mass Soldal Lund, Katerina Papadaki, and George Valvis. The CORAS approach for model-based risk management applied to a telemedicine service. In *Medical Informatics Europe (MIE'2003)*, pages 206–211. IOS Press, 2003.

[26] Yannis C. Stamatiou, Eva Henriksen, Mass Soldal Lund, Eva Mantzouranis, Michalis Psarros, Eva Skipenes, Nikos Stathiakos, and Ketil Stølen. Experiences from using model-based risk assessment to evaluate the security of a telemedicine application. In *Telemedicine in Care Delivery (TICD'2002)*, pages 115–119, 2002.

[27] Nikos Stathiakis, Catherine Chronaki, Eva Skipenes, Eva Henriksen, Elina Charalambous, Adamantios Sykianakis, Georgios Vrouchos, Nikos Antonakis, Manolis Tsiknakis, and Stelios Orphanoudakis. Risk assessment of a cardiology ehealth service in hygeianet. In *Computers in Cardiology (CIC'2003)*, pages 201–204, 2003.

[28] David Wheeler. More than a gigabuck: Estimating GNU/Linux's size. June 2001. Retrieved 2005-02-03 from http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html.

A Misuse case template

Header

Misuse case ID	
Case Name	
Summary	
Author	
Date	

Description

Basic path	
Alternative paths	
Mitigation points	
Extension points	
Triggers	
Preconditions	
Assumptions	
Mitigation guarantee	
Related business rules	
Potential misuser profile	
Stakeholders and threats	
Terminology and explanations	
Scope	

Diagram

¡¡ insert diagram here ¿¿

B Misuse case description

Remark: This description is taken from [24] with some local modifications.

Misuse case ID	Unique alpha-numeric identification, composed of sub-system abbreviation and a sequence number (i.e 'AUTH-001')
Case Name	A short descriptive name for the misuse case (i.e 'Brute Force Password Attack')
Summary	A short summary of the misuse case
Author	Who wrote this case
Date	Date when this misuse case is written, in ISO8601 format (i.e '2004-01-30')
Basic path	This field describes the actions that the misuser(s) and the system go through to harm the system
Alternative paths	This field describes ways to harm the proposed system that are not accounted for by the basic path, but are still su ciently similar to be described as variants of the basic path
Mitigation points	This field identifies those actions in a basic or alternative path where misuse can be mitigated. Several ways to mitigate misuse of a particular action can be described in the same eld and each of them may be further described in a separate security use case. As for extension points, the misuse case must eventually have a mitigate relationship to a corresponding security use case. However, the detailed description of security use cases is optional, because it is often closer to design, requiring detailed analysis of risks and implementation costs that go beyond use and misuse cases
Extension points	In some cases, a misuse case may be extended with optional paths whose details are described in a separate extension misuse case. This eld lists the actions in the main or alternative paths where optional paths may be inserted. As for extension points in regular use cases, the misuse case must have an extend relationship to the misuse case that contains the optional path

Triggers	This field describes the states or events in the system or its environment that may initiate the misuse case. For some misuse cases, the trigger is just the predicate True, indicating a permanently present danger
Preconditions	This field describes the system states that make the misuse case possible
Assumptions	This field describes the states in the system s environment that make the misuse case possible
Mitigation guarantee	This field describes the guaranteed outcome of mitigating a misuse case. If the mitigation points are not yet speci ed in detail, the mitigation guarantee describes the level of security required from the mitigating security use cases that will be designed later. When the mitigation points in the misuse case have been detailed by security use cases, this eld describes the strongest
Related business rules	Typically, business rules will be violated by the misuse. This field contains links to such rules, maybe along with links to rules that enable the threat or that limit how it could be mitigated or eliminated
Potential misuser profile	This field describes whatever can be assumed about the misuser, for example, whether the misuser acts intentionally or inadvertently; whether the misuser is an insider or outsider; and how technically skilled the misuser must be
Stakeholders and threats	This field specifies the major risks for each stakeholder involved in the misuse case. On an abstract level, risks can be described textually, e.g., if the system is unavailable for several hours or a competitor gets hold of sensitive medical data about an applicant . On a concrete level, the likelihood and cost of each misuse variant can be estimated, where the cost includes potential losses, should the threats come true
Technology and data variations	A misuser may carry out a misuse case from a variety of technical platforms, such as a PC or a WAP phone and, since only a few equipment-related actions will differ in each case, it is unnecessary to specify two different paths. Instead, this field lists the candidate types of equipment and explains how they differ in particular actions.
Terminology and explanations	This field contains explanations of technical terms and other issues
Scope	This field indicated whether the proposed system in a misuse case is, e.g., an entire business, a system of both users and computers, or just a software system.