The Journal of Systems & Software 183 (2022) 111090

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Adopting threat modelling in agile software development projects[☆,☆☆]

Karin Bernsmed[a,*], Daniela Soares Cruzes[a], Martin Gilje Jaatun[a], Monica Iovan[b]

[a] *SINTEF Digital, Trondheim, Norway*
[b] *Visma, Romania*

ARTICLE INFO

Article history:
Received 24 November 2020
Received in revised form 14 August 2021
Accepted 7 September 2021
Available online 22 September 2021

Keywords:
Agile
Software
Threat modelling
Data Flow Diagrams
STRIDE
MS-TMT

ABSTRACT

The goal of secure software engineering is to create software that keeps performing as intended, even when exposed to attacks. Threat modelling is considered to be a key activity to reach this goal, but has turned out to be challenging to implement in agile teams. This paper presents results from four different studies, in which we have investigated how agile teams do threat modelling today. Study A is based on observations and document analysis from five teams in a single organisation, Study B is based on interviews with eight individuals from four different organisations, Study C is based on a questionnaire survey of 45 students at two different universities, and Study D is based on interviews with seven teams in a single organisation, supplemented with document analysis. Our results include findings, challenges and current good practice related to the use of Data Flow Diagrams, STRIDE and the Microsoft Threat Modelling Tool. We also cross-check our findings with previous relevant work, and provide recommendations for making the threat modelling activities more useful to agile teams.

1. Introduction

Threat modelling has been identified as one of the most important activities when developing secure software (Jeffries, 2012; McGraw, 2006; Shostack, 2014). The main idea behind threat modelling is to *think like an attacker* during the design phase. A well-defined threat model helps to identify threats to different assets of a system by utilising well-grounded assumptions on the capabilities of any attacker interested in attacking such a system, thereby enabling software development teams to identify critical areas of design that need to be protected.

Threat modelling is a wide concept that encompasses a broad range of techniques that can be utilised to make a system more secure. Our approach, which is based on the Microsoft approach to threat modelling (Swiderski and Snyder, 2004), consists of three main elements (Jaatun et al., 2019):

- Asset identification, which includes identifying information and/or services that are essential or critical for the system;
- Creating a Data Flow Diagram (DFD), which is an overview over how assets are stored, processed or otherwise interact with the system and that includes system interfaces and potential attack surfaces;
- Identifying threats that may affect one or more of the assets. Threats can be identified through the use of existing frameworks, such as STRIDE (Shostack, 2014), which is an acronym for the six threat categories Spoofing identity, Tampering with data, Repudiation threats, Information disclosure, Denial of service and Elevation of privileges, and further analysed and visualised in, for example, attack trees (Schneier, 1999).

These activities are often performed as part of a risk assessment, in which the threats with unacceptably high risk are mitigated through the identification, implementation and deployment of suitable countermeasures in the system.

Over time, various threat modelling approaches and methodologies have been developed, and are nowadays a natural part of the process of designing, for example, secure web applications (OWASP, 2020a). In agile software development, however, the practice of threat modelling has not yet been widely adopted, and the practitioners have few sources of recommendations on how to include such practices in their processes. Many other security practices are also challenging to adopt in agile software development. The agile principles intrinsically generate some of the challenges for security work in agile software development, as reported in the systematic literature review by Oueslati et al. (2015). They identify these challenges and evaluate their causes with respect to the agile values, the agile principles, and the security assurance practices. Oueslati et al. (2015) found that the challenges are related to:

☆ This work was supported by the Science of Security in Agile Software Development (SoS-Agile) project, funded by the Research Council of Norway (grant number 247678).
☆☆ Editor: Kelly Blincoe.
* Corresponding author.
E-mail addresses: karin.bernsmed@sintef.no (K. Bernsmed), daniela.s.cruzes@sintef.no (D.S. Cruzes), martin.g.jaatun@sintef.no (M.G. Jaatun), monica.iovan@visma.com (M. Iovan).

https://doi.org/10.1016/j.jss.2021.111090
0164-1212/© 2021 Elsevier Inc. All rights reserved.

- Software development lifecycle, meaning that the security related activities need to be applied for each development iteration and that the iteration time is limited and may not fit time consuming security activities;
- Incremental development, meaning that changes of requirements and design break system security requirements, and that continuous code changes makes completing the assurance activities difficult;
- Security assurance, meaning that security assessment favours detailed documentation, and that tests are, in general, insufficient to ensure the implementation of security requirements;
- Awareness and collaboration, meaning that security requirements are often neglected, and that the developer role must be separate from security reviewer role to have objective results;
- Security management, meaning that organisations compromise on security activities to accommodate accelerated release schedule.

Studies in software security usually focus on software security activities in general, and there are few empirical studies focusing on specific security practices in agile software development. Researchers have not yet gathered evidence on how to best include threat modelling in a software development process when you do not have an up-front design to rely on. Our research aims to build evidence on the adoption of threat modelling in agile projects, by gathering information on the challenges and experiences in different projects that we have been facilitating during the last few years. The main research problem that we address is *"How can we adapt the threat modelling approach to better suit agile software development projects?"*. The following research questions were thus defined for our study:

- RQ1: How are agile teams doing threat modelling today?
- RQ2: What are the challenges/best practices of doing Data Flow Diagrams?
- RQ3: What are the challenges/best practices of applying STRIDE?
- RQ4: Is the Microsoft Threat Modelling Tool helpful?
- RQ5: How can we make the results from threat modelling more useful to agile teams?

In this paper we present the results from the triangulation of the results from four different studies, in which we have collected and analysed empirical information by means of observations, interviews and surveys (for details on data sources, collection methods and sample sizes, see Table 1). Two of the studies (Cruzes et al., 2018; Bernsmed and Jaatun, 2019) were previously published by the authors of this paper and reused in the analysis of this paper. We have then cross-checked our results with previous relevant work, in order to further strengthen (or reject) our findings.

The paper is organised as follows. In Section 2, we present background and related work. In Section 3, we explain the methodologies that we have used to gather data and triangulate the results from the different studies. The four studies are presented in Section 4. Section 5 presents the results, which are then discussed in Section 6. Finally, Section 7 provides our conclusions and points to future work.

2. Background and related work

Threat modelling was introduced as a concept by Microsoft around the turn of the century, providing another arrow in the quiver of the Trustworthy Computing initiative. It was formally documented in the book by Swiderski and Snyder (Swiderski and Snyder, 2004), and included as a component in the initial release of the Microsoft Security Development Lifecycle (SDL) (Howard and Lipner, 2006). Threat modelling is also part of what McGraw calls Architectural Risk Analysis (McGraw, 2006).

Existing threat modelling approaches vary from conceptual frameworks to practical tools. Myagmar, Lee and Yurcik describe threat modelling as a process aiming to understand the complexity of a system and identifying all possible threats to the system (Myagmar et al., 2005). According to Shostack (Shostack, 2014), threat modelling usually employs two types of models; one that represents the system that is to be built and another one that represents the actual threats to the system.

Regarding tools, Microsoft has made their Threat Modelling Tool (MS-TMT) (Microsoft, 2017) freely available, but there have been some doubts whether there are plans for further versions after the current one (v7). There are other threat modelling tools that have yet to see a wide adoption in the market, in the following we will mention a couple of examples. A recent alternative is the OWASP Threat Dragon tool (OWASP, 2020b), which has support for Windows, Linux and MacOS in addition to a web app. Threat Dragon currently leaves something to be desired in user-friendliness of the drawing function, but supports Confidentiality, Integrity and availability (CIA) analysis and privacy threat modelling (referred to as LINDDUN) in addition to the STRIDE methodology (Bygdås et al., 2021). Another tool is SPARTA, presented by Sion et al. (2018), which extends STRIDE threat modelling using Data Flow Diagrams (DFDs) with more explicit countermeasures linked to each identified threat. SPARTA also performs simulations to estimate the vulnerability of a solution, taking the capabilities of different types of attackers into account.

Regarding previous studies, a systematic mapping study on security in agile requirements engineering from 2018 (Villamizar et al., 2018) provides some insights; one of the most common ways to handle security requirements in an agile context is to propose modifications to the methods that are already being used, or to introduce new artefacts to the existing set of artefacts. It is also common to introduce guidelines to handle security issues. The mapping study also identifies three major limitations of the agile security requirements approaches: lack of time (for addressing security requirements in each release); lack of security skills and awareness; and lack of guidelines for collecting and addressing security requirements.

Behutiye et al. (2020), performed a systematic mapping study of the management of quality requirements in agile and rapid software development. They found that security was one of the most frequently referenced quality areas, and concluded that there is a need for more (and presumably better) tools and lightweight management strategies. Terpstra, Dabeva and Wang (Terpstra et al., 2017) studied discussions related to security requirements engineering on a LinkedIn discussion group, and identified coping strategies related to artefacts, human factors, and the agile process itself. The latter is particularly interesting for our study, hinting at the need for explicitly introducing new activities (e.g., threat modelling) in the software development process.

Heijden, Broasca and Serebrenik (Heijden et al., 2018) provide results from an interview study where they identified security challenges in large-scale agile software development projects.

In 2019, Weir et al. performed an action research study, in which they performed a series of lightweight interventions (workshops facilitated by a security expert), intending to improve the teams' motivation to consider security and awareness (Weir et al., 2019). The results show that the participating teams measurably improved their abilities to deliver secure software.

Dodson, Souppaya and Scarfone (Dodson et al., 2020) present high-level, methodology-agnostic guidelines for evolving a Software Development LifeCycle (SDLC) into a Secure Software Development Lifecycle (SSDL), or as they say, a secure software development framework (SSDF). The framework consists of practices split into four groups, and in the group "Produce Well-Secured Software" (PW), we find the practice "Design Software to Meet Security Requirements and Mitigate Security Risks" and the task PW.1.1: "Use forms of risk modelling, such as threat modelling, attack modelling, or attack surface mapping, to help assess the security risk for the software". The NIST document provides numerous references for each task; but interestingly whereas BSIMM is extensively cited, this task only refers to secure software engineering activities from the Attack modelling (AM) practice, not the Architectural Risk Analysis (AA) practice, which subsumes threat modelling.

3. Methodology

Threat modelling is a practice, and the purpose of our research is to better understand how we can adapt it to better fit agile software development projects. This is a real-life problem and to find an answer, we needed to collect data from the intended users of this practice. Qualitative research is useful for exploring and understanding how individuals, or groups of individuals, describe the aspects of a problem that they are facing. Quantitative research, on the other hand, is often used to test theories, by examining the relationship among variables. However, both approaches have their limitations, and the weaknesses of one approach can be compensated for by the strengths of the others (Creswell, 2014). To address the research questions in this paper, we therefore found it useful to apply a mix of both.

This paper consists of a collection of results from four different studies on threat modelling, in which the problem has been addressed from different angles. Our approach can be characterised as *mixed methods research*, which by Easterbrook et al. (2008) has been described as an approach where one employs data collection and analysis techniques associated with both quantitative and qualitative data. In this paper, we have applied a concurrent triangulation strategy, which means that we have used different methods concurrently, in an attempt to confirm, cross-validate or corroborate our findings from the four different studies. Note that two of the studies that have been included in this paper (referred to as Study A and Study B in this paper) have already been published in other venues (Cruzes et al., 2018; Bernsmed and Jaatun, 2019).

Table 1 provides an overview over the data sources used in this paper. The "Data collection" column indicates what method(s) have been applied to collect the data, "Time period" indicates when the studies were executed and "Data sample" indicates where the participating subjects came from. "Authors involved" indicates who among the authors of this paper were involved in the studies; Karin Bernsmed (KB), Martin Gilje Jaatun (MGJ), Daniela Soarez Cruzes (DSC) and/or Monica Iovan (MI), and "Reference" indicates whether (and where) the results have been previously published. We also indicate what supplementary information has been included in the paper (see Appendix).

Table 2 provides an overview over the findings from each of the studies, and how they contribute to our research questions. The table also includes an overview over what methods have been used to analyse the data. In addition to the four studies, we have also used the relevant previous work presented in Section 2 to further support (or reject) some of the findings that we have derived.

The studies presented in this paper are part of a bigger project,[1] which investigates how to meaningfully integrate software security into agile software development activities. The subjects participating in these studies are hence associated with companies and universities with which we have a long-term collaboration.

4. Empirical studies

In the following we present the four studies (A–D) that form the empirical basis for this article. Studies A and B are based on observations in a single organisation, and interviews in four different organisations, respectively. Study C is based on a questionnaire survey of university students, and Study D is based on interviews and document analysis in a single organisation.

4.1. Study A: Threat modelling in agile teams (I)

Study A is a case study which was performed in an agile software organisation during 2017–2018. The intention of the study was to investigate how development teams apply threat modelling in their projects. The main focus of the study was on the challenges that the teams are facing, but we also wanted to find out how can we adapt the threat modelling approach to better fit with agile development projects.

The results from this study have already been published in full in the paper by Cruzes et al. (2018). This section therefore only reproduces the main findings from the study.

4.1.1. Industrial context

The company that participated in this study is an SME with less than 100 employees, who are stationed in three different geographical locations (Norway, Poland, and Finland). The company has five product teams, who design and implement applications for fleet management, real time transport information systems and travel planning and ticketing systems. The company products are complex in the sense that they involve hardware and software integrated solutions, as well as mobile and web applications.

4.1.2. Data collection and analysis

Observation was our primary source of information in this study. More specifically, we observed eight threat modelling sessions, each lasting around 1–2 h. The data was mainly collected by means of observation templates (see Figure 4 in Appendix), which was filled out by one of the authors of this paper after each session. In addition to the observation templates, our data sources also include the artefacts generated during the sessions (lists of assets, DFDs, list of threats and lists of relevant risks).

In addition to the data collected from the threat modelling sessions, we also arranged periodic meetings with the security champions in the company every other week during six months. We also had a number of informal discussions with the participants who attended the threat modelling sessions. In this study, we coded the data using the MaxQDA qualitative data analysis tool[2] in an exploratory way, so we could see which themes would emerge from the observations. Once themes were identified, we grouped them by phases (preparation, execution and post-execution) and ended up with a total number of 21 general main challenges encountered. We then classified these issues according to the challenges and, finally, the researchers interpreted and discussed the findings together in order to reach consensus. We then generalised the findings into theoretical statements and discussed them in accordance with the literature.

[1] https://www.sintef.no/sos-agile/.

[2] https://www.maxqda.com/.

Table 1
An overview over the data sources used in this paper.

Data source	Data collection	Time period	Data sample	Authors involved	Supplementary material	Publications
Study A	Observations and document analysis	2017–2018	1 organisation, 5 teams	DSC, MGJ, KB	Observation template included in Appendix.	Results published in (Cruzes et al., 2018).
Study B	Interviews	2018	4 organisations, 8 individuals	KB, MGJ	Interview guide included in Appendix.	Results published in (Bernsmed and Jaatun, 2019).
Study C	Survey	2019	2 universities, 45 individuals	KB, MGJ	Data Flow Diagram included in Appendix.	Unpublished work.
Study D	Interviews and document analysis	2020	1 organisation, 7 teams	DSC, MI	Interview guide included in Appendix.	Unpublished work.

Table 2
An overview over the findings from each of the studies.

Data source	Main topics investigated	Data analysis	Main types of findings	Contribution to RQ(s)
Study A	TM, agile	Coding with MaxQDA (VERBI Software GmbH, 2021) and extracting findings.	Challenges, best-practices	RQ1, RQ2, RQ3
Study B	TM, agile	Inductive analysis (Thomas, 2006) of interview transcripts.	Challenges, best-practices	RQ1, RQ2, RQ3
Study C	MS-TMT, STRIDE	Numerical analysis of survey in Microsoft Excel.	Usability & usefulness	RQ4
Study D	Agile, DFD, MS-TMT, STRIDE	Coding and analysis with MaxQDA (VERBI Software GmbH, 2021).	Challenges, best-practice, usefulness	RQ1, RQ2, RQ3, RQ4, RQ5

4.1.3. Main findings from Study A

Here we provide a summary of the main findings of this study, which are related to our research questions. For more details on these findings, we refer to the paper where they were originally published (Cruzes et al., 2018).

RQ1: How are agile teams doing threat modelling today? In this organisation, threat modelling was performed in dedicated sessions, attended by the security champions, the development teams and in some cases also the product owner and a representative from the systems operations department. The baseline for the activities were pre-generated lists with prioritised assets and Data Flow Diagrams, which we had asked them to create before the meetings. The list of assets and the DFDs were then discussed and updated during the sessions. During the sessions, they also used STRIDE to generate lists of threats (one list per asset). The teams were asked to document the results after the sessions had ended, but most of them did not do this. When asked why, they said they did not see how this kind of documentation would be useful.

RQ2: What are the challenges/best practices of doing data flow diagrams? It was challenging to motivate the teams to draw the Data Flow Diagrams and we also observed that it often took a long time. In most cases, it was the security champion who had prepared the diagram before the session started, with help from different people in his/her team. Some of the security champions felt it was overwhelming and that it was hard to produce good drawings that correctly reflected the design of the systems. Some teams spent a long time (more than 10 h) on creating the DFD and they mentioned it felt like they were “taking time from development activities" and that it had a negative impact on their productivity.

We also observed that it was hard to set the right level of abstraction for the DFD, and that this had a negative impact on the discussions. When the DFD was too high-level, the discussion got unfocused and vague. In some cases the team chose to draw a deeper view of the DFD during the meeting, hence taking time from the other planned activities (like threat identification). In other cases the DFD was too complex, leading to the need to arrange more meetings to be able to cover all aspect of the DFD when identifying the threats.

Further, it was challenging to map the interfaces in the DFD with software and systems delivered by other teams. Some teams had problems with drawing the interfaces with other internal systems in the company because the product development teams often comprised many different products. They were also not sure how to deal with cross-cutting concerns. In practice, it was difficult to gather all relevant teams in one meeting, or to know who should be invited from the other teams.

We also observed difficulties with linking the DFD to the actual code. The teams mentioned that they actually did not know how the system was really implemented, and the DFDs that they produced therefore were tainted by the uncertainty whether the system was actually implemented that way, especially where there was a lot of legacy code involved.

The last challenge we observed was that it was difficult to maintain the DFDs. Because of the lack of focus on documentation in agile teams, they did not show motivation to maintain or have a strategy to update the DFD. The teams did not feel like this was an activity that they would want to do frequently. The main impression was that they would do it now, and then maybe in one year or something. It was not easy for the teams to foresee how often they would need to perform a new threat modelling activity for their systems. We as researchers also had problems to state clearly how to decide.

Regarding best practices, we observed that including a security expert for facilitating the meetings helped the discussions to be more focused and also more relevant to the participants. Also, as we mentioned before, this company has adopted the role of Security Champions in each team, and these were very important players in the whole process; they were the ones driving the drawing of the DFDs, scheduling the meetings, documenting the threats found, and creating and following up the risks identified in the meetings. Clearly, they need to be “coached/trained” to build the skills needed to perform these activities, and our role as researchers doing action research with them, influenced this process and helped them to get the needed skills.

RQ3: What are the challenges/best practices of applying STRIDE? During our observations we noticed that many times the use of STRIDE efficiently limited the discussions to the threats that were related to the trust boundaries and communication channels in the system, thereby neglecting potential threats relevant to other parts of the product. Further, we also noticed that occasionally the team was not able to identify any STRIDE-related threats at all, which may lead to a false sense of security when using this approach.

Another challenge was that the list of identified threats was in many cases too vague to be useable. Following up on the identified threats was therefore challenging. After each session we asked the Security Champions in each team to formalise the list of threats into risks. However, the Security Champions did not feel completely comfortable to write them down, and sometimes we needed to ask them many times about it. The meeting also did not focus much on the decision of impact and probability of the threats or on which mitigation actions would be done, because it would take too long. This was a follow-up that the Security Officer had to do with the security champions. We also asked the team to contribute to the documentation of the threats, but most teams did not prioritise this.

Finally, we observed that the list of threats often created a time concern in the development teams. More specifically, the team members were not sure when they would have time to prioritise threats from the parts of the product that they thought had already passed the "definition of done".

Regarding best practices, as for the DFDs, the use of a security expert for facilitating the meetings and the inclusion of Security Champions in the teams turned out be a very positive aspect.

RQ4: Is the microsoft threat modelling tool helpful? The teams in this organisation did not use MS-TMT. We therefore do not have any relevant findings related to this topic.

RQ5: How can we make the results from threat modelling more useful to agile teams? Our observations in this study revealed that there is a need to clarify to the teams the benefits of doing threat modelling in their project, while still acknowledging the possible impacts in time and costs, not only for the meetings themselves, but also on the time needed to prepare the meetings and to follow up on the outputs of the meetings. In addition, the positive side effects of threat modelling should be highlighted, such as: better documentation of the system, awareness of security issues for all team members, better confidence on the way security is addressed in the team, and better visibility of the threats for other stakeholders such as the product owners, managers and the executives.

4.2. Study B: Threat modelling in agile teams (II)

Study B is an interview study, which was performed in 2018. The study focused on threat modelling in agile teams. The intention of the study was to examine how and to what degree Norwegian organisations perform threat modelling as part of their software development activities, what best practices they use, and what challenges they face.

The results from this study has already been published in full in the paper by Bernsmed and Jaatun (Bernsmed and Jaatun, 2019). This section therefore only reproduces the main findings from the study.

4.2.1. Industrial context

The participating organisations in this study are all based in Norway. Their core product is software, and their development teams all employ agile methods. In this section, we refer to the four organisations that we interviewed as Organisation A, B, C and D, respectively.

Organisation A is a supplier of IT and electronic ticketing systems to the public transport sector. They have around 100 employees, whereof the majority are stationed at their Norwegian headquarters. The organisation also operates two offices abroad, where some of the developers are working. We interviewed the Security Champion of one of the development teams stationed in Norway. This particular team is responsible for maintenance of the organisation's existing software product; they rarely develop any new services.

Organisation B is a provider of digital identification and authentication services. The company has around 30 developers, who are stationed in Norway as well as abroad. Three persons participated in the interview; the Chief Information Security Officer (CISO) and two of the security managers.

Organisation C is a small start-up, which offers products and services to help improve the digital security of their customers. We interviewed the person responsible for integrating security in the organisation's development and deployment processes (who referred to himself as the "SecDevOp"), who in addition also is the Chief Executive Officer (CEO) of the company.

Organisation D is a software development company that delivers software and related services to customers in the energy sector all around the world. They have around 20 developers, whereof ten are stationed in Norway. Three persons participated in the interview: a technical consultant, the Software Development Manager and the Chief Architect (the latter two participated through Skype).

4.2.2. Data collection and analysis

In this study we sought to extract in-depth information on how the participating organisations perform threat modelling in practice. We therefore chose a qualitative research method based on relatively few informants, which enabled us to perform a rich and detailed analysis of the selected organisations. More specifically, we performed a multiple case study (involving the four previously mentioned organisations) and we used semi-structured interviews as the main source of information. Interviews are a well-known and powerful tool for information collection in qualitative research. They give the researchers insights into the research topic from the interviewees' perspective (Myers and Newman, 2007). We used what is referred to in the literature as semi-structured interviews, which are driven by open questions, have a limited degree of structure, and tend to focus on specific situations and experiences made by the interviewee (Cassell and Symon, 2004). The interviews were performed either face-to-face or over Skype. No personal data was recorded during the interviews. The interview guide has been included in Figure 5 in Appendix.

Since we wanted to derive patterns from our observations, rather than evaluating existing hypotheses, we used the inductive research approach described in the paper by Thomas (Thomas, 2006) to interpret the collected data. The paper includes a systematic set of procedures for analysing qualitative data and explains a straightforward approach for deriving findings guided by research questions.

4.2.3. Main findings from Study B

Here we provide a summary of the main findings of this study, which are related to our research questions. For more details of these findings, we refer to the paper where they were originally published (Bernsmed and Jaatun, 2019).

RQ1: How are agile teams doing threat modelling today? A common factor identified in all four of the interviewed organisations, was the involvement of developers in the threat modelling activities. This appeared to be a successful approach, regardless of whether the developers were only contributing to the TM activities (as in Organisation A), or if they did most of the work themselves (as in Organisation B, C and D). However, the initial step of identifying assets was not done by developers in any of these organisations; the common opinion appeared to be that this is an exercise for people higher up the "food chain", who have the business perspective and understand the value of the organisation's assets.

All four organisations relied on the use of checklists, clearly defined processes, routines and the like, at least to some degree. Organisation A used them because they felt they lacked experience to identify and assess all relevant threats themselves. Organisation B said checklists and the like were particularly useful to their newly employed developers. Organisation C also advocated checklists, however without further specifying why. Organisation D used guidance documents for threat modelling, which they had developed internally.

All four organisations used an agile software development process with short iterations and they all pointed out the benefit of doing threat modelling activities at regular time intervals.

RQ2/RQ3: What are the challenges/best practices of doing data flow diagrams/applying STRIDE? Lack of motivation was identified as a common challenge in three of the four organisation. In Organisation A, the security champion could at first not explain why they did threat modelling, even though he later stated that it is good for learning purposes and that it eventually may lead to a more secure product. In Organisation B, the lack of motivation amongst some of the developers was illustrated by their tendency to skip it when they have the chance. Organisation C, on the other hand, seemed to be highly motivated, but their motivation was related to having the right "mindset" rather than on going through all the steps of the threat modelling activities. On the other hand, lack of motivation was not referred to as a challenge at all by the interviewees from Organisation D who, in contrast stated that all of their involved employees were motivated to participate in the threat modelling sessions.

Identifying relevant threats was another challenge identified in all the four organisations that we interviewed, even though the reason why differed. In Organisation A, the security champion felt that he lacked experience and that existing guidelines were too shallow. He also considered this as a time consuming activity. In Organisation B, the CISO suspected that relevant threats could be overlooked, because the developers did not understand the value of the information they processed. Organisation C's opinion was that doing threat identification in accordance to STRIDE was not worth the effort. Finally, Organisation D often found it hard to know whether, for example, publicly know threats were relevant to them or not.

That threat modelling is time-consuming was also reported as a challenge by the first three interviewed organisations. In Organisation A, they found it hard to find time to do these activities, in addition to all their other obligations. Organisation B had a "worst case" example where a lot of time had been spent on these activities. Finally, as mentioned above, Organisation C considered in particular the threat identification and analysis part to be a waste of time. Time was not perceived as a challenge in Organisation D though; the reason for this is most likely that they always have already existing DFDs to be used as a baseline and that the amount of documentation that they produce is minimal.

Finally, none of the four organisations had a clear definition of when they were "done" with the threat modelling activity. Organisation A stopped when they felt they had spent "sufficient time" (without being able to quantify this). Organisation B had a clear definition of when the risk assessment was done, but not of the individual threat modelling activities that they did during this assessment. Organisation C, on the other hand, stated that they will never be done; their approach is to think about threats constantly, at all times; a mindset that they seem to share with Organisation D.

RQ4: Is the microsoft threat modelling tool helpful? In this study, it was only Organisation A that used MS-TMT on a regular basis. They found it to be helpful for creating Data Flow Diagrams and for generating an initial set of threats, which they then supplemented by means of OWASP Top 10 (OWASP, 2017) and with an internally pre-generated table that included relevant threat actors and their motivations.

RQ5: How can we make the results from threat modelling more useful to agile teams? The findings show that, even though all of the four organisations considered threat modelling to lead to a more secure product, they all struggled with practical aspects of the established theory. On the bright side, our study also revealed that many things worked really well. In particular involving the developers in the threat modelling activities, using checklists and clearly defined processes and routines, and triggering the threat modelling activities at regular time intervals stood out as best practices that all the four organisations had (at least partly) adopted with good results.

4.3. Study C: Threat modelling using MS-TMT

In this study, we performed a controlled experiment, a scientific method for identifying cause–effect relationships, and thus a means to "acquire general knowledge about which technology (process, method, technique, language or tool) is useful for whom to conduct which tasks in which environments" (Sjøberg et al., 2005). The experiment was performed in the fall of 2019 and included approximately 100 MSc and BSc students from the Norwegian University of Science and Technology (NTNU) and the University of Stavanger (UiS). The purpose of the experience was to investigate the students' acceptance and usage of two different threat modelling techniques; one of them being the Microsoft Threat Modelling Tool (MS-TMT), thereby contributing to answering *RQ4: Is the Microsoft Threat Modelling Tool helpful?*

4.3.1. Case: Digital exam system

The experiment focused on a scenario in which a fictitious university had decided to procure a digital exam system. The digital exam system has three main types of users: students, teachers and external censors, and it supports four main functions: (1) Creation of exams by teachers (external sensors are also able to help), (2) Safekeeping of exams until the exact time the examination begins, (3) Examination, including hand-in of completed exams, and (4) Grading of completed exams by teachers and external censors. In the scenario, we also assumed that the university wanted to add a new functionality to the digital exam system: (5) Storing of results (grades) in a database at the university.

Before the experiment started, we had already modelled the first four functionalities in a Data Flow Diagram (see Figure 6 in Appendix). The task of the students participating in the experiment was then to do threat modelling of the new envisioned functionality (i.e., storing of results in the database at the university). More specifically, we asked them to (1) add the new functionality to the existing DFD, and (2) identify relevant threats to this new functionality, using STRIDE. The intention of these two tasks was to let the students get hands-on experience with two of the core activities in threat modelling, within a reasonable time, without having to do any particular preparations, using a real-life example.

4.3.2. Experiment set-up

The experiment started with a lecture, in which the students were introduced to the basic concepts of threat modelling, including how to draw Data Flow Diagrams (DFD) and how to do threat identification using STRIDE. The students were then introduced to the case described above. The students were divided into two groups, where one group was asked to update the existing DFD with the new functionality and identify the relevant threats manually, using pen and paper; and the other group was asked to do the same task, but by using the Microsoft Threat Modelling Tool (MS-TMT tool). It should be noted that the students were not randomly assigned into these two groups; those who had access to a laptop PC had been asked in advance to pre-install MS-TMT, make sure that it worked, and to bring their PCs to the lecture. These students were then assigned to the "MS-TMT group". The remaining students, who either did not own or had not brought any PC with them, were assigned to the "pen and paper group". Incidentally, these two groups turned out to be almost equal in size.

The students were encouraged to work in pairs to solve the task, but they were not allowed to interact with other pairs. When they were finished, they were asked to hand in their results and to fill out an evaluation sheet (one per student).

4.3.3. Data collection

Two types of data sources were generated in this experiment; the threat models that the students produced; i.e., the updated DFDs and the list of threats that they had identified as relevant for the new functionality, and their answers in the evaluation sheet.

We received in total 45 completed evaluation sheets; 20 from the students who had used the "pen and paper" approach to threat modelling and 25 from the students who had used MS-TMT. The results from this data source will be presented in the next subsection. Unfortunately, very few students chose to hand in their threat models, and we were therefore not able to use these as a source of data in our data analysis. No identifiable information or any kind of personal data was collected in this experiment.

4.3.4. Evaluation of user acceptance and usage

The Technology Acceptance Model (TAM) (Chuttur, 2009) identifies *perceived usefulness* and *perceived ease of use* as the main variables for determining user acceptance of new technologies. To measure the students' acceptance and usage of the two different threat modelling techniques, we used a modified version of the TAM, which is commonly referred to as "TAM2" (Venkatesh and Davis, 2000). Using TAM as the starting point, TAM2 incorporates additional key determinants of TAM's perceived usefulness and usage intention constructs, thereby allowing one to better understand the user acceptance of the evaluated technology. More specifically, TAM2 also includes "Subjective norm", "Image", "Job relevance", "Output quality", "Result demonstrability", "Experience" and "Voluntariness" as additional variables to the model.

Since the subjects in this study were students, we had to make some modifications to the questions in the original TAM2 paper (Venkatesh and Davis, 2000). More specifically, we chose not to include the questions in the categories "subjective norm" and "image", since these are specifically focusing on how the subjects' opinions are influenced by colleagues that they have professional relations with. We also had to adapt the wording of those questions that were formulated for a work environment, to better fit into the context of the university students[3]. The

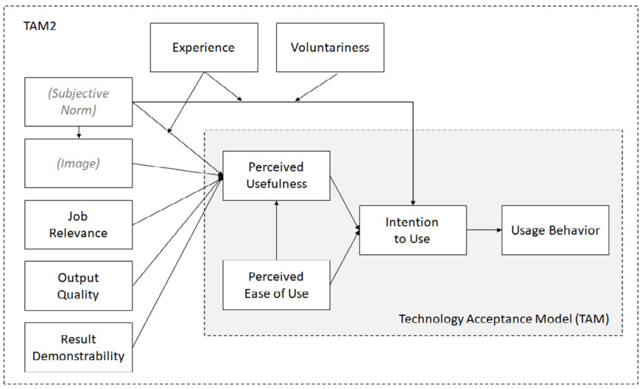


Fig. 1. The modified version of TAM that was used to evaluate the user acceptance of threat modelling in Study C.

value range used in the questionnaire was 1–5 where 1 = strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, and 5 = strongly agree.

In addition, to include the TAM2 attribute "Experience" in our analysis we also asked the students whether they had any previous experience in any of the following roles: (1) developer, (2) architect, (3) tester, and/or (4) product owner.

Fig. 1 provides an overview over the evaluation model that we used.

An overview over the questions and the average score (mean values) of the answers from the students is provided in Fig. 2. The average score (mean values) of the answers from the students who used the MS-TMT in Study C is provided in Fig. 3.

4.3.5. Main findings from Study C

The findings from this study are mainly related to *RQ4: Is the Microsoft Threat Modelling Tool helpful?*

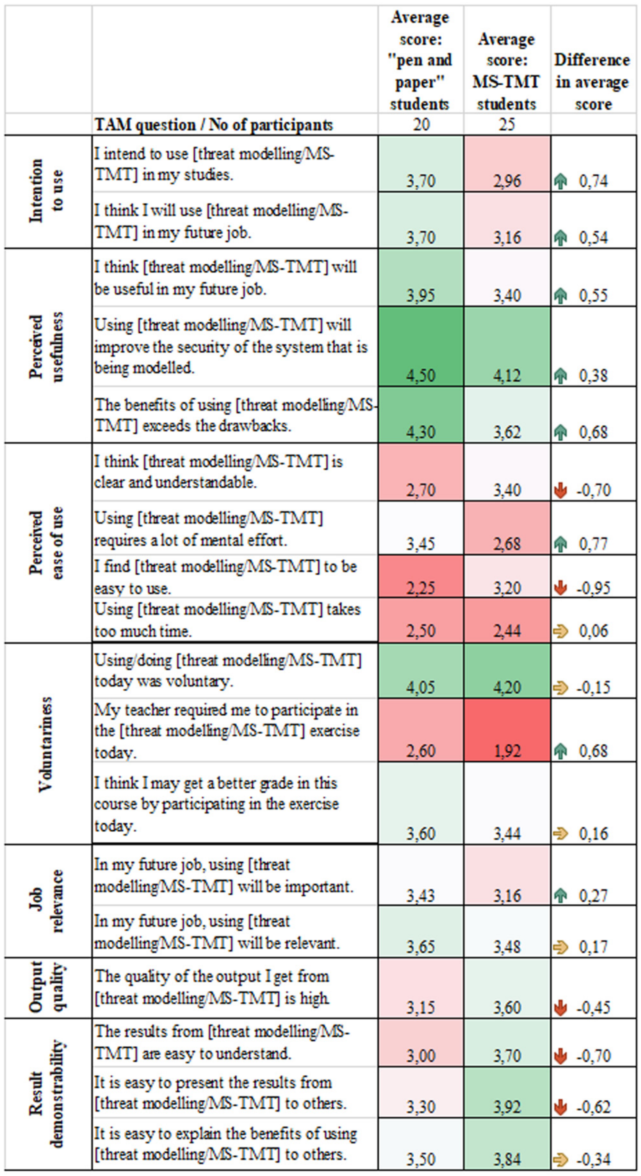
As can be seen in Fig. 2, both modelling methods received a very good score in terms of their perceived usefulness, in particular w.r.t whether the students think they will improve the security of the system that is being modelled (4.50 for the pen and paper students and 4.12 for the MS-TMT-students, respectively). Regarding their intention to use, the students who had modelled using pen and paper were slightly more positive than the students who had used the MS-TMT (3.70 and 3.70, compared to 2.96 and 3.16). Both modelling methods received rather low scores on their perceived ease of use (most of the average scores are below 3.00, which is indicated by the red colour in the figure).

Regarding output quality and result demonstrability, the students who had used MS-TMT were consistently more positive (all their average scores are above 3,00, as indicated by the green colour shading) than the students who had modelled using pen and paper (whose average scores are consistently below the MS-TMT students' average scores in these two categories).

The biggest difference in average score between the different groups of students is for the question *"I find [threat modelling/MS-TMT] to be easy to use"*, where the students who had used pen and paper were far more negative than the students who had used MS-TMT (average score was 2.25 for the former and 3.20 for the latter).

Looking further into the opinions of the students who used MS-TMT (Fig. 3), there are only minor differences in the average

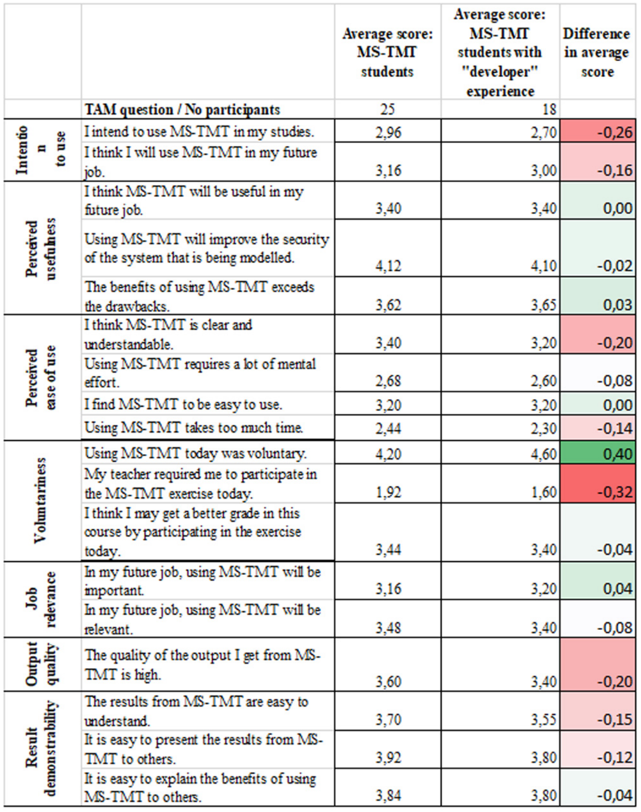
[3] For example, the TAM2 question "*Assuming I have access to the system, I intend to use it*", which measures the subject's intention to use the system in his/her daily job, was reformulated to "*I intend to use the [threat modelling/MS-TMT] in my studies*". Similarly, the TAM2 question "*In my job, usage of the system is important*", which measures the relevance of the system to the subject's work position, was reformulated to "*In my future job, using [threat modelling/MS-TMT] will be important*".



	TAM question / No of participants	Average score: "pen and paper" students	Average score: MS-TMT students	Difference in average score
		20	25	
Intention to use	I intend to use [threat modelling/MS-TMT] in my studies.	3,70	2,96	⬆ 0,74
	I think I will use [threat modelling/MS-TMT] in my future job.	3,70	3,16	⬆ 0,54
Perceived usefulness	I think [threat modelling/MS-TMT] will be useful in my future job.	3,95	3,40	⬆ 0,55
	Using [threat modelling/MS-TMT] will improve the security of the system that is being modelled.	4,50	4,12	⬆ 0,38
	The benefits of using [threat modelling/MS-TMT] exceeds the drawbacks.	4,30	3,62	⬆ 0,68
Perceived ease of use	I think [threat modelling/MS-TMT] is clear and understandable.	2,70	3,40	⬇ -0,70
	Using [threat modelling/MS-TMT] requires a lot of mental effort.	3,45	2,68	⬆ 0,77
	I find [threat modelling/MS-TMT] to be easy to use.	2,25	3,20	⬇ -0,95
	Using [threat modelling/MS-TMT] takes too much time.	2,50	2,44	⇨ 0,06
Voluntariness	Using/doing [threat modelling/MS-TMT] today was voluntary.	4,05	4,20	⇨ -0,15
	My teacher required me to participate in the [threat modelling/MS-TMT] exercise today.	2,60	1,92	⬆ 0,68
	I think I may get a better grade in this course by participating in the exercise today.	3,60	3,44	⇨ 0,16
Job relevance	In my future job, using [threat modelling/MS-TMT] will be important.	3,43	3,16	⬆ 0,27
	In my future job, using [threat modelling/MS-TMT] will be relevant.	3,65	3,48	⇨ 0,17
Output quality	The quality of the output I get from [threat modelling/MS-TMT] is high.	3,15	3,60	⬇ -0,45
Result demonstrability	The results from [threat modelling/MS-TMT] are easy to understand.	3,00	3,70	⬇ -0,70
	It is easy to present the results from [threat modelling/MS-TMT] to others.	3,30	3,92	⬇ -0,62
	It is easy to explain the benefits of using [threat modelling/MS-TMT] to others.	3,50	3,84	⇨ -0,34

Fig. 2. The average score (mean values) of the answers from the students in Study C. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

score between the students who had previous experience with software development (18 persons), compared to the students who had not previews experience in this field (7 persons). However, we observe that the students with developer experience were slightly more negative in their answers to the questions on their intention to use the tool in the future. They were also somewhat more negative in their answers to the question on the output quality of the tool, and whether they thought the tool was clear and understandable. Finally, these students also indicated that they had felt less pressure to participate in the exercise that the students who did not have any previous experience. Based on the findings presented above, it appears that the MS-TMT indeed is a very "useful" tool (cf. the questions on "Perceived usefulness"), the result from the tool is easy to understand and present to others (cf. the questions on "Results demonstrability"), but it is difficult to use the tool (cf. the questions on "Perceived ease of use"). We note that these are our preliminary conclusions, but we have used them as a data source when triangulating the



	TAM question / No participants	Average score: MS-TMT students	Average score: MS-TMT students with "developer" experience	Difference in average score
		25	18	
Intentio n to use	I intend to use MS-TMT in my studies.	2,96	2,70	-0,26
	I think I will use MS-TMT in my future job.	3,16	3,00	-0,16
Perceived usefulness	I think MS-TMT will be useful in my future job.	3,40	3,40	0,00
	Using MS-TMT will improve the security of the system that is being modelled.	4,12	4,10	-0,02
	The benefits of using MS-TMT exceeds the drawbacks.	3,62	3,65	0,03
Perceived ease of use	I think MS-TMT is clear and understandable.	3,40	3,20	-0,20
	Using MS-TMT requires a lot of mental effort.	2,68	2,60	-0,08
	I find MS-TMT to be easy to use.	3,20	3,20	0,00
	Using MS-TMT takes too much time.	2,44	2,30	-0,14
Voluntariness	Using MS-TMT today was voluntary.	4,20	4,60	0,40
	My teacher required me to participate in the MS-TMT exercise today.	1,92	1,60	-0,32
	I think I may get a better grade in this course by participating in the exercise today.	3,44	3,40	-0,04
Job relevance	In my future job, using MS-TMT will be important.	3,16	3,20	0,04
	In my future job, using MS-TMT will be relevant.	3,48	3,40	-0,08
Output quality	The quality of the output I get from MS-TMT is high.	3,60	3,40	-0,20
Result demonstrability	The results from MS-TMT are easy to understand.	3,70	3,55	-0,15
	It is easy to present the results from MS-TMT to others.	3,92	3,80	-0,12
	It is easy to explain the benefits of using MS-TMT to others.	3,84	3,80	-0,04

Fig. 3. The average score (mean values) of the answers from the students who used the MS-TMT in Study C.

results from our different studies in Section 5. Threats to validity and limitations associated with this study will be discussed in Section 6.

4.4. Study D: Threat modelling in agile teams using MS-TMT

Study D is an interview study, supplemented by data gathered from document analysis, in an agile software organisation, which was performed during the spring of 2020. The intention of the study was to investigate how development teams apply threat modelling (TM) in their projects, which challenges they face and what best practices they apply. We also wanted to study the teams' experiences with the Microsoft Threat Modelling Tool (MS-TMT).

4.4.1. Industrial context

The company that participated in this study is a software company, which simplifies and digitises core business processes in the private and public sector. Their headquarter is located in Norway, but they have a presence across the entire Nordic region, as well as in Benelux and in central and eastern Europe. The company is an amalgamation of more than 145 individual agile companies, each with its own culture and way of working. Due to this diversity, each of the sub-companies is composed of one or more self-managed agile teams, which are responsible for the security of their own services. A centralised Product Security Team (PST) is driving the software security efforts on an overall level, by creating and enforcing a secure development lifecycle, which is a standardised way of working with security across the individual teams.

4.4.2. Data collection and analysis

Seven teams participated in this study. They were carefully selected to ensure diversity, in terms of different team sizes, different types of software products, different geographic locations and different maturity of security in their projects. They also had varying histories regarding their team members and their products; some had been acquired by the company while others had been developed internally. The data collection was performed through a two-step process. In the first step, we collected and analysed the DFDs that the selected teams had produced in their threat modelling activities. The intention of this first step was to understand upfront the approach for the DFDs from those particular teams' perspectives, and to identify further questions that we would focus on during the interviews.

In the second step, we interviewed the security engineers from these seven teams. In this organisation, the "security engineers" are team members who are responsible for promoting and supporting the adoption of security activities inside the team, without breaking the agility, continuous delivery, self-management and autonomy of their projects. The security engineers were asked about how they and their team members integrate threat modelling in their software development activities. More specifically, we asked about the drawing of DFDs, their usage of MS-TMT and STRIDE, and about their impression and opinions of these activities overall. They were also asked to give concrete examples of challenges and best practices that they had adopted when doing threat modelling in their projects. All the interviews were performed over Google Meet.[4] and lasted about 1 h. The interview guide has been included in Figure 7 in Appendix.

We then transcribed the interviews and performed open coding of our analysis of the DFDs and of the interview transcripts using the MaxQDA tool.[2] The coding resulted in 22 codes from the interview transcripts and 19 codes from the document analysis of the DFDs. We then organised the codes according to the five research questions, and analysed them with the intention of deriving relevant findings.

4.4.3. Main findings from Study D

Here we provide the main findings of this study, which are related to our research questions.

RQ1: How are agile teams doing threat modelling today? Drawing Data Flow Diagrams (DFDs) is part of the security development lifecycle in company D, and each team is required to update their diagrams at least once a year. Most of the times these diagrams are created and maintained by the security engineers, sometimes with help from other senior team members, such as the system architect. In this company, the Microsoft Threat Modelling Tool (MS-TMT) has been selected as the preferred tool to draw the DFDs. The teams have not received dedicated training for using this tool, but they receive support from the centralised PST (Product Security Team) during the review of the diagrams, or whenever they have questions.

When drawing the DFDs, the teams focus mostly on connections and integration to systems outside the scope of their own system. Most of the development teams draws the DFDs while at the same time doing analysis on the connections to find possible problems with information disclosure. Some teams also looked at possible risks related to tampering with information. A few of the teams also added information about different types of users and logging to their DFDs.

In this company, there is no a mandatory threat identification and analysis activity included in the secure development lifecycle. Even though MS-TMT provides support for doing threat analysis using STRIDE, most teams do not use this functionality. However, the PST has designed a very detailed questionnaire for gathering information about the security of the products and services, including questions about input validation, denial of service, elevation of privilege and other relevant attacks. The aim is to complement the drawing of the DFDs and to make teams more aware of possible threats. According to the security engineers (as the security champions are denoted in the company), they usually do not find threats solely based on the DFDs, but when they do, they create Jira tasks to investigate these further.

Regarding the "definition of done" of the drawing of the DFDs, most of the security engineers mentioned that they decided to stop when all the connections and integration to other systems and services had been included in the diagrams. In order to do a review of the diagram, some of the security engineers present it to the rest of their teams and discuss around it to make sure nothing was missing.

Regarding the time aspect, we did not find any evidence that the data flow diagrams had been created or updated before the actual implementation of the features in the code; in most cases they were used to documented changes to the system or software, after they had been implemented. In general, the creation of the diagrams was seen as a time consuming and tedious task, which required two persons spending at least two full days to complete it, occasionally even spread over up to three weeks. This could be the reason why updates of the diagrams usually only are done once a year in this organisation, the exception being some teams who do updates whenever new interactions are added to their systems.

Before drawing of DFDs was included as a compulsory part of the security development lifecycle, most teams did not have any formal drawing of their systems. Some of them had architecture diagrams; however, according to the security engineers, those diagrams could not be reused, because they did not include connections and interactions with other systems.

RQ2: What are the challenges/best practices of doing data flow diagrams? During the interviews, the security engineers raised several challenges related to the drawing of the DFDs. Firstly, they affirm it is challenging to identify which elements to include in the DFD and how detailed that information should be. One team commented about the lack of security knowledge to do this type of work. They also found it hard to know when the DFD is good enough. For bigger systems it is even more challenging, because the task easily becomes time consuming, it is hard to decide where to focus, and the drawings often become crowded. One team divided their DFDs into different sub-diagrams: high level (including external connections) and low level (focusing on the internal structure).

Overall, the interviewees stated that the task is time consuming and boring, and the teams did not seem to understand the usefulness of this activity. Many teams also commented about the lack of guidelines and templates to help them getting started with the process. As one interviewee expressed it: "*What I lacked when we were doing this was the lack of guidelines, because no one in our team had done this before. At that point it would have been good to have got a bit more guidelines on how to structure it and what to use it for. Because if we had started off doing it correctly from the beginning and looking at it such as first adding a service, then configure it, generate a report, before adding new services. We might have gotten tips on how to proceed. But since we did not have any guidelines, we just started to add everything and then when we created the report, it was so big that we thought we did not have time to do it and we gave up. If we maybe we had done in a more structured way it would have been more useful*". Another challenge frequently mentioned was the need to have a senior member, or the security engineer, involved in the drawing of the

[4] https://meet.google.com/.

DFD, especially in cases where the person(s) who had produced the code had left the company.

When doing document analyses of the DFDs, we noticed that some of the teams had struggled to correctly identify and model the trust boundaries of their systems, especially for cloud solutions. The reason is most likely that this concept is very different in cloud environments compared to in-house solutions. Also, the MS-TMT does not provide much support, in terms of available elements, when modelling cloud systems.

From the interviews we identified some recurring topics that the teams perceived as good practices:

- The drawing of the DFD is seen as a good practice. According to the teams, including all components and connections in such a diagram is a good way of documenting their systems,
- Making sure that all connections and interactions with other systems are included, is a good criterion for "Definition of done" for the DFD;
- Asking for help from a security expert (in the PST or from another team) whenever needed;
- Evaluate when to involve other team members in the discussions;
- Using the DFDs drawings for onboarding of new employees. The diagrams can also be used in customer meetings;
- Updating the DFD every time there is a new integration or any other big change to the system;
- Sharing of knowledge between the teams, in particular on experiences and lessons learnt from drawing the DFDs.

RQ3: What are the challenges/best practices of applying STRIDE? In this study we can conclude that STRIDE does not seem to be a concept that is in the mind of the Security Engineers when they do the analysis of the DFD. When they were asked about STRIDE they needed reminders or explanation of what each of the letters mean. Teams mentioned that, in general, except when using the Threat Modelling Tool, they do not use STRIDE as a concept. For example when asked if they use STRIDE, one Security Engineer mentioned: "*Not really those terms, I do not think we use them. We do not use STRIDE outside of data flow diagram.*" Some mentioned that STRIDE seems to be more theoretical than practical.

Because the focus of the DFDs was mostly on the connections, the Security Engineers could not visualise how they can do analysis on elevation of privilege using these diagrams. Mostly they believe that elevation of privilege is already handled by the authentication/authorisation mechanisms that are implemented in their systems.

RQ4: Is the microsoft threat modelling tool helpful? In this company, the Microsoft Threat Modelling Tool (MS-TMT) is the default tool used for drawing the data flow diagram. However, since there is only a Windows version of the tool, some teams had decided to use draw.io[5] instead. When comparing the two products, the teams mentioned the ability of describing the security properties of each element in MS-TMT as an advantage. On the other hand, they found MS-TMT more difficult to use than draw.io when two or more people needed to collaborate to draw a diagram.

Regarding usability and usefulness, the teams found it easy to learn to use the drawing functionality of MS-TMT, however, the rest of the functionalities were not appreciated. For example, they commented about having to add all properties, whereof many do not make sense to them, before they could generate a useful threat analysis. They also reported that the analysis results in many false positives. In addition, one team pointed out that the tool does not work well for modelling and analysing cloud-based solutions: *"Because the tool is not very good for cloud. Sometimes I feel like the tool does not understand what we do. Does not understand the communication level, a lot of misleading questions about the properties of the elements. When you have the cloud you have other levels of security. And we had to answer a lot of questions about each link you make that it did not make sense or we do not have to deal with those things anymore because we use the cloud. The communication protocols, asks different questions that are misleading. We spent a lot of time trying to understand what everything meant".*

According to the security engineers, the teams tend to give up quickly when they realise that the analysis task is time consuming. They also find it boring and they claim they do not find any useful information by analysing the report that is generated by the tool. The few relevant findings that had been discovered by the Security Engineers were threats related to insecure connections. One security engineer said: *"We tried to set properties in the Microsoft Threat Modelling Tools in the communications part. But that is not complete, so we have not used the analysis part of the tool. We tried earlier and we thought it was a lot of things to look at. So if we had more time we could have done but I do not know how much this will give us".*

Regarding the drawing functionality, the teams are pleased to have a full overview of the system, which they think is helpful, but they also mentioned several drawbacks:

- The list of pre-defined elements lacks many of the services and components that are commonly used nowadays (for example, in cloud-based environments);
- The list of pre-defined data flows does not include two-way communication connections, hence forcing the teams to add two separate connections for each data flow;
- Complexity, and lack of readability, quickly become a problem when models grow large.
- There is a lack of filtering or other visualisation of the properties, especially when they need to update the diagram.

When asked whether they would recommend other teams to use the MS-TMT, most of the interviewed security engineers said that yes they would recommend the tool to other teams, pointing out that the diagrams provide a good overview of the system, and can serve as a good instrument in further discussions about security.

RQ5: How can we make the results from threat modelling more useful to agile teams? According to the security engineers, the Data Flow Diagrams (DFDs) would be more useful if the PST could provide better guidelines for how to create them, for example on how to split them into separate diagrams to represent different levels of abstractions. The teams should also be better informed on how they could use the DFDs for security discussions, in selling processes and for onboarding of new employees. The DFDs would also be useful if the teams could use them to support the verification of security when migrating products to other cloud providers or micro-services architectures.

Once the product is growing in complexity and size, and more teams start to use micro-service architectures, one challenge is the identification of relevant components to include in the DFDs. Since most of the company products are stored in cloud environments, it will be helpful if MS-TMT could be integrated with existing cloud systems, in order to facilitate self-discovery of the relevant system elements. Also, in order to reduce the number of false positives in the threat analysis report, the tool should be better on modelling the different cloud protection mechanism/services. This means that the tool has to continuously integrate new technologies and stay up to date with the new development approaches.

5 https://drawio-app.com/.

As discussed, simply just drawing the DFDs did not help much in identifying threats. Neither was STRIDE considered to be particularly useful. The questionnaire, on the other hand, was of great help, since it asked concrete questions, such as "How do you do input validation?", "Where do you store your logs?", etc. Such questions made it easier to identify relevant threats, which could then be added to the issues tracking systems (such as Jira) for further investigation and following-up by the teams and/or the PST.

One of the reasons why the teams found threat modelling to be time consuming and tedious, was because they were not sure if what they did was what they were supposed to do. Therefore, having an security threat modelling expert to do a walk-through with the team, where they start drawing and analysing the diagrams, may help them understand the level of abstraction that best suits their system. This would help the teams improve the readability of their diagrams, and eventually also help them identify and mitigate the threats that are relevant for their products.

Finally, to be more useful, there should be a better integration between the threat modelling activities and the development pipeline, which today in this organisation is seen as two separate activities.

5. Results achieved from triangulation

To better answer the research questions, we triangulated the results from the four studies presented in Section 4.1–4.4. In this section, we will go into the details of these results. An overview over which studies that have contributed to which findings is presented in Tables 3–6.

5.1. How agile teams are doing threat modelling today

Data regarding the use of threat modelling in agile teams (RQ1) were collected in Study A, B and D. The results that we derived from these studies are:

Finding: Asset identification is not included in the threat modelling activities. Asset identification and prioritisation is usually viewed as the first activity in a threat modelling process (Jaatun et al., 2019). However, as was shown in Study A, B and D, it is very rare that agile software development teams go through this particular activity every time they do threat modelling. In fact, developers are often not even involved in this part at all; for example, in Study B, one of the interviewees explained that the assets had already been identified on the management level and stored in the company repository, which could then be accessed by the developers whenever needed.

Finding: The threat modelling is done by the developers. The involvement of developers in the threat modelling activities appeared to be common practice in all of the participating organisations in Study A, B and D.

Finding: The threat modelling is not done as part of the daily activities. Another common factor identified in Study A, B and D, was that threat modelling was not fully integrated as a part of the daily activities of the teams, but rather performed as a separate activity in dedicated sessions. The collected data also indicated that threat modelling was by many perceived as an additional activity that took time away from the actual development work.

Finding: Threat modelling is scheduled at regular time intervals. Doing threat modelling at regular time intervals was a common intention amongst a majority of the participants in Study A, B and D, even though it varied how well they managed to fulfil it in practice. The organisation participating in Study A had just got started with threat modelling, but they planned to keep doing it on a regular basis. In Study B, the interviewees from all the four participating organisations pointed out the benefit of doing threat modelling regularly, even though it varied how often they managed to do it in practice. In Study D, the interviewees claimed they strive to do this on a regular basis, but with varying success. The secure lifecycle in this organisation demanded that this activity should be done once a year and therefore the teams followed the demands from the procedure. Further, some mentioned that they also do the activity when they feel like they have done too many changes to the architecture of the system.

Finding: Relevant threats are found through checklists. Even though the process for identifying threats varied in the three studies, all of the participating organisations relied on some kind of supporting material. In Study A, they used STRIDE to generate lists of relevant threats. In Study B, the four interviewed organisations all informed that they used checklists, guidance documents and other types of supporting documents to identify threats. Similarly, in Study D the participating organisation relied on a questionnaire to find relevant threats. The participating organisations all seemed to agree that the checklists etc. helped them identify relevant threats.

5.2. Challenges and best practices of doing data flow diagrams

Data regarding doing Data Flow Diagrams (RQ2) were collected in Study A and D, and to some degree also in Study B. The results that we derived from these studies are:

Challenge: Lack of motivation. This challenge was raised by the participants in Study A, B as well as D. Common concerns were that threat modelling is considered to be time consuming and tedious, and that it is difficult to motivate the developers to attend the sessions. In Study A, the participants also claimed such activities takes time from the development work and hence impacts their productivity. A related issue that was voiced in both Study A and D, was that some the teams do not know what to do with the results afterwards.

Challenge: Level of detail to include. The challenge of setting the right level of abstraction when drawing the DFDs was raised as an issue by participants from both Study A and D.

Challenge: Producing accurate models. How to correctly model the software that they were working on, as a DFD, was raised as a challenge in both Study A and D. Issues mentioned included difficulties to map the models with interfaces to other systems (Study A), challenging to link to actual code (Study A) and problems with defining trust boundaries (Study D).

Best practice: Use DFDs to document the systems. In Study D, it was revealed that the DFD is a good way of documenting the system that is being developed, and that it also can be used for onboarding of new employees and for customer meetings. We note that this finding partly contradicts with the previously identified challenge of lack of motivation.

Best practice: Involve security expert. Both Study A and D showed that involving a security expert when drawing the DFDs is in general a good practice.

Table 3
Triangulating the results from the different studies to answer RQ1.

Research Question	Result	Study A	Study B	Study C	Study D
RQ1: How are agile teams doing threat modelling (TM) today?	Finding: Asset identification is not included in the threat modelling activities.	x	x		x
	Finding: The threat modelling is done by the developers.	x	x		x
	Finding: The threat modelling is done in dedicated sessions.	x	x		x
	Finding: Threat modelling is scheduled at regular time intervals.	x	x		x
	Finding: Relevant threats are found through checklists and the like.	x	x		x

Table 4
Triangulating the results from the different studies to answer RQ2.

Research Question	Result	Study A	Study B	Study C	Study D
RQ2 (part I): What are the challenges of doing Data Flow Diagrams?	Challenge: Lack of motivation.	x	x		x
	Challenge: What level of detail to include.	x			x
	Challenge: Produce accurate models.	x			x
RQ2 (part II): What are the best practices of doing Data Flow Diagrams?	Best practice: Use the DFDs to document the systems.				x
	Best practice: Involve a security expert.	x			x

5.3. Challenges and best practices of applying STRIDE

Data regarding the use of applying STRIDE (RQ3) was collected in Study A and D, and to some degree also in Study B. The results that we derived from these studies are:

Challenge: Limited scope. The limited scope of STRIDE was identified as a challenge in both Study A and D. In Study A, the participating organisation claimed these types of threats have a very narrow scope, they are too vague to be useable, and that it is difficult to translate them into risks. In Study D, the interviewees could not even relate to the term; most of them they did not even remember what the acronym stands for. In Study B, one of the organisations even claimed that using STRIDE to identify threats was a waste of time.

Best practice: Generate initial set of threats. Despite its limitations, one of the organisations in Study A informed they found STRIDE useful for generating an initial set of threats, which they could then build upon and that this was something they had positive experiences from.

5.4. Usefulness of the MS-TMT tool

Data regarding the usefulness of the Microsoft Threat Modelling Tool (RQ4) was collected in Study A, C and D. The results that we derived from these studies are:

Finding: The tool works well for drawing DFDs. That the tool works well for drawing DFDs was shown in both Study A, where the organisation that used the tool on a regular basis explicitly said they found it useful for creating such diagrams, and in Study D where the interviewees told that the diagrams they draw by means of the tool represent a nice overview over their systems. The tool also received good scores on its "perceived usefulness" and "results demonstrability" in the student experiment (Study C).

Finding: The tool is easy to use. In Study D, it was shown that the teams found the drawing functionality easy to learn. We note however that this finding contradicts with the results from Study C, where the tool received rather low scores on its "perceived ease of use". However, as was shown in Study C, using the tool to model new functionality in a DFD was perceived far more easier than using pen and paper to do the same better exercise.

Finding: The tool lacks important elements. The interviewees participating in Study D considered it a main disadvantage that the tool lacks important elements, especially for modelling cloud services. They also pointed out that the tool lacks elements for modelling common mitigation mechanisms, such as firewalls.

Finding: The tool generates irrelevant list threats in the report. In Study D, it was pointed out that, in addition to having a limited scope (the challenge identified in RQ3), the tool generates lots of irrelevant threats, and hence false positives that the teams have to deal with.

Finding: The tool lacks support for collaboration. The lacking support for collaboration was identified as a major disadvantage in Study D. Since many of the developers in this organisation work remotely, using different platforms and technologies, they experienced challenges sharing information, not only between different teams, but also between members in the same team.

5.5. Making the results more useful to agile teams

How to make the results from threat modelling more useful to agile teams was the fifth research question defined in Section 1 (RQ5). By analysing our collected data, and cross-checking it with previous relevant research (Section 2) we have arrived at three main recommendations:

First, the threat modelling activities need to be better integrated into the agile software development life cycle. This has already been recognised in several previous studies (Weir et al., 2019; Heijden et al., 2018; Terpstra et al., 2017), and our results further confirm this conclusion. A starting point could be to integrate those parts of the activities that can be automated, such as the generation of relevant threats when new functionality is being modelled, and the generation of action points (such as Jira tickets) for those threats that have been reviewed and tagged to be relevant. It would also be valuable for the developers to have access to better, and more updated, templates that can be used to get them started with DFDs, with a minimal amount of effort.

Second, there is clearly a need for better tooling, to help developers draw their DFDs and to help them identify and analyse relevant threats, but MS-TMT does not seem to be an up to date fit. Threat modelling is a collaborative task, but the tool does not support online collaboration. Even though the focus in agile is on working software over comprehensive documentation, there is still value in this type of documentation. Having a tool that allows the developers to easily generate, update and maintain their diagrams will support their ability to adapt to changing requirements and continuous delivery, while at the same time helping them to deliver more secure software. Also, existing tools and techniques need to be kept up to date with the development of new technologies, in order to be and to stay relevant for agile software development teams. This was particularly visible in our data from Study D, where the developers explicitly requested better integration with their cloud system and existing security countermeasures.

Table 5
Triangulating the results from the different studies to answer RQ3.

Research Question	Result	Study A	Study B	Study C	Study D
RQ3: What are the challenges / best practices of applying STRIDE?	Challenge: Limited scope.	x	x		x
	Best practice: Generate initial set of threats.	x			

Table 6
Triangulating the results from the different studies to answer RQ4.

Research Question	Result	Study A	Study B	Study C	Study D
RQ4: Is the Microsoft Threat Modelling Tool helpful?	Finding: The tool works well for drawing DFDs.	x		x	x
	Finding: The tool is easy to use.			x	x
	Finding: The tool lacks important elements.				x
	Finding: The tool generates irrelevant threats.				x
	Finding: The tool lacks support for collaboration.				x

Third, the findings from our study includes the involvement of a security expert as an identified best practice when drawing the DFDs (see Section 8.2). Adding an expert to the development teams has also recommended in other studies, such as the one by Terpstra et al. (2017). However, one should be aware this is a quick fix that may not be practically feasible in all organisation and which may not help the teams in the long run. As was shown in the study by Weir et al. (2019), a developer-centric approach, in which the security expert is "sensitizing the developers to their security needs, allowing them to choose for themselves which tools and techniques to use", i.e., raising the awareness and the competence of the teams, is a better approach than telling the developers what to do and how to do it.

6. Discussion

There is a conflict between an academic desire to have a threat modelling scheme that is as accurate and detailed as possible, and developers' wish for tools that are painless, efficient and relevant. The former approach leads to scientifically satisfying solutions that end up being too cumbersome to actually be used in the real world. In this sense, this is a perfect example of "best" being an enemy of "good enough", to paraphrase Voltaire. The most difficult challenge we see is how to make the threat modelling activity an integrated part of the agile development life cycle and changing the mind set of the development team to constantly address security, in particular during big changes that happen on the design level of the products. We would like to emphasise the importance of improving threat modelling as an activity for finding security problems in the design early.

Our observations show that development teams perform this activity mostly for compliance and not because they see that it improves the security of the product, and thus there is low motivation for performing the activities early in the process. Furthermore, in dealing with this challenge, the issue of how to organise and maintain the availability of security experts for better analysis of the DFDs is an open question that needs to be further investigated.

We have also observed that following up the output of the sessions is harder than one would expect, and just creating issues on the issue tracking system does not always work, since issues that keep being ignored or deprecated eventually disappear from the issue tracking system without having been satisfactorily resolved. Development teams should also expect that some threats that are found will not be mitigated, but rather be subsumed as part of the accepted risks in the product.

The Microsoft TMT is a free resource that gives good support for the drawing of the DFDs and also support to describe some important characteristics of the elements for security. However, one should be aware that it is time consuming to get all the information needed for the tool to be able to create a useful list of threats.

6.1. Recommendations for practice

In this section, we outline the recommendations based on the cross-study analysis of the threat modelling activity, followed by a few general implications that we derived from the execution of our investigations. For the companies that are interested in threat modelling we advise:

Asset identification.

- Create a list of assets (Jaatun and Tøndel, 2008), involving infrastructure personnel; one possibility is to use different discovery systems such as the ones used in configuration management databases; if using infrastructure as code, the development team can extract these assets from there;
- Create an overview of cryptographic keys and other secrets, and specify how they are used in the system.
- Create a list of systems/services that send or receive data from the product;

Creating the data flow diagram.

- The diagrams should focus not only on the integration with other systems, but also on other aspects that should be addressed, such as users, logging, storage and trust boundaries (the latter is particularly important in cloud environments);
- The organisation can create templates and guidelines to support the drawing of the DFDs in which the most important aspects of the systems are highlighted;
- The guidelines should also contain the definition of done for the DFDs and examples on how the teams can create different levels of DFDs;
- Senior developers and domain experts should be involved in the creation of the DFDs because they know the history of the architectural decisions.

TMT - threat modelling tool.

- Companies should investigate if there are new tools in the market that have elements that better describe the contemporary context of development, such as being more suitable to cloud;
- One important functionality that TMT has is the description of the characteristics of the elements tailored to the specific type of element. Search of this functionality in other tools.
- If the development team does not intend to specify all the characteristics of the various elements, nor use the automatic threat generation functionality, any conventional drawing tool can be used to draw the DFDs.

Threat modelling sessions.

- When possible, the whole team should participate in the threat modelling discussions, not only seniors.

- If possible, involve a security expert who can help to support the discussions, focusing on the most important scenarios for security (Boström et al., 2006).
- Repeat these meetings once a year (at least), or whenever there is a substantial change in the design of the product.
- Appoint a team member (e.g., a security engineer), to make sure these meetings will happen.
- Establish a definition of done for the threat modelling session. For example, the security expert can use STRIDE as a way to define when the discussion is done, for example when all the letters in the acronym are discussed then the meeting is done.

Output of the sessions.

- The identified threats and issues from the modelling session should have a responsible assigned by the of the meeting, preferably more than one team member should be assigned to these issues for follow up.

6.2. Threats to validity and limitations

As any empirical study, this study has limitations. In the following, we discuss the reliability of our results along with threats to internal, construct and external validity, as described by Runeson and Höst (Runeson et al., 2012).

Reliability is related to the repeatability of a study, i.e., how dependent are the data and analysis on the involved researchers (Runeson et al., 2012). To minimise this threat, four researchers were involved in the design and execution of this multi-case study. It is well known that 'what people say' is often different than 'what people do'. The main benefit from this approach is therefore that by triangulating using evidence converging from different data sources, one will improve the validity of the final result (Creswell, 2014; Easterbrook et al., 2008). Furthermore, we developed an explicit case study protocol to guide the investigations on each study. Finally, the observations and findings were verified with the companies' representatives to avoid false interpretations and inconsistencies. Besides, the researchers have a long-term relationship with the case companies, which means that the researchers know more about the context of the companies, not basing the findings and interpretations only on the focused interviews.

Internal validity is related to factors that researchers are unaware of or cannot control regarding their effect on the variables under investigation (Runeson et al., 2012). The main internal validity threats related to this paper are the triangulation of the results from independent studies, and the fact that for Study A and B we used the scientific publications (Cruzes et al., 2018) and (Bernsmed and Jaatun, 2019) to derive the findings, meaning that we did not go back to the primary source of information (the original interview transcripts), although we had been involved in all the studies that were reported. Investigator bias was mitigated by involving three researchers during the design of the interview and workshop guides (investigator triangulation).

Construct validity reflects how well the measures used actually represent the constructs the study intends to measure (Runeson et al., 2012). The main threat to construct validity in our investigation is that we used different methods to investigate the phenomena. This strengthens the results we have because it shows the phenomena from different perspectives. For some aspects we could show that independent of the method, we always got to the same conclusions.

External validity is concerned with the generalisation of the findings (Runeson et al., 2012). The main findings are strongly bound by the context of the selected cases. To mitigate this threat, we conducted four studies, three in companies and one in an educational environment in which we could control some of the context in the studies. Furthermore, our main contribution lies in the cross-study comparisons based on which we concluded about findings on threat modelling. As such, it shall be valid despite the limitations of the cases. One limitation in study C is the fact that the study is performed with students, and the validity of using students as subjects in a simplified example in a university setting has been discussed by Falessi et al. (2017), Salman et al. (2015) and Svahnberg et al. (2008). The consensus in these studies seems to be that students are reasonable proxies for young developers starting out in the industry; to mitigate this possible age bias, we have triangulated the results with other studies and focused on the evaluation of the MS-TMT. The main contributions of this paper may be of interest and applicable to researchers and practitioners that work in similar contexts. To allow the transferability of the findings of this work, we detailed the description of the investigated cases, within the limits imposed by the associated non-disclosure agreements.

7. Conclusion

Introducing software security activities in an agile development life cycle is not an easy task. The results presented in this paper contribute to the body of knowledge in how threat modelling activities are being applied in the agile context and what can be done to ease the process. More specifically, we have investigated and documented a number of findings, challenges and best practices related to the use of Data Flow Diagrams, STRIDE and the Microsoft Threat Modelling Tool. The approach taken in our studies is to observe, interview and survey, in order to gain a solid understanding of how agile software development teams organise their daily work, so that we can base our recommendations on these. Study A is based on observations and document analysis from five teams in a single organisation, Study B is based on interviews with eight individuals from four different organisations, Study C is based on a questionnaire survey of 45 students at two different universities, and Study D is based on interviews with seven teams in a single organisation, supplemented with document analysis. Based on the results, we bring forward a list of recommendations that can support companies to improve their threat modelling strategies, as well as some implications for future research on this topic. Further, we have proposed three approaches to make the results from threat modelling more useful to agile teams. The results from our studies show that, while most of the concepts of threat modelling have already been by adopted by teams participating in our studies, they struggle with the practical aspects of transforming these concepts into practice.

Finally, we put forward the following future research directions:

- More empirical studies investigating how to integrate threat modelling in agile projects.
- Further development of tools for threat modelling that incorporate the trends in technology that the software products are being developed upon.
- Further development of templates for better drawings of DFDs.
- Studies that evaluate the effectiveness of threat modelling strategies for agile development, improving the relevance and the impact of the threat modelling activities for security.

CRediT authorship contribution statement

Karin Bernsmed: Conceptualization, Methodology, Validation, Investigation, Formal analysis, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization.

Daniela Soares Cruzes: Conceptualization, Methodology, Validation, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Project administration, Funding acquisition. **Martin Gilje Jaatun:** Conceptualization, Methodology, Validation, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing. **Monica Iovan:** Validation, Investigation, Resources.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The author Monica Iovan is employed by Visma, which is one of the studied organisations, but this has not influenced the results in any way.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jss.2021.111090.

References

Behutiye, W., Karhapää, P., López, L., Burgués, X., Martínez-Fernández, S., Vollmer, A.M., Rodríguez, P., Franch, X., Oivo, M., 2020. Management of quality requirements in agile and rapid software development: A systematic mapping study. Inf. Softw. Technol. 123, 106225. http://dx.doi.org/10.1016/j.infsof.2019.106225, URL http://www.sciencedirect.com/science/article/pii/S095058491930240X.

Bernsmed, K., Jaatun, M.G., 2019. Threat modelling and agile software development: Identified practice in four Norwegian organisations. In: 2019 International Conference on Cyber Security and Protection of Digital Services. Cyber Security. pp. 1–8.

Boström, G., Wäyrynen, J., Bodén, M., Beznosov, K., Kruchten, P., 2006. Extending XP practices to support security requirements engineering. In: Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems. ACM, pp. 11–18.

Bygdås, E., Jaatun, L.A., Antonsen, S.B., Ringen, A., Eiring, E., 2021. Evaluating threat modeling tools: Microsoft TMT versus OWASP threat dragon. In: Cyber Science 2021 – CyberSA for Trustworthy and Transparent Artificial Intelligence (AI). pp. 195–201.

Cassell, C., Symon, G., 2004. Essential Guide To Qualitative Methods in Organizational Research. Sage Publications Limited.

Chuttur, M., 2009. Overview of the technology acceptance model: Origins, developments and future directions. In: Sprouts: Working Papers on Information Systems, Vol. 9.

Creswell, J.W., 2014. Research Design: Qualitative, Quantitative and Mixed Methods Approaches, fourth ed. Sage Publications Limited.

Cruzes, D.S., Jaatun, M.G., Bernsmed, K., Tøndel, I.A., 2018. Challenges and experiences with applying Microsoft threat modeling in agile development projects. In: 2018 25th Australasian Software Engineering Conference. ASWEC. pp. 111–120.

Dodson, D., Souppaya, M., Scarfone, K., 2020. Mitigating the risk of software vulnerabilities by adopting a secure software development framework (SSDF). http://dx.doi.org/10.6028/NIST.CSWP.04232020, URL https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04232020.pdf. NIST whitepaper.

Easterbrook, S., Singer, J., Storey, M.-A., Damian, D., 2008. Selecting empirical methods for software engineering research. In: Shull, F., Singer, J., Sjøberg, D.I.K. (Eds.), Guide To Advanced Empirical Software Engineering. Springer London, London, pp. 285–311. http://dx.doi.org/10.1007/978-1-84800-044-5_11.

Falessi, D., Juristo, N., Wohlin, C., Turhan, B., Münch, J., Jedlitschka, A., Oivo, M., 2017. Empirical software engineering experts on the use of students and professionals in experiments. Empir. Softw. Eng. http://dx.doi.org/10.1007/s10664-017-9523-3.

Heijden, A., Broasca, C., Serebrenik, A., 2018. An empirical perspective on security challenges in large-scale agile software development. In: Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 1–4. http://dx.doi.org/10.1145/3239235.3267426.

Howard, M., Lipner, S., 2006. The Security Development Lifecycle. Microsoft Press, Redmond, WA.

Jaatun, M.G., Bernsmed, K., Cruzes, D.S., Tøndel, I.A., 2019. Threat modeling in agile software development. In: Felderer, M., Scandariato, R. (Eds.), Exploring Security in Software Architecture and Design. IGI Global, pp. 1–14.

Jaatun, M.G., Tøndel, I.A., 2008. Covering your assets in software engineering. In: The Third International Conference on Availability, Reliability and Security. ARES 2008. Barcelona, Spain. pp. 1172–1179.

Jeffries, C., 2012. Threat modeling and agile development practices. URL https://technet.microsoft.com/en-us/security/hh855044.aspx.

McGraw, G., 2006. Software Security: Building Security in. Addison-Wesley.

Microsoft, 2017. Microsoft security development lifecycle threat modelling. URL https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling.

Myagmar, S., Lee, A.J., Yurcik, W., 2005. Threat modeling as a basis for security requirements. In: Symposium on Requirements Engineering for Information Security, Vol. 2005. SREIS, Citeseer, pp. 1–8.

Myers, M., Newman, M., 2007. The qualitative interview in IS research: Examining the craft. Inf. Organ. 17, 2–26. http://dx.doi.org/10.1016/j.infoandorg.2006.11.001.

Oueslati, H., Rahman, M.M., ben Othmane, L., 2015. Literature review of the challenges of developing secure software using the agile approach. In: 2015 10th International Conference on Availability, Reliability and Security. IEEE, pp. 540–547.

OWASP, 2017. OWASP top ten. URL https://owasp.org/www-project-top-ten/.

OWASP, 2020a. Threat modeling. URL https://owasp.org/www-community/Threat_Modeling.

OWASP, 2020b. Threat dragon. URL https://threatdragon.org/.

Runeson, P., Höst, M., Rainer, A., Regnell, B., 2012. Case Study Research in Software Engineering: Guidelines and Examples. John Wiley & Sons.

Salman, I., Misirli, A., Juristo, N., 2015. Are Students Representatives of Professionals in Software Engineering Experiments? In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Vol. 1. pp. 666–676.

Schneier, B., 1999. Attack trees. Dr. Dobb's J. 24 (12), 21–29.

Shostack, A., 2014. Threat Modeling: Designing for Security. Wiley.

Sion, L., Van Landuyt, D., Yskout, K., Joosen, W., 2018. SPARTA: Security privacy architecture through risk-driven threat assessment. In: 2018 IEEE International Conference on Software Architecture Companion. ICSA-C. pp. 89–92.

Sjøberg, D., Hannay, J., Hansen, O., Kampenes, V., Karahasanovic, A., Liborg, N.-K., Rekdal, A., 2005. A survey of controlled experiments in software engineering. Softw. Eng. IEEE Trans. 31, 733–753. http://dx.doi.org/10.1109/TSE.2005.97.

Svahnberg, M., Aurum, A., Wohlin, C., 2008. Using students as subjects - an empirical evaluation. In: Proceedings of the 2008 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. ESEM'08, pp. 288–290. http://dx.doi.org/10.1145/1414004.1414055.

Swiderski, F., Snyder, W., 2004. Threat Modeling. Microsoft Press, Redmond, WA.

Terpstra, E., Daneva, M., Wang, C., 2017. Agile practitioners' understanding of security requirements: Insights from a grounded theory analysis. In: 2017 IEEE 25th International Requirements Engineering Conference Workshops. REW. pp. 439–442.

Thomas, D.R., 2006. A general inductive approach for analyzing qualitative evaluation data. Am. J. Eval. 27 (2), 237–246.

Venkatesh, V., Davis, F.D., 2000. A theoretical extension of the technology acceptance model: Four longitudinal field studies. Manage. Sci. 46 (2), 186–204.

VERBI Software GmbH, 2021. MAXQDA All-in-one tool for qualitative data analysis & mixed methods. URL https://www.maxqda.com/.

Villamizar, H., Kalinowski, M., Viana, M., Méndez Fernández, D., 2018. A systematic mapping study on security in agile requirements engineering. In: 2018 44th Euromicro Conference on Software Engineering and Advanced Applications. SEAA, pp. 454–461. http://dx.doi.org/10.1109/SEAA.2018.00080.

Weir, C., Becker, I., Noble, J., Blair, L., Sasse, A., Rashid, A., 2019. Interventions for software security: Creating a lightweight program of assurance techniques for developers. In: 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice. ICSE-SEIP. pp. 41–50.

Dr. Karin Bernsmed is a Senior Research Scientist at SINTEF Digital. She has a Ph.D. from the Norwegian University of Science and Technology (NTNU), where she also holds an Adjunct Associate Professor position. Karin has a long experience in cyber security threat and risk assessment, requirements engineering and design of secure and robust ICT systems in a number of different domains, including aviation, maritime and the energy sector. She is a certified ISO/IEC 27001 Lead Implementer.

Dr. Daniela S. Cruzes is a Professor at the Norwegian University of Science and Technology (NTNU). Previously, she worked as a senior research scientist at SINTEF in Norway. She has also been a researcher fellow at the University of Maryland and Fraunhofer Center for Experimental Software Engineering-Maryland. Dr. Daniela Cruzes received her Ph.D. in experimental software engineering from the University of Campinas - UNICAMP in Brazil in 2007. Her research interests are empirical software engineering, research methods and theory development, synthesis of SE studies, software security, software testing and agile and DevOps.

Dr. Martin Gilje Jaatun is a Senior Scientist at SINTEF Digital and an Adjunct Professor at the University of Stavanger. He graduated from the Norwegian Institute of Technology (NTH) in 1992, and holds a Dr.Philos. degree from the University of Stavanger. Previous positions include scientist at the Norwegian Defence Research Establishment (FFI), and Senior Lecturer in information security at the Bodø Graduate School of Business. His research interests include software security, security in cloud computing, and security of critical information infrastructures. He is vice chairman of the Cloud Computing Association (cloudcom.org), vice chair of IEEE CS TCCLD, vice chair of IEEE CS STC Blockchain, and a Senior Member of the IEEE. He is also an IEEE Cybersecurity ambassador.

Dr. Monica Iovan is the Head of Security Development in Visma focusing on developing services as part of the security program. Since 2019, she is also a Research Security Engineer in the application security domain. From her long experience, she gained a wide combination of technical expertise within software development. She received her Ph.D. from the Politehnica University Timişoara in 2013. Her research interests are software security in agile, self-management and diffusion of innovations theories.