Lecture Notes
in Business Information Processing

Series Editors

Wil van der Aalst
  Eindhoven Technical University, The Netherlands
John Mylopoulos
  University of Trento, Italy
Michael Rosemann
  Queensland University of Technology, Brisbane, Qld, Australia
Michael J. Shaw
  University of Illinois, Urbana-Champaign, IL, USA
Clemens Szyperski
  Microsoft Research, Redmond, WA, USA
Preface

These are the proceedings of the Third IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling, held in Delft (The Netherlands) on November 9 and 10, 2010. It followed the success of PoEM 2008 and 2009 (both held in Stockholm), which each attracted over 50 participants from all over the world, representing both industry and academia. This indicates that enterprise modeling (EM) has gained popularity both in the academic community and among practitioners. The interactive format of the previous conferences sparked constructive interaction between research and practice. PoEM 2010 further strengthened this interaction.

The PoEM conferences contribute to establishing a dedicated forum where the use of EM in practice is addressed by bringing together researchers, users and practitioners. The main focus of PoEM is EM methods, approaches, and tools, and how they are used in practice. The goal of the conference was to further a better understanding of the practice of EM and improve the theory behind the practice, contributing to improved EM practice and to the sharing of knowledge.

For this third edition, the founders of PoEM, Anne Persson and Janis Stirna, passed the torch for the first time; we hope we lived up to the high standards set by them and thank them for their initiative, commitment, and excellent work. PoEM will return to Scandinavia next year, and will remain to do so every other year.

The 17 high-quality papers (out of 44 submissions) presented at PoEM 2010 display a welcome diversity in topics while being duly centered around the enterprise modeling theme. A number of submissions reflected the trend for both practitioners and academics to look into domains and conceptualizations that are more and more distant from those that are the focus in traditional information systems engineering, addressing a continuation of the move towards dedicated and far reaching “business-orientation.” Also, we observe that the field is slowly but surely maturing, as indicated by an increase in detail and specialization of the contributions.

In its 2010 edition, PoEM saw relatively few submissions concerning enterprise architecture. It is very likely this is because this year PoEM was co-located with two other events: the Practice-Driven Research in Enterprise Transformations (PRET 2010) working conference, and the Trends in Enterprise Architecture Research (TEAR 2010) workshop. Proceedings of both events also appear in the Springer LNBIP series. The three events together constituted Enterprise Engineering Week, with Erik Proper at the helm.
We would like to extend warm thanks to everyone involved in making PoEM 2010 happen: the members of the PC, the people who submitted papers, keynote speakers Etienne Rouwette (Radboud University Business School) and Jeroen van Grondelle (Be Informed), and everyone involved in the organization.

September 2010

Patrick van Bommel
Stijn Hoppenbrouwers
Sietse Overbeek
Erik Proper
Joseph Barjis
Organization

POEM 2010 was organized by the Radboud University Nijmegen and the Delft University of Technology.

Executive Committee

Conference Chair Stijn Hoppenbrouwers, Radboud University Nijmegen
PC Chair Patrick van Bommel, Radboud University Nijmegen
PC Co-chair Sietse Overbeek, Delft University of Technology
PC Co-chair Erik Proper, Radboud University Nijmegen

Program Committee

Marko Bajec University of Ljubljana, Slovenia
Rimantas Butleris Kaunas University of Technology, Lithuania
Wolfgang Deiters Fraunhofer ISST, Germany
Eric Dubois CRP Henri Tudor, Luxembourg
Mathias Ekstedt KTH, Sweden
Gustaf Ericsson Alfa Laval AB, Sweden
Xavier Franch Universitat de Catalunya, Spain
Remigijus Gustas Karlstad University, Sweden
Terry Halpin LogicBlox, Australia
Jarl Hoglund Allmentor AB, Sweden
Stijn Hoppenbrouwers Radboud University Nijmegen, The Netherlands
Pontus Johnsson KTH, Sweden
Havard Jorgensen Commitment AS, Norway
John Krogstie Norwegian University of Science and Technology, Norway
Michel Léonard University of Geneva, Switzerland
Peri Loucopoulos Loughborough University, UK
Raimundas Matulevicius University of Tartu, Estonia
Graham McLeod PROMIS Solutions AG, South Africa
Daniel Moody University of Twente, The Netherlands
Christer Nellborn Nellborn Management Consulting AB, Sweden
Bjorn Nilsson Anates AB, Luxembourg
Andreas Opdahl University of Bergen, Norway
Sietse Overbeek Delft University of Technology, The Netherlands
Oscar Pastor Valencia University of Technology, Spain
Anne Persson University of Skövde, Sweden
<table>
<thead>
<tr>
<th>Name</th>
<th>Institution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naveen Prakash</td>
<td>GCET, India</td>
</tr>
<tr>
<td>Erik Proper</td>
<td>Radboud University Nijmegen, The Netherlands</td>
</tr>
<tr>
<td>Jolita Ralytė</td>
<td>University of Geneva, Switzerland</td>
</tr>
<tr>
<td>Peter Rittgen</td>
<td>Vlerick Leuven Gent Management School, Belgium</td>
</tr>
<tr>
<td>Colette Rolland</td>
<td>University of Paris 1 Pantheon Sorbonne, France</td>
</tr>
<tr>
<td>Kurt Sandkuhl</td>
<td>Jonkoping Technical University, Sweden</td>
</tr>
<tr>
<td>Ulf Seigerroth</td>
<td>Jonkoping International Business School, Sweden</td>
</tr>
<tr>
<td>Keng Siau</td>
<td>University of Nebraska-Lincoln, USA</td>
</tr>
<tr>
<td>Pnina Soffer</td>
<td>University of Haifa, Israel</td>
</tr>
<tr>
<td>Maarten Steen</td>
<td>Novay, The Netherlands</td>
</tr>
<tr>
<td>Janis Stirna</td>
<td>University of Stockholm, Sweden</td>
</tr>
<tr>
<td>Renate Strazdina</td>
<td>Ernst and Young SIA, Latvia</td>
</tr>
<tr>
<td>Patrick van Bommel</td>
<td>Radboud University Nijmegen, The Netherlands</td>
</tr>
<tr>
<td>Olegas Vasilecas</td>
<td>Vilnius Gediminas Technical University, Lithuania</td>
</tr>
<tr>
<td>Eric Yu</td>
<td>University of Toronto, Canada</td>
</tr>
</tbody>
</table>
Table of Contents

Comparing Two Techniques for Intrusion Visualization ............... 1
   Vikash Katta, Peter Karpati, Andreas L. Opdahl,
   Christian Raspotnig, and Guttorm Sindre

Needs-Driven Bundling of Hosted ICT Services ....................... 16
   Jaap Gordijn, Floris de Haan, Sybren de Kinderen, and
   Hans Akkermans

Enterprise Modeling for Business Intelligence ....................... 31
   Daniele Barone, Eric Yu, Jihyun Won, Lei Jiang, and
   John Mylopoulos

Business Modeling Experience for a State Pension Voluntary
   Insurance Case ................................................. 46
   Alcedo Coenen, Bas van Gils, Charlotte Bowy, Roel Kerkhofs, and
   Sander Akkermans

Composition of Semantic Process Fragments to Domain-Related
   Process Families ............................................... 61
   Claudia Reuter

Assessing Collaborative Modeling Quality Based on Modeling
   Artifacts .......................................................... 76
   D. (Denis) Ssebuggwawo, S.J.B.A. (Stijn) Hoppenbrouwers, and
   Erik Proper

Patient Care across Health Care Institutions: An Enterprise Modelling
   Approach .......................................................... 91
   Sobah Abbas Petersen, Grete Bach, and Astrid Brevik Svarlein

The Practice of Competence Modelling ................................. 106
   Thomas Albertsen, Kurt Sandkuhl, Ulf Seigerroth, and
   Vladimir Tarasov

Modeling Network-Based Defence: Success and Failure of an Enterprise
   Modeling Endeavour ............................................... 121
   Thomas Albertsen, Kurt Sandkuhl, Ulf Seigerroth, and
   Vladimir Tarasov

Interactive Goal Model Analysis Applied – Systematic Procedures
   versus Ad hoc Analysis ........................................... 130
   Jennifer Horkoff, Eric Yu, and Arup Ghose
Adapting UML Activity Diagrams for Mobile Work Process Modelling: Experimental Comparison of Two Notation Alternatives .......................... 145
   Sundar Gopalakrishnan, John Krogstie, and Guttorm Sindre

A Repository Architecture for Business Process Characterizing Models .................................................................................. 162
   Shang Gao and John Krogstie

A Rule-Based Approach for the Recognition of Similarities and Differences in the Integration of Structural Karlstad Enterprise Modeling Schemata .......................................................... 177
   Peter Bellström

Focused Conceptualisation: Framing Questioning and Answering in Model-Oriented Dialogue Games ................................................. 190
   S.J.B.A. (Stijn) Hoppenbrouwers and I. (Ilona) Wilmont

   Constantinos Giannoulis, Michael Petit, and Jelena Zdravkovic

Integration of Interactive, Behavioral and Structural Aspects of Conceptual Models ................................................................. 217
   Remigijus Gustas

Towards Defining a Competence Profile for the Enterprise Modeling Practitioner ................................................................. 232
   Anne Persson and Janis Stirna

Author Index .............................................................................................................. 247
Comparing Two Techniques for Intrusion Visualization

Vikash Katta\textsuperscript{1,3}, Peter Karpati\textsuperscript{1}, Andreas L. Opdahl\textsuperscript{2}, Christian Raspotnig\textsuperscript{2,3}, and Guttorm Sindre\textsuperscript{1}

\textsuperscript{1}Norwegian University of Science and Technology, NO-7491 Trondheim, Norway
\{kpeter, guttors\}@idi.ntnu.no
\textsuperscript{2}University of Bergen, NO-5020 Bergen, Norway
andreas.opdahl@uib.no
\textsuperscript{3}Institute for Energy Technology, NO-1751 Halden, Norway
\{vikashk, christir\}@hrp.no

Abstract. Various techniques have been proposed to model attacks on systems. In order to understand such attacks and thereby propose efficient mitigations, the sequence of steps in the attack should be analysed thoroughly. However, there is a lack of techniques to represent intrusion scenarios across a system architecture. This paper proposes a new technique called misuse sequence diagrams (MUSD). MUSD represents the sequence of attacker interactions with system components and how they were misused over time by exploiting their vulnerabilities. The paper investigates MUSD in a controlled experiment with 42 students, comparing it with a similar technique called misuse case maps (MUCM). The results suggest that the two mostly perform equally well and they are complementary regarding architectural issues and temporal sequences of actions though MUSD was perceived more favourably.

Keywords: requirements engineering, security, experiment, threat modeling.

1 Introduction

The increased web presence of modern enterprises due to e-commuting, e-commerce, and distributed, inter-organizational workflows have also created new opportunities for computer crime, thus accentuating the need to focus on enterprise security [1]. Security must be in focus on many levels, from high level strategy and managerial policies and down to the competence and awareness of each single employee and correct implementation and operation of each ICT application. Whether talking about the construction of new information systems or the daily operation of existing ones, a vital precondition for improving their security is the ability to learn from previous failures. One possibility is to look at textual descriptions of successful attacks, like [2], but there may be several advantages in combining this with more generic visual models of attacks, both related to understandability, knowledge reuse, and integration with model-based systems engineering. Many techniques have been proposed to capture threat and attack-oriented information, for example attack trees [3], misuse cases [4] and CORAS [1], and more recently misuse case maps (MUCM) [5], focusing on modelling complex intrusions. This technique highlights the relation between security
and system architecture, and thereby provides integrated overviews of user-oriented security threats, mitigations and architecture. Even though MUCM has its strengths to show intrusion across the system architecture, it might be confusing to follow the sequence of steps of the intrusion. Evaluations of MUCM [5,6] have suggested a need for better visualization of the sequence of attack steps. To the authors’ knowledge, there is a lack of techniques to visualize such sequences.

This paper proposes a new threat modelling technique called misuse sequence diagrams (MUSD), aiming to give an overview on the sequence of the attacker’s actions during an intrusion for different stakeholder groups. The technique is based on UML sequence diagrams [7] and misuse cases [2], and utilizes security concepts like vulnerability exploitation and mitigation. Since MUSD was meant to improve on some shortcomings of MUCM, we found it interesting to evaluate experimentally whether MUSD really provided improvement over MUCM related to these issues. Hence we performed a controlled experiment with 42 students to evaluate the participants’ performance with the two techniques, as well as their opinions about the techniques.

The two techniques are relevant to enterprise-modelling practice because service-orientation has made information systems increasingly distributed across internal and external organization boundaries. Hence, software architecture has moved from being a technological concern inside "black-box" information systems, to being a "white-box" concern on the boundary between organization and technology. The techniques we present are, to the best of our knowledge, the first attempt to conceptualize this boundary and support it with useful modelling notations. We hope MUCMs and MUSDs can contribute to understanding distributed service-oriented information systems and their architectures from an enterprise and organizational context.

The rest of the paper is structured as follows. Section 2 discusses background and related work. Then, section 3 presents misuse sequence diagrams. Sections 4 and 5 present the research method and the experiment results, respectively. Section 6 discusses the findings, and section 7 concludes the paper.

2 Background and Related Work

According to the definitions of RFC 2828 [8], a vulnerability is a weakness in a system's design, implementation, or operation/management that can be exploited to violate its security policy. A threat is a potential for violation of security, depending on circumstance, capability, and an action / event that could cause harm. A countermeasure/mitigation is something that reduces a threat or attack by eliminating or preventing it, minimizing the harm caused, or by reporting it to enable corrective action.

Misuse cases (MUC) [4,9] are used for threat modelling and security requirements elicitation. While use cases (UC) present the required behaviour of a system, MUC capture undesired behaviour, extending UC with new elements like misuser, misuse case and mitigation use case, as well as new relations like threaten and mitigate. MUC allow an early focus on security in the development process and facilitate discussion among a wide group of stakeholders.

Misuse case maps (MUCM) [5] is a recently proposed technique combining MUC and use case maps (UCM) [10] for an integrated view of security issues and system architecture. MUCM can be used to visualize the trace of an intrusion on the
Comparing Two Techniques for Intrusion Visualization

Fig. 1. MUCM notation and its interpretation

architecture of the system, visually based on the UCM notation extended by vulnerabilities, exploit paths and mitigations. An experiment performed to evaluate MUCM showed that it significantly improved the understanding of intrusions and identification of mitigations when compared to MUC combined with a system architecture outline [5].

Fig. 1 shows the MUCM notation. It extends UCM’s basic notation with intrusions, represented by exploit paths that cut through vulnerable parts of the system. Each path starts with a triangle and different symbols at the end indicate the outcome. The number on the path indicates its order in a bigger sequence of paths. The system may have vulnerable points or parts which are susceptible to threats, materialized in the attack if the exploit path crosses the vulnerability.

Fig. 2 shows a partial example about a technical entry into the computer system of a company for penetration test purposes [2]. The tester first made 3 unsuccessful attack attempts against the Apache server and firewall. The fourth attempt utilised an undocumented Solaris feature (portmapper - rpcbind - bound to port 32770) to get the dynamic port of the mount daemon (mountd) from the portmapper and direct an NFS request to it, thus succeeding to remotely mount and download the target file system.

As for other related methods, [11] proposes a framework for object-oriented security requirements analysis based on UC, MUC and security use cases for the elicitation and analysis of security requirements in embedded systems. In addition to the framework, misuse sequence diagrams have been introduced to better explain a single misuse case scenario. [12] proposes an aspect-oriented methodology for designing secure applications. The methodology uses sequence diagrams for three purposes: describing functionality (primary model), describing attacks on the functionality (misuse model), and describing the incorporation of security mechanisms (security-treated model). The MUSD technique proposed here is similar to those of [11] and [12]. However, they neither visualize complex multi-stage intrusion scenarios nor how vulnerabilities of system components are exploited and mitigated.
CORAS [1] is a method for security analysis using system descriptions based on UML diagrams as an input to traditional risk analysis techniques, such as HazOp [13] and Fault Tree Analysis (FTA) [14]. While CORAS offers a specialized set of diagrams for security risk analysis, only using UML diagrams as an input, the MUSD is solely based on MUC and SD. MUSD is not a method for security risk analysis, just a technique for visualizing complex intrusion scenarios.

It could be tempting to compare MUSD to STAIRS [15], which is a method with focal point on refinement of interactions in UML. STAIRS allows among others for the specification of behaviour that shall not be allowed in an implementation, but does not refer to the security or risk domain. However, MUSD presents a specialized notion for security and intrusion visualization, while STAIRS’ focus is on a general refinement of UML based specifications thus a comparison does not seem relevant.

UMLSec is an UML extension for secure systems development which can be used “to evaluate UML specifications for vulnerabilities using a formal semantics of a simplified fragment of UML” [16]. Its focus is on formal verification of specifications (e.g. for a protocol) against different adversary types. Although sequence diagrams are included in UMLSec, they rely on other diagram types specifying the broader context. Furthermore, UMLSec applies heavyweight methods which need specific trainings. MUSD aims to facilitate discussion of different stakeholder groups allowing competency transfer and trade-off considerations early in the system development.

3 MUSD

Misuse sequence diagrams (MUSD) combine misuse cases (MUC) and sequence diagrams (SD), to depict and analyze complex intrusion scenarios. MUSD show involved objects, their vulnerabilities and how these were misused. The notation extends UML sequence diagrams as shown in Fig. 3. Just like by the MUC notation, regular and misuse symbols can be combined in the same MUSD diagram. Attack-related...
symbols are shown in red color since inversion (a la misuse cases) does not work for arrows. Exploit messages are messages originating from the attacker with the intention of harming the system. Intrusions are represented by one or more exploit messages using vulnerabilities of the objects in the system. Action symbols are used to represent parts of the intrusion scenario which are unclear or not detailed enough. Exploit actions are performed by the attacker with the intention of attacking the system. Objects which are part of an action will have their lifelines covered by the rectangle denoting the action.

The sequence of exploit messages and actions shows the steps taken by the attacker. These steps are mostly causally related; each building on the result of the previous steps. Notes might appear for explanations. Fig. 4 presents a part of the MUSD depicting the same penetration test case as shown with a MUCM in Fig. 2.

4 Research Method

The purpose of the experiment was to evaluate whether MUSD would be better than MUCM for conveying attack sequences. On the other hand, a gain in this respect could result in other weaknesses instead, especially it would be natural to suspect that MUSD would be poorer than MUCM in conveying the relationship between attacks and system architecture. Hence, the experiment compared the understanding of case descriptions resulting from usage of two notations, both for attack sequence and architectural aspects. Understanding may be a goal in its own right in enterprise systems development, but more often it will be the basis for various problem solving activities. Hence the subjects performed one task measuring understanding by a set of True/False questions, and another addressing problem solving in terms of identifying threats and possible mitigations in the given cases. In the following the experiment design, variables and hypotheses are explained in more detail.
Experimental Design

To compare the two techniques, we conducted a controlled experiment with 42 subjects who used the two techniques individually on two different cases described in the literature, a bank intrusion [2] and a penetration test [2]. To control for the order in which the techniques were used and the cases were solved, a Latin-Squares experimental design was used as shown in Table 1. The within-experiment data regarding understanding, performance and perception thereby became paired, comprising two dependent samples. Table 1 shows the order in which the techniques were used and cases were solved by the groups. Group 1’s experiment sheet is available at [17].

| Case order: |
| Bank intrusion before penetration test | Penetration test before bank intrusion |
| MUCM before MUSD | Group 1 | Group 2 |
| MUSD before MUCM | Group 3 | Group 4 |
4.2 Variables

After controlling for the participants’ backgrounds, for each combination of technique and case, three types of tasks were solved: an understanding task, a performance task and a perception task. Table 2 summarizes the main variables used in the analysis.

Background was measured by a pre-task questionnaire addressing the participants’ self-assessed knowledge of SD, MUSD, UCM, MUCM and security in general on a 5-point scale. They were also asked to report their numbers of completed semesters of ICT studies and months of ICT-relevant work experience.

<table>
<thead>
<tr>
<th>Name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TECH=MUCM, TECH=MUSD</td>
<td>The technique used in that part of the experiment, either MUCM or MUSD.</td>
</tr>
<tr>
<td>CASE=BANK, CASE=PEN</td>
<td>The case solved in that part of the experiment, either BANK (the bank intrusion) or PEN (the penetration test).</td>
</tr>
<tr>
<td>KNOW_MOD, KNOW_SD, KNOW_MUSD, KNOW_UCM, KNOW_MUCM, KNOW_SEC</td>
<td>The participants' self-assessed knowledge about systems modelling (KNOW_MOD), sequence diagrams (KNOW_SD), misuse sequence diagrams (KNOW_MUSD), use case diagrams (KNOW_UCM), misuse case diagrams (KNOW_MUCM) and security analysis (KNOW_SEC) on a 5-point scale, where 1 is “Never heard about it” and 5 is “Expert”.</td>
</tr>
<tr>
<td>STUDY</td>
<td>The participants' self-reported semesters of ICT-studies.</td>
</tr>
<tr>
<td>JOB</td>
<td>The participants' self-reported months of ICT-relevant work experience.</td>
</tr>
<tr>
<td>UND_1, UND_2, ...</td>
<td>The 20 statements about the case, scored by the participants as either true or false. A correct assessment is scored 1 and a wrong one -1.</td>
</tr>
<tr>
<td>UND_MUCM, UND_MUSD, UND_NEUT</td>
<td>Sum of scores for the statements about of MUCM (architectural issues), MUSD (temporal sequence issues) and neutral aspects of the problem cases.</td>
</tr>
<tr>
<td>UND_TOT</td>
<td>Sum of scores for the all twenty statements about the problem cases.</td>
</tr>
<tr>
<td>VULN, MITIG</td>
<td>The numbers of unique vulnerabilities and mitigations identified by the participants.</td>
</tr>
<tr>
<td>VUMI</td>
<td>The sum of unique vulnerabilities and mitigations identified by the participants.</td>
</tr>
<tr>
<td>PER_1, PER_2, ...</td>
<td>Scores on the 5-point Likert scales for the individual statements about perception of the techniques.</td>
</tr>
<tr>
<td>PER_PU, PER_PEOU, PER_ITU</td>
<td>Average scores on the 5-point Likert scales for the four statements about perceived usefulness of, perceived ease of use of and intention to use of the techniques.</td>
</tr>
<tr>
<td>PER_AVE</td>
<td>Average scores on the 5-point Likert scales for all the twelve statements about the techniques.</td>
</tr>
</tbody>
</table>
Understanding was measured by 20 true/false questions about the case, scoring 1 point for a correct answer, -1 for incorrect, and 0 points for no answer. The statements were designed so that 6 of the statements addressed aspects where MUCM was assumed to be a better technique, 6 addressed aspects where MUSD was assumed to be better and 8 addressed aspects where neither technique was assumed to have an advantage. Specifically, we assumed MUCM would perform best for statements relating to architecture and that MUSD would perform better for statements about the sequence of activities. In each group, half the statements had a positive and half a negative formulation (so that equal numbers of true and false answers were expected).

Performance was measured by asking the participants to identify and list as many vulnerabilities and mitigations as they could in the planned system. Then the numbers of unique (and relevant) vulnerabilities and mitigations were counted. Three example vulnerabilities were given for both cases. Even though the type and criticality of the vulnerabilities are important, these issues were out of scope for the experiment. Such issues can be considered as a part of the future work.

Perception was measured by a post-task questionnaire adapted from the Technology Acceptance Model (TAM) [18,23]. 4 questions addressed perceived usefulness (PU), 4 addressed perceived ease of use (PEOU) and 4 investigated the participants’ intention to use (ITU) the technique in the future. One statement in each group was negative, with a lower score reflecting a positive opinion, inverted before analysis.

| H1 | Better score for architectural Q's with MUCM than MUSD. That is, more questions about architectural issues were assessed correctly with MUCMs than with MUSDs. | \[ \text{UND}_\text{MUCM}[\text{MUCM}] > \text{UND}_\text{MUCM}[\text{MUSD}] \] |
| H2 | Better score for temporal sequence Q's with MUSD than MUCM. That is, more questions about temporal sequence issues were assessed correctly with MUSDs than with MUCMs. | \[ \text{UND}_\text{MUSD}[\text{MUSD}] < \text{UND}_\text{MUSD}[\text{MUCM}] \] |
| H3 | Different numbers of statements in the NEUT group were assessed correctly with MUCMs and MUSDs. | \[ \text{UND}_\text{NEUT}[\text{MUCM}] \neq \text{UND}_\text{NEUT}[\text{MUSD}] \] |
| H4 | Different numbers of statements were assessed correctly with MUCMs and with MUSDs. | \[ \text{UND}_\text{TOT}[\text{MUCM}] \neq \text{UND}_\text{TOT}[\text{MUSD}] \] |
| H5 | Different numbers of vulnerabilities were identified with MUCMs and with MUSDs. | \[ \text{VULN}[\text{MUCM}] \neq \text{VULN}[\text{MUSD}] \] |
| H6 | Different numbers of mitigations were identified with MUCMs and with MUSDs. | \[ \text{MITI}[\text{MUCM}] \neq \text{MITI}[\text{MUSD}] \] |
| H7 | Different numbers of vulnerabilities and mitigations were identified with MUCMs and with MUSDs. | \[ \text{VUMI}[\text{MUCM}] \neq \text{VUMI}[\text{MUSD}] \] |
| H8 | The usefulness of MUCMs and MUSDs were perceived differently. | \[ \text{PER}_\text{PU}[\text{MUCM}] \neq \text{PER}_\text{PU}[\text{MUSD}] \] |
| H9 | The ease of use of MUCMs and MUSDs were perceived differently. | \[ \text{PER}_\text{PEOU}[\text{MUCM}] \neq \text{PER}_\text{PEOU}[\text{MUSD}] \] |
| H10 | The intentions to use MUCMs and MUSDs again were different. | \[ \text{PER}_\text{ITU}[\text{MUCM}] \neq \text{PER}_\text{ITU}[\text{MUSD}] \] |
| H11 | MUCMs and MUSDs were perceived differently. | \[ \text{PER}_\text{AVE}[\text{MUCM}] \neq \text{PER}_\text{AVE}[\text{MUSD}] \] |

Table 3. Hypotheses of the comparison experiment
4.3 Hypotheses

The hypotheses for our experiment are listed in Table 3. The corresponding set of null hypotheses, as well as an additional set of hypotheses about correlations between understanding (H1-H41), performance (H5-H71) and perception (H8-H111), are omitted here for space reasons.

4.4 Experimental Procedure

The 42 participants of the experiment were recruited from a class of second year computer science students, receiving financial support for an excursion as "payment" for participating. Each group consisted of 10 or 11 students solving the task under equal conditions (same room, same time limits). The experiment comprised 10 steps:

1. Filling in the pre-experiment questionnaire (2 min)
2. Reading a short introduction to the experiment (1 min)
3. Using the first assigned technique on the first assigned case:
   a. Reading the introduction to the first technique (1.5 pages, 9 min)
   b. Reading the textual description of case 1 while looking at the related diagrams (3-4 pages, 12 min)
   c. Answering 20 true/false questions about the case (8 min)
   d. Finding as many vulnerabilities and mitigations as possible (11 min)
   e. Filling in post-experiment questionnaire (4 min)
4. Easy physical exercises as a break (2 min)
5. Repeat steps 3a-e for the second technique and case (7+ 14 + 5 + 10 + 4 min)

The duration of the steps was decided dynamically. There was always enough time to finish the steps, except for steps 3c and 5c, which we stopped before everyone could finish because we wanted to see how efficient the participants were.

5 Results

5.1 Comparing Backgrounds

We used Kruskal-Wallis H tests of four independent groups for all background variables to control for differences between the participant groups. We found no significant differences, neither with respect to knowledge, study semesters nor job months.

We used 2-tailed Wilcoxon signed-rank tests to compare the participants self-assessed knowledge backgrounds in the following areas. They reported being significantly more knowledgeable about systems modelling than security analysis (KNOW_MOD = 2.79, KNOW_SEC = 1.81, p = .000), as well as about sequence diagrams versus use case maps (KNOW_SD = 3.20, KNOW_UCM = 2.74, p = .003). There was no significant difference in knowledge about misuse sequence diagrams and misuse case maps (KNOW_MUSD = 1.37, KNOW_MUCM 1.36).

The participants reported between 2 and 4 semesters of ICT-studies, with a mean of 3.05 and a small standard deviation. They reported 2.07 months of ICT-relevant work experience. Here the standard deviation was higher due to three outliers with 6, 19 and 54 months. All the others reported 2 months or less.