2006

Abuse Cases Revised: An Action Research Experience

Juhani Heikka
Oulu University, Finland, juhani.heikka@tol.oulu.fi

Mikko Siponen
Oulu University, Finland, mikko.siponen@tol.oulu.fi

Abuse Cases Revised: An Action Research Experience

Juhani Heikka
Department of Information Processing Science, University of Oulu, P.O.Box 3000, 90014 Oulu University, Finland
juhani.heikka@tol.oulu.fi

Mikko Siponen
Department of Information Processing Science, University of Oulu, P.O.Box 3000, 90014 Oulu University, Finland
mikko.siponen@tol.oulu.fi

Abstract

In the era of the information society, it is important to secure systems against security breaches. To understand and model such security breaches, the concept of abuse cases has been introduced. While the extant research on abuse cases offer important insights into secure systems design, these approaches do not offer (a) empirical evidence on their usefulness and relevance in practice, (b) means to prioritize security requirements, (c) explicit support for designing countermeasures, and (d) explicit support for integrating a risk management process into abuse cases. In this paper, we refine an extended abuse case model developed in our earlier research to address these four issues. This approach is then tested in practice with action research. The action research experience demonstrates that the refined approach was useful and easy to embed into information systems development methods in practice.

Keywords: IS security, secure systems design, abuse cases, misuse cases

1. Introduction

Modern society is increasingly based on information processing and global networks. IS security is a vital and a growing concern in such an IT-based society. Critical information regarding, for instance, health care, banking and telecommunication must be secured. To ensure that information systems (IS) are secured, several methods for the development of secure information systems have been provided. While information security standards and checklists, such as SSE-CMM or BS7799, are the most commonly used IS security methods (Mattia and Dhillon 2003; Siponen 2003), they do not provide any guidance on how to model security requirements. Therefore, to model IS security requirements and constraints, approaches like the Logical Modeling method (Baskerville 1989), Responsibility Modeling (Backhouse and Dhillon 1996), Meta-Notation (Siponen and Baskerville 2001), and UMLsec (Jürjens 2002; Popp et al. 2003) have been proposed (see review articles by Baskerville 1993; Dhillon and Backhouse 2001; Siponen 2005; Villarroel et al. 2005).

To capture and model systems' unauthorized use, McDermott and Fox (1999) introduced the concept of abuse cases. While use cases illustrate systems use, abuse cases are used to understand unauthorized use. Later, similar ideas, termed "misuse cases" (Sindre and Opdahl 2000), were introduced. While these two abuse case approaches (McDermott and Fox 1999;

Sindre and Opdahl 2000) provided seminal insights into the designing of secure systems and software, these approaches do not offer (a) empirical evidence on their usefulness and relevance in practice, (b) means to prioritize security requirements, (c) explicit support for designing countermeasures and (d) explicit support for integrating a risk management process into abuse cases. To address these concerns, we refine an extended abuse case model (Siponen et al. 2005) that addresses these issues, and then test it in practice with action research.

This paper is organized as follows. The second section reviews the existing abuse case approaches, and demonstrates how these four weaknesses pertain in these methods. The extended model is introduced in the third section. The fourth section introduces the research methods and the fifth section presents the action research results. In the sixth, section we discuss the validity of the research and the last section presents the conclusions.

2. Existing Research on Abuse and Misuse Cases

The existing abuse case approaches are: *abuse cases* by McDermott and Fox (1999), and *misuse cases* by Sindre and Opdahl (2000). While use cases model a system's authorized use, abuse cases model a system's unauthorized use. The idea is to understand potential threats to IS, which can range from a hacker's organized denial of service attacks to employees attempting to access an unauthorized file. Like use cases, abuse case descriptions can be graphical or textual. While abuse cases by McDermott and Fox (1999) are graphical, concentrating on abuse case notations and diagrams, Sindre and Opdahl (2000) consider it more important to produce textual versions of abuse cases. The other difference between these two is that McDermott and Fox (1999) recommend the modeling of the abuse case scenarios separately from use case diagrams, while Sindre and Opdahl (2000) suggest modeling the abuse in the same diagram as normal uses, to indicate the link between user, abuser, and system resources.

Next we describe how these abuse case approaches (McDermott and Fox 1999; Sindre and Opdahl 2000) entail the four weaknesses: lack of empirical evidence on their usefulness and relevance in practice, lack of explicit support for designing countermeasures, lack of support for integrating risk management process into abuse cases and lack of means to prioritize security requirements.

2.1 Lack of Empirical Evidence

An important feature of any IS security method is empirical evidence, demonstrating the method's usefulness in organizations. Both of the existing abuse case approaches (McDermott and Fox 1999; Sindre and Opdahl 2000) lack empirical evidence evaluating the approach in an industrial setting. According to Sindre and Opdahl *"the serious weakness of this work is that it has not been evaluated in practical software development projects"* (Sindre and Opdahl 2000, p. 130). Alexander (2002) provides initial experiences of the use of abuse cases in trade-off analysis, but the study lacks evidence of using abuse cases as part of the software design process, which is important to overcome developmental duality (Baskerville 1992). An abuse case approach should not only be designed such that it can be integrated into requirement analysis of IS and software development, but the integration of abuse cases with IS and the software development process should include the linking of abuse cases to all relevant documents in the development process.

The existing abuse case studies (Alexander 2002; Alexander 2003; Hope et al. 2004; McDermott and Fox 1999; McDermott 2001; Pauli and Xu 2005; Sindre and Opdahl 2000; Sindre and Opdahl 2001) provide only weak evidence or no evidence of integration of the abuse cases with IS and the software development process. Both abuse case approaches (McDermott and Fox, 1999; Sindre and Opdahl 2000) discuss the exploitation of abuse cases in some phases of the software development process, but do not provide information on how their model can be integrated systematically into different phases of IS development. According to McDermott and Fox (1999, p. 63): *"abuse case models can be helpful during the requirements, design, and testing phases of a security engineering process."* Sindre and Opdahl (2001) present some ideas as to how abuse cases can be used in different development phases, but, as they stated, it is important *"to investigate how misuse case analysis can be integrated with existing RE (requirement engineering) methods and techniques."* As can be seen from these extracts, it is not clear how these two abuse case approaches should be integrated with IS or the software development process to overcome developmental duality.

2.2 Support for Designing Countermeasures

The sequence of abuse cases and abuse case countermeasures are other important features for abuse case scenarios. While the abuse case template by Sindre and Opdahl (2000) introduces useful text fields, such as preconditions, assumptions, abuser profiles, and stakeholder threats, their template does not contain any fields for the sequence of abuse cases (how the abuse might be carried out) or countermeasures (how the abuse can be prevented). It is important that the countermeasures are included in the abuse case template, so that the whole development team, including IS developers and security experts, can participate in the analysis of security requirements and potential countermeasures.

According to White and Dhillon (2005) in many IS development projects, the security design teams and IS development teams are two separate groups, with differing development methods and objectives in systems development. In extreme cases little communication exists between these two teams; the IS being built by the systems developers and security solutions being added later by security experts. This often leads to an IS that is not properly secured and unable to be effectively used in the environment. (Baskerville 1992; White and Dhillon 2005.)

The abuse case approaches by McDermott and Fox (1999) and Sindre and Opdahl (2000) do not highlight the countermeasures; this is important in communication between different stakeholders (IS developers, security experts, clients, etc.). In specifying effective countermeasures, every team member's contribution should be sought. In this way, developer groups can form a common design plan, and the client can see what assets are in danger if the countermeasure is not effective or is not implemented. Also, by presenting the potential countermeasures in abuse cases, the developers will be able to notice if the same countermeasures can be used against many security threats. In the case of tracking abuse, the potential countermeasure could be the use of log files which could also be used to show non-repudiation, for example in an eCommerce application. In that case it would be economical to implement the specific security feature. If the countermeasure is broken down into smaller steps, a sequence diagram describing countermeasure mechanisms can be drawn.

2.3 Prioritization and Risk Management

The ability to prioritize security requirements is important in the industry. It is very common that software is developed in several software development cycles or releases, where the most important features are handled first and the other requirements in later releases. Thus, it is very important that the requirements, including security requirements, can be prioritized.

The existing abuse case approaches (McDermott and Fox 1999; Sindre and Opdahl 2000) do not address the prioritization aspect. In addition, when the development is carried out in several development releases (Baskerville et al. 2003), the developers should be informed in which version or release the abuse case is prevented (i.e., the countermeasure is implemented), so that the correct functioning of the countermeasure can be verified. It is also important to manage the development risks during requirement trade-offs and prioritization. The existing abuse case approaches do not entail explicit risk management processes.

To summarize, an abuse case approach should cover all security relevant aspects of development to improve communication between stakeholders in order to overcome the developmental duality, offer means to prioritize security requirements, highlight countermeasures, and integrate a risk management process into abuse cases.

3. Extended Abuse Cases

To overcome these shortcomings presented in the company, we crafted the abuse case model in our earlier research (cf., Siponen et al. 2005). We used this as a point of departure in the action research settings and integrated the fine-tuned model into company's IS development process. Table 1 provides an example of an extended abuse case approach.

Table 1. An example of an extended abuse case template. The fields marked with * are derived from the risk analysis process of Saltmarsh and Browne (1983).

Abuse case	What is the goal of this abuse case? For example, a hacker is hacking into the system.
Priority *	For example: high / medium / low
Frequency *	How often does this threat occur? Daily? Monthly?
Total cost of recovering *	What is the cost of recovering from the threat when it occurs? This information can be used in the prioritization of security requirements and abuse cases.
Abuse subject	Who is the abuser? What is his motivation? For example: hacker (unauthorized), malicious employee (authorized)
Preconditions	What are the preconditions for the abuse to take place? What is the state of the system at the beginning of the abuse? For example: "a hacker tries to break into the locked system by guessing a user's password."
Outcome	What is the outcome of the abuse? What state is the system in after this abuse case?
Description and countermeasure *	Describe how the system reacts to the abuse and how the abuse target is secured. The sensitive information which needs to be secured can be recognized from this description. For example: in login, the system allows three login attempts and if these fail, the account is locked. 1. A hacker feeds in the passwords and user ID 2. The system checks the password and the used ID (security requirement SR-01) 3. The hacker… 4. The system…

Extensions	Used to describe alternative scenarios and exceptions. 2a. …the rule for how the system discovers the extension - Describe how the extension is handled 2b. …the rule for how the system discovers the extension - Describe how the extension is handled
Data	What data is manipulated in this abuse case – for example, what classes? If there are messages between systems list the messages or references to messages here. Testing necessitates that the data manipulated (for example the password and user ID) in the abuse case is described here. In the case of unauthorized access to address data the data could be: first name, last name, street address, zip code, city, etc. This information helps the testing phase.
Requirements	References to requirements in the requirement catalog that are relevant to this abuse case.
Test cases	References to test cases relevant to this abuse case or the countermeasures described.
Version	In which version or release is the countermeasure completed? After that the functioning of the countermeasure can be tested.
Other issues	Other issues related to this abuse case.

In order that security aspects can be integrated into software and IS development methods, it is important that the security features are taken care of in each phase of software development (Baskerville 1989; Baskerville 1992). This means that the security requirements and features should be specified systematically, for example, by using this approach. Furthermore, the specified security requirements have to be linked logically through the software development process.

In Table 1, the relevant functional and security requirements are linked to the abuse case to emphasize the connection of security and functional requirements. The documents that are utilized after specifying abuse cases (e.g., class descriptions and diagrams, sequence diagrams, security plan, architectural designs, technical manual) are also linked to abuse cases to ensure that security requirements are handled carefully in the development process. To cover the whole development process, we also linked the abuse cases to the relevant test cases. In the modeling of abuse cases and security requirements, we modified Sindre and Opdahl's (2000) original notation (Fig. 1).

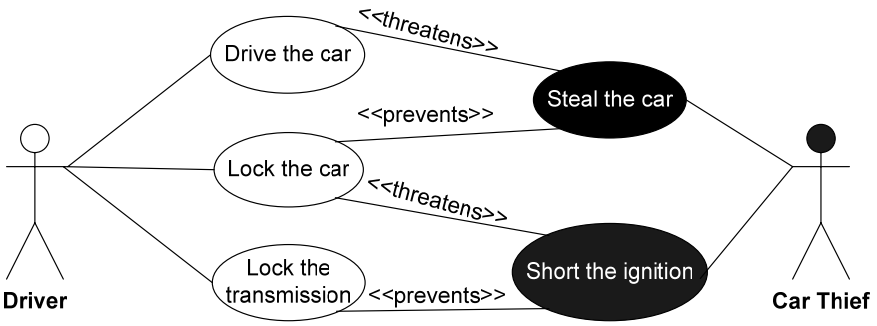

Figure 1. The abuse case diagram. Abuse is expressed by using black (Alexander 2003).

To highlight the countermeasures, we extended the abuse case description with the abuse sequence (Description and countermeasure in Table 1). The purpose of the sequence is to clarify the countermeasures by describing the steps in which attention should be paid to the security issues and requirements. While it might not be possible to divide every abuse case into a simple sequence of steps, this should be done when possible to help support the next phases of software development, for example, the construction of a sequence diagram or test case of the countermeasure.

To solve the risk management concern, we decided to integrate the five-phased process modified from Saltmarsh and Browne (1983) into the abuse cases (demonstrated with * in Table 1). In this way we can use abuse cases 1) to identify risks and 2) to calculate what damage the abuse might cause. Furthermore, we can 3) evaluate how often the abuse will probably happen and 4) assess how the risk can be minimized. Then, using the above-mentioned information we can 5) evaluate the relevance of countermeasures. This information can be also used in the prioritization of security requirements (e.g., it is not reasonable to spend €10000 to protect assets worth €1000).

After presentation of the research methodology, we test and refine our extended version of the abuse case through action research in the fifth section.

4. Research Methodology

Action research has been advocated as ideal for studying IS development methods in their practical settings (Avison et al. 2001; Baskerville and Wood-Harper 1996; Baskerville and Wood-Harper 1998; Baskerville 1999). By putting theories to work in practice, not only is the scientific knowledge expanded, but concrete problems are also solved in the participating organizations (Baskerville and Wood-Harper 1998). It is therefore no wonder that action research studies examining the relevance of IS security methods in practice have been recently called for by IS security scholars (Baskerville 1994; Baskerville and Myers 2004; Dhillon and Backhouse 2001). Action research is a form of field intervention consisting of four phases: diagnosis, action planning, action taking and evaluation.

Baskerville and Wood-Harper (1998) have proposed seven validity criteria for information systems action research: (1) the research should be set in multivariate social situations; (2) the observations should be recorded and analyzed in an interpretive frame; (3) researcher actions intervene in the research setting; (4) the method of data collection includes participatory observation; (5) changes in the social setting are studied; (6) the immediate problem in the social setting must have been resolved during the research; and (7) the research should illuminate a theoretical framework that explains how the actions led to a favorable outcome. These criteria are applied to evaluate the results of the action research intervention.

5. Action Research Intervention

The action research intervention was carried out in a large software company operating in Finland, during the year 2005. The company uses in-house IS development methods and Unified

Modeling Language (UML) in their IS development. The goal of the research was to provide a new approach to the developers for the handling of security requirements in IS development.

In the ***diagnosis phase****,* we interviewed six employees from the organization (head designer, IT chief, software architect, software engineer, method specialists, project manager) to gain knowledge about the current situation regarding security issues in IS development. Prior to the research project, the company used checklists and separate security instructions in the IS development process. During the interviews the practitioners pointed out two problems in the existing development process. First, the organization lacked a systematic approach for handling security requirements. According to the software architect, *"in the current approach we are not sufficiently methodical… if the client does not emphasize the importance of security it might go by the board."* Another developer stated: *"The problem seems to be that the development teams by themselves do not see what kinds of risks the use of the system causes. Rarely, if ever, do the developers consider how critical the systems are, what the actual risks are, and what kind of information security it would be important to invest in."* Second, while separate instructions exist, the developers did not use these since they were not integrated in the actual software development process. The chief developer described this situation: *"The development method does not have integrated security instructions for design or implementation of information systems, but the instructions are separate and the use of them is not supervised."*

In the ***action planning*** phase, we held discussions with the developers and we proposed our extended abuse cases to overcome the identified problems. The developers agreed that the extended abuse cases would solve the problems. The developers wanted to standardize the extended abuse cases with the use cases used in the company. According to one developer, *"the metaphors should be same in the normal use cases as in the abuse cases."* We decided to combine the extended abuse template with the company's use case template to form a template to fulfill the company's needs.

In the ***action taking*** phase, we integrated the abuse cases into the software development method in three action research cycles. In the first cycle, we integrated the abuse cases into the requirement analysis phase. At the end of the first integration cycle, the developers reviewed and provided comments on the construction. The company exploited user scenarios in the early phases of requirement analysis, and the technology chief suggested that *"it would be good also to have negative / abuse scenarios that are easy to understand that can be used in communication with clients' top management."* The abuse scenarios were added in the second cycle. The abuse scenarios can be used as a basis for abuse cases to identify security threats (abusers and their actions) at a high level. Later these scenarios can be used in the prioritization of security requirements. This way the client can state their opinion about the identified threats, focus financial resources on the most critical development targets and make a reasoned trade-off if necessary. In the third cycle, we concentrated on the linking of abuse cases to all phases throughout the whole system development process. After the cycles, the developers reviewed the security-enriched method. They requested a process (Fig. 2) which maps the action carried out in each phase.

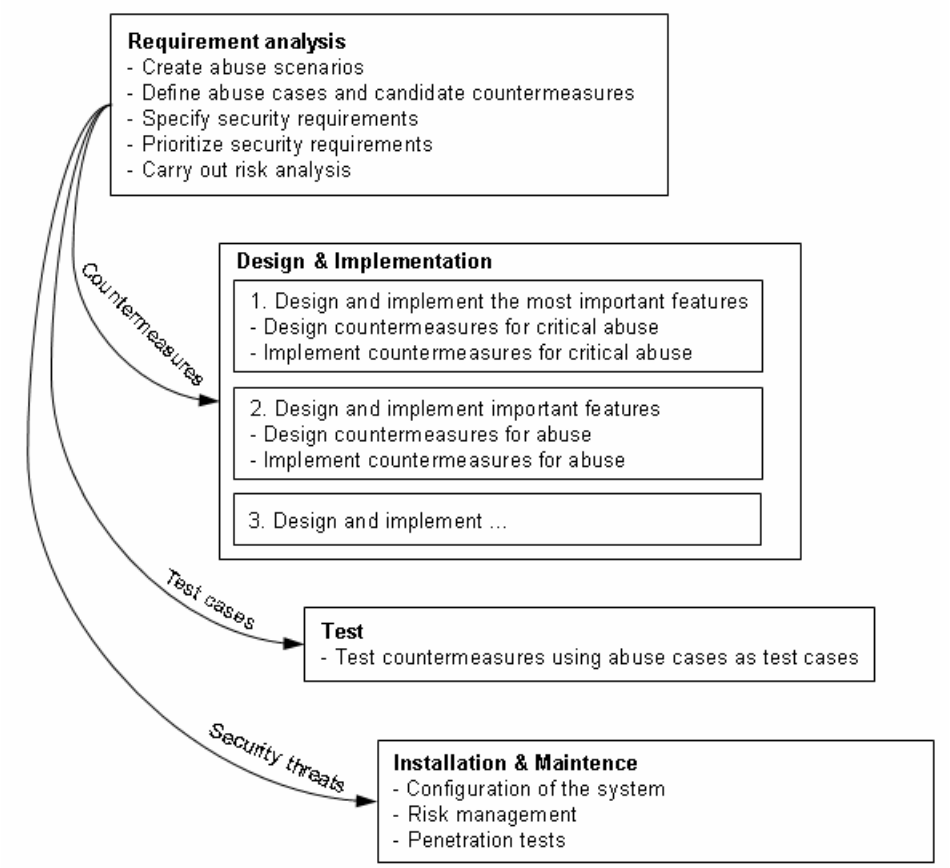

Figure 2. Process for exploitation of abuse cases in IS development.

In the ***evaluation phase***, a group of developers tested the abuse scenarios and cases. The developers started the testing by creating abuse scenarios which were prioritized using weighted mean values. In this phase, some of the abuse scenarios were discarded because the development team found out that these threats were irrelevant. Next the developers created an abuse case diagram, but they felt that the original versions (McDermott and Fox 1999; Sindre and Opdahl 2000) did not emphasize the countermeasures. The developers, together with the researchers, solved the problem by adding an extra color to the diagram to highlight the countermeasures (yellow). In addition, we decided to divide the diagram (Fig. 3) into three different zones: use cases (left), abuse cases (right), and countermeasures (middle).

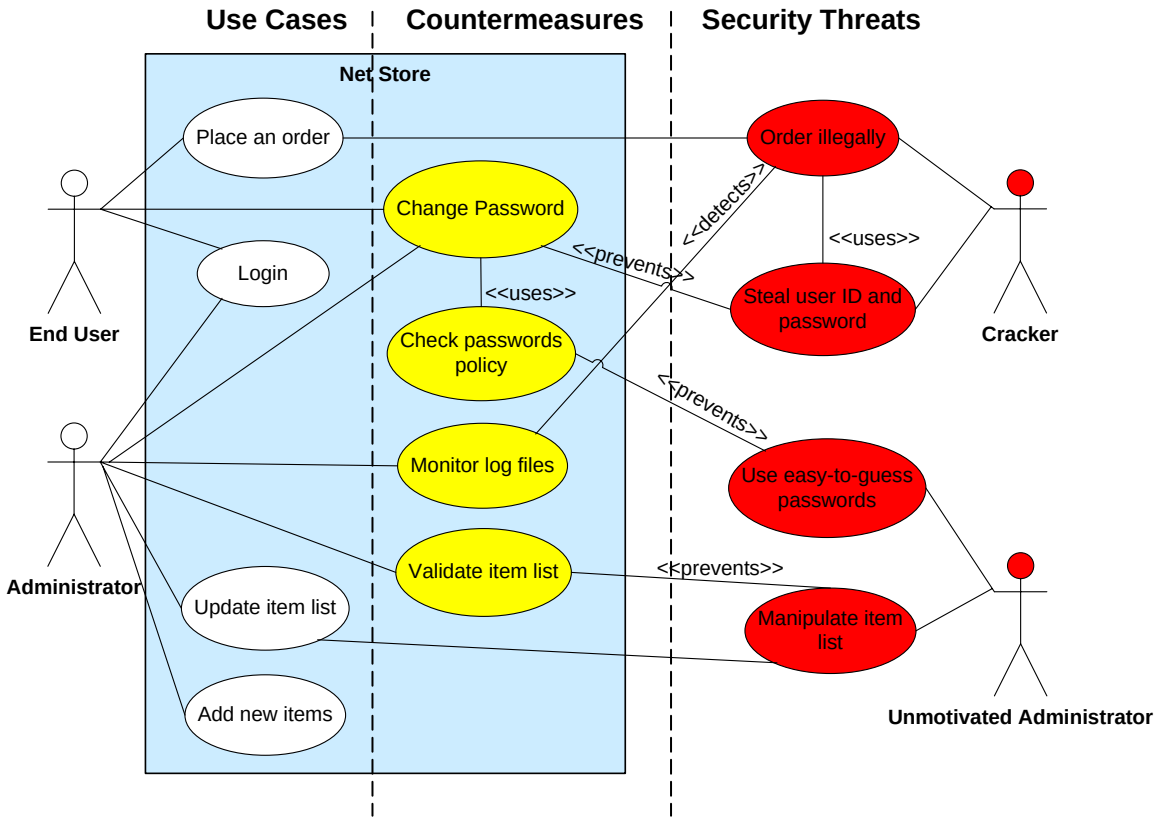

Figure 3. An example of an extended abuse case diagram. The countermeasures are highlighted in yellow and the abuse cases in red.

By using these extensions, developers could form a summary of the requirement analysis phase in which the whole development team could see the functionality, security threats related to these functions, and countermeasures, at a glance. In addition, the developers saw the need to crosscheck the different requirements to ensure that all critical parts of the IS are covered with the necessary security controls (Table 2). In this way, the development team, including IS developers and security experts, can communicate with each other and achieve a shared design which reduces the gap between different objectives.

Table 2. The matrix of use and abuse cases and their "access" to IS resources.

	Use case 1	*Use case 2*	*Use case 3*	...
Abuse case 1	*Resource x*			
Abuse case 2	*User ID and password*	*Order information*		
Abuse case 3			*Resource y*	
...				

In the testing of the new approach, the developers found the abuse scenarios and the extended abuse cases very useful in practice. For example, the software developer described this situation as follows: *"Abuse scenarios can be used to picture the operational environment such that we*

can prioritize the security requirements later in the development process. Thinking about security threats gives an overview of security mechanisms which helps later phases." Extended abuse cases were easy to learn and easy to use. For instance, the software developer stated: *"There is no noticeable difference in using abuse cases [compared to normal use cases]."* The developers also found that the approach helps in the analysis, design and testing of security requirements. As a developer described it, *"abuse cases help requirement analysis and link the defined requirements to the design. In the design phase we can model technical aspects of the countermeasures. […] The approach is also useful in testing phase – it can be used to cover areas that normal test cases do not cover."*

6. Discussion: Validity of Research

The research in this part of the thesis conforms to the seven validity criteria for IS action research (Baskerville and Wood-Harper 1998). *First*, the research was set in a multivariate social situation involving intricate relationships and communication between different stakeholders in the development project. The company was lacking a methodological approach for the handling of their clients' security requirements. *Second*, the observations and interviews were recorded and analyzed in an interpretive frame. This data and the interpretation have been provided in the text. *Third*, the researchers actively intervened, working directly with the developers to introduce the new approach (extended abuse cases) into their practices. *Fourth*, the method of data collection included participatory observation as well as interviews. The observations were made both during the integration of abuse cases as well as during the exploitation of extended abuse cases in empirical testing. *Fifth*, the developers assessed reviews of the usability and success of the extended abuse case approach. *Sixth*, the immediate problem in the research was resolved (integration of extended abuse cases) during the research, according to the evaluation of the collaborators. *Finally*, the actions in the setting were tightly linked to the extended abuse case framework. This framework defined the actions and explains how the actions led to the positive outcome.

7. Conclusions

In the era of the information society, it is important to secure information systems against security breaches. To understand and model such security breaches, the concept of abuse cases has been introduced. While the extant abuse case approaches offer important insights into secure systems design, these approaches do not offer (a) empirical evidence on their usefulness and relevance in practice, (b) means to prioritize security requirements, (c) explicit support for designing countermeasures, and (d) explicit support for integrating a risk management process into abuse cases. To address these concerns, we refined an extended abuse case model (Siponen et al. 2005) and the abuse case notation (Sindre and Opdahl 2000). This approach is then tested and refined in practice through an action research. The action research experience demonstrates that the refined approach was useful and easy to embed into information systems development methods in practice.

References

Alexander, I. "Initial Industrial Experience of Misuse Cases in Trade-Off Analysis," *Proceedings of IEEE Joint International Requirements Engineering Conference*, Essen, 2002, pp. 61–68.

Alexander, I. "Misuse Cases: Use Cases with Hostile Intent," *IEEE Software* (20:1), 2003, pp. 58–66.

Avison, D., Baskerville, R. and Myers, M. "Controlling Action Research Projects," *Information Technology & People* (14:1), MCB University Press, 2001, pp. 28–45.

Backhouse, J., and Dhillon, G. "Structures of Responsibilities and Security of Information Systems," *European Journal of Information Systems 5*, 1996, pp. 2–10.

Baskerville, R. "Logical Controls Specification: An Approach to Information System Security," In *Systems Development for Human Progress*, Klein H and Kumar K (eds.), North-Holland, Amsterdam, 1989, pp. 241–255.

Baskerville, R. "The Developmental Duality of Information System Security," *Journal of Management Systems 4*, 1992, pp. 1–12.

Baskerville, R. "Information Systems Security Design Methods: Implications for Information Systems Development," *ACM Computing Surveys* (25:4), 1993, pp. 375–414.

Baskerville, R. "Research Directions in Information Systems Security," *International Journal of Information Management 14*, 1994, pp. 85–87.

Baskerville, R. "Investigating Information Systems with Action Research," *Communications of the Association for Information Systems* (2:3), Article19, November 1999, pp. 1–32.

Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B. and Slaughter, S. "Is Internet-speed Software Development Different?" *IEEE Software* (20:6), 2003, pp. 102–107.

Baskerville, R. and Myers, M.D. "Special Issue on Action Research in Information Systems: Making IS Research Relevant to Practice – Foreword," *MIS Quarterly* (28:3), September 2004, pp. 329–335.

Baskerville, R. and Wood-Harper, T. "A Critical Perspective on Action Research as a Method for Information Systems Research," *Journal of Information Technology 11*, 1996, pp. 235–246.

Baskerville, R. and Wood-Harper, T. "Diversity in Information Systems Action Research Methods," *European Journal of Information Systems 7*, 1998, pp. 90–107.

Dhillon, G. and Backhouse, J. "Current Directions in IS Security Research: Toward Socio-Organizational Perspectives," *Information Systems Journal* (11:2), 2001, pp. 129–156.

Hope, P., McGraw, G. and Anton, A.I. "Misuse and Abuse Cases: Getting Past the Positive," *IEEE Security and Privacy* (2:3), 2004, pp. 90–92.

Jürjens, J. "UMLsec: Extending UML for Secure Systems Development," *Lecture Notes in Computer Science*, Volume 2460, Springer-Verlag, 2002, pp. 412– 425.

Mattia A. and Dhillon, G. "Applying Double Loop Learning to Interpret Implications for Information Systems Security Design," *IEEE Systems, Man & Cybernetics Conference*, Washington DC, 2003, pp. 2521–2526.

McDermott, J. and Fox, C. "Using Abuse Case Models for Security Requirements Analysis," *Proceedings of the 15th Annual Computer Security Applications Conference* (ACSAC '99), 1999, pp. 55–64.

McDermott, J. "Abuse-Case-Based Assurance Arguments. Computer Security Application Conference," *Proceedings of the 17th Annual Computer Security Applications Conference* (ACSAC'01), 2001, pp. 366–374.

Pauli, J.J. and Xu, D. "Misuse Case-Based Design and Analysis of Secure Software Architecture," *Proceedings of the International Conference on Information Technology: Coding and Computing* (ITCC´05), Vol. II, April 2005, pp. 398–403.

Popp, G., Jürjens, J., Wimmel, G. and Breu, R. "Security-Critical System Development with Extended Use Cases," *Proceedings of the Tenth Asia-Pacific Software Engineering Conference* (APSEC'03). December 2003, pp. 478–487.

Saltmarsh, T.J. and Browne, P.S. "Data Processing – Risk Assesment," In *Advantages in Computer Security Management*, Vol. 2, John Wiley and Sons Ltd, 1983, pp. 93–116.

Sindre, G. and Opdahl, A.L. "Eliciting Security Requirements by Misuse Cases," *Proceedings of the 37th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS-Pacific)*, 2000, pp. 120–131.

Sindre, G. and Opdahl, A.L. "Capturing Security Requirements through Misuse Cases," *Proceedings of Norsk informatikkonferanse (NIK'2001)*, Trondheim, Norway, 2001, pp. 219–230.

Siponen, M.T. "Analysis of Modern IS Security Development Approaches: Towards the Next Generation of Social and Adaptable ISS Methods", *Information and Organization (15:4),* October 2005, pp. 339–375.

Siponen, M.T. "Information Security Management: Problems and Solutions," *Proceedings of the 7th Pacific Asia Conference on Information Systems*, Adelaide, South Australia, July 2003, pp. 1550-1561.

Siponen M.T. and Baskerville, R. "A New Paradigm for Adding Security into IS Development Methods," *Proceedings of the IFIP TC11 WG11.1/WG11.2 Eigth Annual Working Conference on Advances in Information Security Management & Small Systems Security*, Las Vegas, Nevada, USA, 2001, pp. 99–112.

Siponen, M., Baskerville, R. and Kuivalainen, T. "Integrating Security into Agile Development Methods," In R. H. Sprague (Ed.), *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, Los Alamitos, California: IEEE Computer Society, 2005, pp. 1–7.

Villarroel, R., Fernandez-Medina, E. and Piattini, M. "Secure Information Systems Development – a Survey and Comparison," *Computers and Security* (24:4), 2005, pp. 308–321.

White, E.F.R. and Dhillon, G. "Synthesizing Information System Design Ideals to Overcome Developmental Duality in Securing Information Systems," *Proceedings of the 38th Annual Hawaii International Conference on System Sciences* (HICSS'05) - Track 7, 2005, pp. 186–194.