An Extensive Systematic Review on the Model-Driven Development of Secure Systems

Phu H. Nguyen[a,*], Max Kramer[b], Jacques Klein[a], Yves Le Traon[a]

[a] *Interdisciplinary Center for Security, Reliability and Trust (SnT)*
University of Luxembourg, 4 rue Alphonse Weicker, L-2721 Luxembourg
[b] *Karlsruhe Institute of Technology*
Am Fasanengarten 5, D-76131 Karlsruhe, Germany

Abstract

Context: Model-Driven Security (MDS) is as a specialised Model-Driven Engineering research area for supporting the development of secure systems. Over a decade of research on MDS has resulted in a large number of publications.

Objective: To provide a detailed analysis of the state of the art in MDS, a systematic literature review (SLR) is essential.

Method: We conducted an extensive SLR on MDS. Derived from our research questions, we designed a rigorous, extensive search and selection process to identify a set of primary MDS studies that is as complete as possible. Our three-pronged search process consists of automatic searching, manual searching, and snowballing. After discovering and considering more than thousand relevant papers, we identified, strictly selected, and reviewed 108 MDS publications.

Results: The results of our SLR show the overall status of the key artefacts of MDS, and the identified primary MDS studies. E.g. regarding security modelling artefact, we found that developing domain-specific languages plays a key role in many MDS approaches. The current limitations in each MDS artefact are pointed out and corresponding potential research directions are suggested. Moreover, we categorise the identified primary MDS studies into 5 significant MDS studies, and other emerging or less common MDS studies. Finally, some trend analyses of MDS research are given.

[*] Corresponding author
Email address: `phu.nguyen@acm.org` (Phu H. Nguyen)

Conclusion: Our results suggest the need for addressing multiple security concerns more systematically and simultaneously, for tool chains supporting the MDS development cycle, and for more empirical studies on the application of MDS methodologies. To the best of our knowledge, this SLR is the first in the field of Software Engineering that combines a snowballing strategy with database searching. This combination has delivered an extensive literature study on MDS.

Keywords: systematic review, model-driven security

1. Introduction

With more and more IT systems being developed and used, approaches for systematically engineering *secure* IT systems are becoming increasingly important. *Model-Driven Security* (MDS) emerged more than a decade ago as a special area of *Model-Driven Engineering* (MDE) for supporting the development of secure systems. MDE has been considered by some researchers as a solution to handle complex and evolving software systems [22]. It leverages *models* and *transformations* as main artefacts at every development stage. MDS specialises MDE by taking security requirements and functional requirements into account at every stage of the development process. By *modelling* and manipulating models, the level of abstraction is higher than code-level that brings several significant benefits, especially regarding security engineering. *First*, security concerns can be considered together with business logic and other quality requirements such as performance from the very beginning, and throughout the MDS development life cycle. *Second*, reasoning about systems at the model level, e.g. with model-based verification and validation methods, makes it possible to check security requirements and other requirements at early design stages. These methods can perform formal verification as well as security testing based on models. Moreover, models that abstract away from target platform details can increase cross-platform interoperability. *Third*, MDS can be more productive, and supposedly less error-prone than traditional development methods by leveraging automated *model-to-model transformations* (MMTs) and *model-to-text transformations* (MTTs, code generation).

For more than a decade since MDS first appeared, a considerable number of MDS publications has shown a great attention of the research community to this area. The MDS approaches vary greatly in many artefacts such as the security concerns addressed, the modeling techniques used, the model transformations techniques used, the targeted application domains, or the evaluation methods used. To provide a detailed state of the art in MDS, a full systematic literature review (SLR) is needed.

So far, a full SLR on MDS does not exist. Surveys on MDS approaches ([13, 71, 79, 121]) could provide in-depth analyses of some well-known MDS approaches, but do not summarize the complete research area systematically. [62] could be closer to our work, but has several limitations in terms of scope and methodology. E.g., it missed many important primary MDS approaches such as UMLsec [65], and aspect-oriented approaches. In contrast, our SLR is performed in both width and depth of MDS research that reveals an extensive set of primary MDS studies. Furthermore, our review provides a detailed overview on key artefacts of every MDS approach such as used modeling techniques, considered security concerns, employment of model transformations, verification or validation methods, and targeted application domains. Finally, we present trend analyses for MDS publications, and for the addressed security concerns and other key artefacts.

This paper is an extended and improved version of [101]. In the previous version, we reported the results of a SLR based on 80 MDS papers found from an automatic search and a rigorous selection process. In this extended version, we improved our set of primary MDS papers by conducting two more search strategies: manual search and snowballing. On the resulting set of 108 finally selected MDS papers, we performed more detailed analyses for key artefacts, primary MDS studies, and trend analyses for a period of more than a decade.

The main contributions of this paper are: 1) detailed and condensed results on key MDS artefacts of all identified primary MDS publications; 2) a diagnosis of limitations of current MDS approaches with suggestions for potential MDS research directions; 3) a classification of significant and emerging/less common MDS approaches; and 4) trend analyses.

The remainder of this paper is structured as follows. Section 2 provides some main background concepts and definitions that are used in this paper. The objective of this

SLR, its research questions, search strategy, and selection process are described in Section 3. In Section 4, we present our evaluation criteria and data extraction strategy. Section 5 shows the main results of our review. Threats to validity are discussed in Section 6. In Section 7, we position this work regarding related work. Section 8 concludes the paper by summarising the results, highlighting open issues, and giving some thoughts on future work.

2. Background Concepts and Definitions

2.1. Systematic Literature Review and Snowballing

SLR is a means for thoroughly answering a particular research question, or examining a particular research topic area, or phenomenon of interest, by systematically identifying, evaluating, and interpreting all available relevant research [77]. Well-known guidelines for conducting SLRs in software engineering were provided by Kitchenham [77] and Biolchini et al. [23]. All individual studies that are identified as relevant research contributing to a SLR are called *primary* studies [77]. In this paper, based on the numbers of publications and citations of *primary* MDS studies, we further classify them into *significant* MDS studies, and *less common* or *emerging* MDS studies.

In a SLR, it is crucial to transparently and correctly identify as many relevant research papers in the focus of the review as possible. The search strategy is key to the identification of primary studies and ultimately to the actual outcome of the review [128]. The guidelines by Kitchenham [77] for SLRs in software engineering suggest to start with a database search that is based on a search string and also called *automatic search* in this paper. They also recommend complementary searches, e.g. a *manual search* on journals and conferences proceedings, references lists, and publications lists of researchers in the field.

Both automatic search and manual search have limitations [128]: The former depends on the selection of databases, on database interfaces and their limitations, on the construction of search strings, and on the identification of synonyms. The latter depends on the selection of research outlets, e.g. journals or conferences, and cannot be exhaustive. Therefore Wohlin et al. [128] proposed the snowballing search strat-

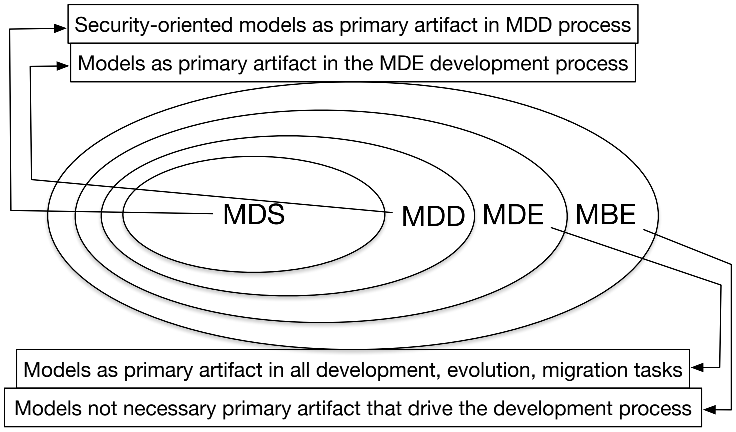

Security-oriented models as primary artifact in MDD process
Models as primary artifact in the MDE development process
MDS
MDD
MDE
MBE
Models as primary artifact in all development, evolution, migration tasks
Models not necessary primary artifact that drive the development process

Figure 1: Relations among MBE, MDE, MDD and MDS.

egy as a first step to systematic literature studies. The key actions of the snowballing search strategy are: 1) identify a starting set of primary papers; 2) identify further primary papers using the reference lists of each primary paper (backward snowballing); 3) identify further primary papers that cite the primary papers (forward snowballing); 4) repeat steps 2 and 3 until no new primary papers are found. We are convinced, that the snowballing search strategy complements the automatic and manual search strategies of Kitchenham [77]. In our SLR we defined and performed a snowballing search strategy that builds on the set of primary papers found in automatic and manual searches. Details of our search strategy are presented in Section 3.

2.2. A Definition of MDS

Numerous security engineering techniques exist which support the development of secure systems. There are also many MDE techniques for the development and maintenance of software systems in general. Our focus, however, is only on MDE approaches that are specifically customized for supporting the development of secure systems. As we already mentioned, MDS can be considered a subset of MDE. We will now clarify the relations between MDE, Model-Based Engineering (MBE), Model-Driven Development (MDD), security engineering, and MDS, which are important for our inclusion and exclusion criteria (Section 3.3). Regarding MBE, MDE, and MDD, we agree with the point of view presented by Brambilla et al. [31, p. 9]. Specifically, MBE can be used for development processes in which models may not necessarily be the central artifacts for development. E.g., if models are only used for documentation purposes and not in automated transformations. MDE can be seen as a subset of MBE in which models have to be the key artifacts throughout the development, i.e. models "drive" the process in every step. In other words, MDE is truly model-driven in every task of a complete software engineering process. This means that all development, evolution, and migration tasks have to be influenced by explicit models. Regarding MDE, model-to-model transformations (MMTs) or model-to-text transformations (MTTs) could be used by an MDE approach not only in development phase, but also in evolution or migration phases. MDD can be considered a subset of MDE that only denotes development activities with models as the primary artifact. Normally, MMTs and MTTs are used in MDD to obtain other models or to generate code in development activities. The core part of a MDD process includes modeling/designing phase which could lead to code generation phase. Other activities such as requirement engineering, testing might be also included. Regarding MDS, security-oriented models is a key artifacts. MMTs and MTTs could be used to manipulate security-oriented models in the MDS activities. Thus, MDS refers to all research approaches that focus on a MDD process for building secure systems. Figure 1 depicts these subset relations.

3. Our systematic review method

Our SLR method follows the guidelines of Kitchenham [77], and uses a variant of the snowballing strategy of Wohlin et al. [128]. We presented the motivation for our review in Section 1 and state our research questions in the next section. Based on these research questions, we developed a review protocol, which was evaluated before conducting the review. Figure 2 shows an overview of our SLR process. We combined an automated database search (Section 3.2.2), a manual search in relevant journals and conference proceedings (Section 3.2.3), and a snowballing strategy (Section 3.2.4) to identify as many primary MDS papers as possible. For our predefined protocol we clarify the selection criteria (Section 3.3) to reduce a possible bias in the selection process (Section 3.4). The quality assessment, data extraction and synthesis of the primary

MDS studies are based on a fixed set of evaluation criteria (Section 4). The results obtained from classifying, synthesising, analysing, and comparing the data extracted from the primary MDS studies are presented in Section 5.

3.1. Research Questions

This SLR aims to answer the following research questions:

RQ1: How do existing **MDS** ***approaches support the development of secure systems?***

This question is further divided into the following subquestions:

RQ1.1: What is the statistic of *security concerns* addressed by the MDS approaches?

RQ1.2: How do the MDS approaches *specify* or *model* security requirements together with functional requirements? Is there any tool that supports the *modelling* process?

RQ1.3: How are *model-to-model transformations* (MMTs) used and which MMT engines are used? Is there any tool support for the transformation process?

RQ1.4: How are *model-to-text transformations* (MTTs) used to generate code, including security infrastructure and configuration? Which tools are used for the generation process?

RQ1.5: Which *methods* were used to evaluate the approaches? What results have been obtained?

RQ1.6: Which *application domains* are addressed by the MDS approaches?

RQ2: What are current limitations of existing **MDS** ***research?***

RQ3: What are open issues to be further investigated?

3.2. Search Strategy

We developed a hybrid strategy to exhaustively search for MDS papers. The goal was not to miss any relevant MDS paper and therefore to find as many primary MDS papers as possible. Our hybrid strategy consists of three parts: automatic search (Section 3.2.2), manual search (Section 3.2.3), and snowballing (Section 3.2.4). In each step, we applied inclusion and exclusion criteria (Section 3.3) to select primary MDS studies.

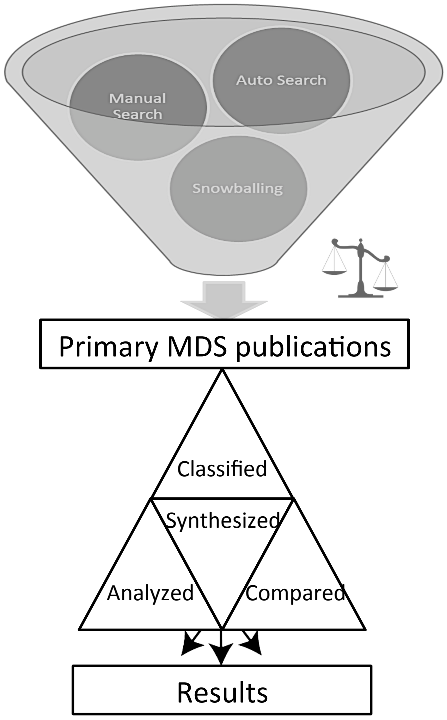


Figure 2: An overview of our SLR process.

3.2.1. Identification of a Search String

Based on the research questions (Sect. 3.1), we created search terms to form search strings, e.g. *model-driven*, *model-based*, *security*. We divided our search terms into three categories: MDE (model-driven, model-based, model*, MDA, UML), modeling (specify*, design*), transformations (transform*, code generation) and security.

To form the search string, we used a conjunction that combines disjunctions of the keywords of each term group. We had to refine our search string several times to make sure that as many potential relevant papers as possible are reached and had to adapt it according to the required format of the search engines. Some initial keywords were too specific and therefore the initial search result did not contain all the popular MDS papers that we used to assess the quality of search results. Other keywords were too general and resulted in many false positives. Our final set of keywords could have been more specific but our goals was to identify as many primary MDS papers as possible so that we preferred having too much paper candidates over having too few.

3.2.2. Step 1: Automatic Search in Databases for Scientific Literature

Using the search string described earlier, we performed automatic search within five electronic databases for publications between 2000 and 2014: IEEE Xplore[1], ACM Digital Library[1], Web of Knowledge (ISI)[1], ScienceDirect (Elsevier)[2], and SpringerLink (MetaPress)[2]. We did not use Google Scholar to identify paper candidates as it also lists unpublished work and drafts that differ from published versions of an article.

3.2.3. Step 2: Manual Search in Conferences Proceedings and Journals

To ensure the correctness and completeness of our review, we also conducted two manual searches: a manual search in potentially relevant peer-reviewed journals, and another one in potentially related conference proceedings. We selected journals and conferences that are highly ranked either in the domain of software engineering (SE) or security and privacy (S&P). We manually searched for all published papers from 2001 to 2014 in 10 journals and 10 conference proceedings as shown in Table 1 and 2.

[1] ieeexplore.ieee.org, dl.acm.org, apps.webofknowledge.com

[2] sciencedirect.com, link.springer.com

Table 1: Journals used in our manual search.

Acronym	Full Name	Field	Rating
TSE	IEEE Transactions on Software Engineering	SE	56
JSS	Journal of Systems and Software	SE	34
IEEE S&P	IEEE Security & Privacy	S&P	31
TISSEC	ACM Transactions on Information and System Security	S&P	29
TDSC	IEEE Transactions on Dependable and Secure Computing	S&P	28
COMPSEC	Computers & Security	S&P	27
INFSOF	Information & Software Technology	SE	27
SOSYM	Software and System Modeling	SE	27
TOSEM	ACM Transactions on Software Engineering and Methodology	SE	25
ESE	Empirical Software Engineering	SE	20

The 10 journals are chosen based on the relevance, the high impact index (Journal Citation Reports 2011), and the field ranking in the last 10 years according to the Microsoft Research website. 6 journals from SE and 4 journals from S & P were selected. We added the Empirical Software Engineering journal in order to find empirical validations of MDS approaches. The 10 conferences are also chosen on the relevance, and the conferences field ranking in the last 10 years according to the Microsoft Research website.

3.2.4. Step 3: Snowballing for a complete set of primary MDS *papers*

The automatic search and manual search processes yielded a set 95 primary MDS papers. To make sure that our final set of MDS papers is complete we adopted the snowballing strategy presented by Wohlin et al. [128]. We use the big set of primary MDS papers provided by automatic and manual searches as input for our snowballing strategy as follows.

Figure 3 shows how we formed the input set of MDS papers for snowballing. After conducting the automated search and applying the primary study selection procedures,

Table 2: Conference proceedings used in our manual search.

Acronym	Full Name	Field	Rating
ICSE	International Conference on Software Engineering	SE	60
CCS	ACM Conference on Computer and Communications Security	S&P	54
S&P	IEEE Symposium on Security and Privacy	S&P	49
USENIX	USENIX Security Symposium	S&P	39
AOSD	Modularity/Aspect-Oriented Software Development	SE	37
NDSS	Network and Distributed System Security Symposium	S&P	35
ACSAC	Annual Computer Security Applications Conference	S&P	29
SACMAT	Symposium on Access Control Models and Technologies	S&P	28
ESORICS	European Symposium on Research in Computer Security	S&P	24
MODELS	Model Driven Engineering Languages and Systems	SE	21

we obtained a first set of 80 MDS papers (Step 1). Similarly, after conducting the manual search and applying the primary study selection procedures, we obtained a second set of 29 MDS papers (Step 2). We merged these two sets in order to form a set of selected MDS papers that was used for partially conducting our snowballing strategy. Jalali et al. [61] provided a comparison between the SLR method and the snowballing method. They state that the snowballing method can be used to complement the automated search and manual search in terms of closing the final set of primary MDS papers. Because we already performed the automatic and manual searches for obtaining a set of 95 primary MDS papers, we only adopted the following 3 out of 5 steps of the snowballing strategy:

1. *Backward snowballing*: identify further potential primary MDS papers in the reference lists of the current primary MDS papers. Initially this is the set of papers found by the automated search and manual search.
2. *Forward snowballing*: identify further potential primary MDS papers by searching for papers that cite a current primary MDS papers. We used Google Schol-

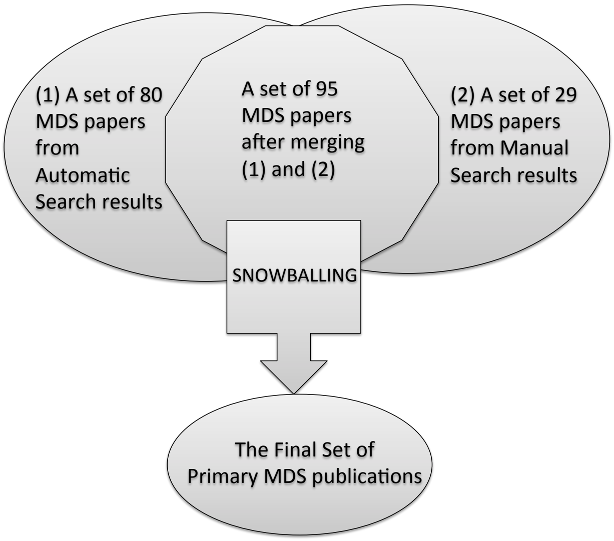

(1) A set of 80 MDS papers from Automatic Search results
A set of 95 MDS papers after merging (1) and (2)
(2) A set of 29 MDS papers from Manual Search results
SNOWBALLING
The Final Set of Primary MDS publications

Figure 3: Snowballing after Automatic Search & Manual Search.

aras recommended [128], because it captures more than individual databases.

3. If no new papers are found by repeating steps 1 & 2, then identify further primary MDS papers by searching publications lists on personal homepages or author pages of database and institutions for the primary authors of the identified primary MDS approaches. This step was performed to ensure that the most recent publications on the same or similar topics are included. If additional papers are identified then go back to Step 1.

Once no additional papers were found in step 3, we closed the cycle of identified primary MDS papers for data extraction, synthesis, and evaluation.

3.3. Inclusion and Exclusion Criteria

We already discuss our definition of MDS to give a better idea how we consider a paper as an MDS paper in Section 2. Here, we show in detail the inclusion and exclusion criteria that have been used in our primary MDS studies selection process.

MDS approaches for developing secure system vary a great deal as different security concerns can be addressed and different model-driven techniques can be used.

Therefore, it was absolutely necessary to define thorough inclusion and exclusion criteria to select the primary studies for answering our research questions:

1. Papers not written in English were excluded and already filtered out in our search process.

2. Papers with less than 5 pages in IEEE *double-column format or less than 7 pages in* LNCS *single-column format were excluded.*

3. Papers not concerned with MDE *were excluded. For example, papers addressing security problems without using* MDE *techniques were excluded.*

4. Papers proposing model-driven approaches without a focus on security concerns were excluded. E.g., model-driven approaches for performance analysis were excluded.

5. When a single approach is presented in more than one paper describing different parts of the approach, we included all these papers, but still considerd them as a single approach.

6. When more than one paper described the same or similar approaches, we only included the one with the most complete description of the approach. E.g., an extended paper [103] published in a journal will be selected instead of its shorter version [102] published in a conference proceeding.

7. Papers with insufficient technical information regarding their approaches were excluded. E.g., papers that neither provide a detailed description of secure models, nor a precise security notion, nor transformation techniques, were considered incomplete and were excluded.

8. Only papers with a MDD *perspectiove, i.e.* MDE *papers in which models are central artifacts throughout the development phase, were selected. Papers using model-based techniques only for verifying or analyzing security mechanisms without a link to the implementation code were excluded.*

9. Papers published n years ago with currently less than $2n - 2$ *citations as reported by Google Scholar were excluded.*

With these 9 clearly defined inclusion and exclusion criteria, we were able to perform the selection process in a more transparent and less biased way.

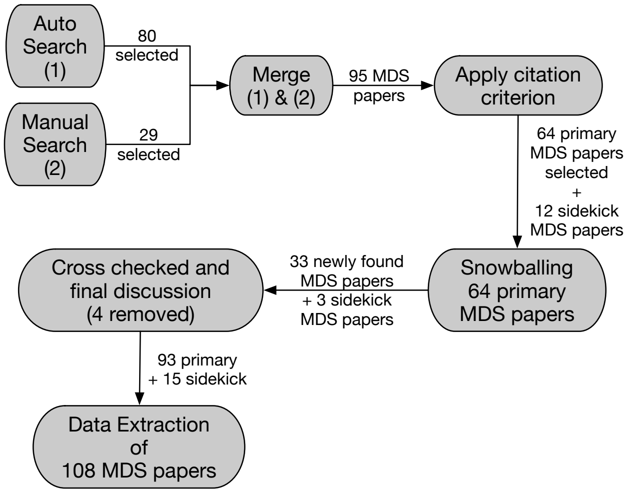

Auto Search (1)
80 selected
Manual Search (2)
29 selected
Merge (1) & (2)
95 MDS papers
Apply citation criterion
64 primary MDS papers selected + 12 sidekick MDS papers
Snowballing 64 primary MDS papers
33 newly found MDS papers + 3 sidekick MDS papers
Cross checked and final discussion (4 removed)
93 primary + 15 sidekick
Data Extraction of 108 MDS papers

Figure 4: The Selection Process with all the steps

3.4. Primary Studies Selection and Its Results

Here we present the selection process conducted while performing each search step in the three-pronged search process and its results. Figure 4 shows details of our whole selection process with all the numbers of MDS papers selected in each step.

3.4.1. Selection Process in the Automatic Search Step

Table 3 shows the results of our automatic search that is explained as follows. The papers found from the repositories described in Section 3.2.2 were divided among reviewers. For each paper, we first read the paper's title, keywords, and the venue where the paper was published to see whether it is relevant to our research topic. If the title and keywords of a paper were insufficient for deciding whether to include or exclude it, we further checked the paper's abstract. If the abstract of the paper were insufficient for deciding whether to include or exclude it, we further skimmed (and scanned if necessary) the paper's full text. Once each reviewer had done selecting candidate papers from his repositories, all the candidate papers from different repositories were merged to remove duplicates. We kept track of this merging process to see which duplicates were found. Duplicated papers were directly included in the final set of selected papers. All other candidate papers, were discussed by at least two reviewers. Some border-line

Table 3: Summary of the selection process based on Automatic Search

Source	IEEE	ACM	ISI	SD	SL	**Total**
Search results	2997	1506	3299	828	2003	**10633**
After reviewing titles/keywords	109	90	91	24	81	**395**
After reading abstracts	78	44	35	19	61	**237**
After skimming/scanning	31	21	17	15	20	**104**
After final discussion						**93**
Finally selected						**80**

papers were checked by all reviewers. We maintained a list of rejected candidate papers, with reasons for the rejection, after discussion among reviewers. In the end, 80 MDS papers were selected.

3.4.2. Selection Process in the Manual Search Step

29 candidate MDS papers were found in the manual search step. By merging with the set of 80 papers above, we obtained in total 95 MDS papers.

3.4.3. Selection Process in the Snowballing Step

After the first two steps, we conducted the snowballing as described in Section 3.2.4. However, once obtaining all the numbers of citations of every paper in the set of 95 MDS papers above, we found out that some papers are much less cited than others, or even having no citation at all. We argue that the papers without a minimum number of citations after getting published for a specific period could be considered as not significant in terms of research impact and continuation. On the other hand, we also were not too strict on this aspect. Specifically, we decided that papers that were published n years ago with the number of Google Scholar citations[3] less than $2n - 2$ citations are excluded. Thus, the selection criterion 9 about number of Google Scholar citations was added. This means we leave out the papers that do not have a minimum

[3] The citations of these 95 MDS papers were dated on May 19, 2014

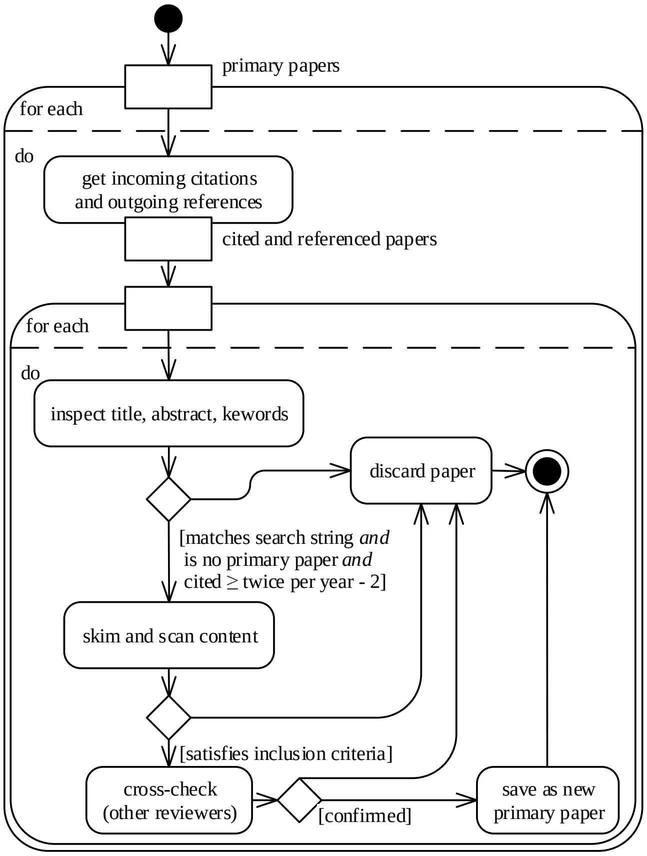


Figure 5: Our selection process while snowballing

impact. The subtraction accounts for the first two years of a publication, for which we allow zero citation, as a paper may be cited less often in the first years regardless of its quality or later impact. Of course, this also means the recent MDS papers published in 2013 and 2014 are not excluded by this citation criterion.

In 95 MDS papers, 31 papers were removed according to this citation criterion. Consequently, we used 64 primary MDS papers as the input for our snowballing process. In the snowballing step, we also apply the citation criterion[4] together with other criteria to select primary MDS papers. Details of our selection process while snowballing are shown in Figure 5. It is also important to note that every MDS candidate paper is cross-checked by three reviewers before any inclusion or exclusion decision. After all three steps, we have ended up with 93 primary MDS papers. However, we realised that some MDS papers, which were removed because of the citation criterion, should be put back in the final set as “sidekick” MDS papers. The main reason is that those MDS papers contain extra details of the approaches presented in the selected primary MDS papers. A “sidekick” MDS paper is a true MDS paper that was only excluded because of the citation criterion. Every “sidekick” MDS paper is part of a primary MDS approach. If they were removed, some important properties of the relevant primary MDS approaches could be missing in the data analysis. E.g., a paper presents an empirical study of a primary MDS approach. We would miss that empirical study of the primary MDS approach if the “sidekick” paper was removed because of the citation criterion. Thus, 15 “sidekick” MDS papers were put back in the final set. In the end, the final set of 108 MDS papers is used for data extraction and evaluation. We show the total numbers of citations per selected MDS studies for comparison in Section 5.

4. Evaluation criteria & Data extraction strategy

Classifications and taxonomies are important in any research domain, e.g. [39], [81]. In this section, we describe a set of key artefacts of MDS that forms a so-called

[4] The citations of MDS papers found in snowballing were dated on-the-fly.

evaluation taxonomy of MDS. We derived our evaluation taxonomy from our research questions. Moreover, our evaluation taxonomy are also based on the synthesis of evaluation criteria described in [73] and [71]. Having an evaluation taxonomy makes it more systematic to assess key artefacts of MDS as well as classify and compare different MDS approaches.

Our taxonomy of MDS classifies different dimensions that one has to take into account while leveraging MDE techniques for developing secure systems. The elements of our taxonomy are described as follows. For each element, the data extraction strategy is described to show how we extracted data from the primary studies to answer our research questions.

Security concerns: In this dimension, we classify primary studies according to the security concerns/mechanisms that the MDS approaches are dealing with. The range of security concerns is broad, e.g. authorisation, authenticity, availability, confidentiality, integrity, etc. We will count the number of papers addressing each security concern. Thus, security topic areas that addressed by the MDS approaches are measured quantitatively.

Modelling approaches: Security concerns can be modelled separately or not from the business logic, and by using different modelling techniques/languages. Primary studies can be classified by the paradigms of modelling, i.e. *Aspect-Oriented Modelling* (AOM) or non-AOM. In AOM approaches, security concerns are modelled in separate *aspect models* to be eventually woven (integrated) into the *primary model(s)*. Using AOM, security concerns can be modelled separately, modularly in design units (aspects) [113]. Vice versa, in non-AOM approaches, security concerns are not modelled as AOM aspects. That means security concerns can be modelled together with business logic in every place where they are needed. But, we also classify as non-AOM approaches where security concerns modelled separately (*separation of concerns*) from the business logic that can be integrated later into the system. E.g., a non-AOM approach could (separately) specify an access control policy using a *Domain-Specific*

Language (DSL)[5], and then transform and/or generate XACML[6] standard file for enforcing the access control policy. In other words, we would like to know the percentage of non-AOM approaches compared to the percentage of "full" AOM/*Aspect-Oriented Software Development* (AOSD) approaches. Separation of concerns can be considered as a key principle to cope with modern complex systems. Furthermore, approaches are also classified by the modelling languages, e.g. UML diagrams, UML profiles, or some kinds of DSLs, used to model security concerns and business logic. The outcome models are classified as of type standard or non-standard, and structural, behavioural, functional or other types. The granularity levels of outcome models are also reviewed.

Model-to-model transformations (MMTs) & tools: MMTs can take part in the key steps of the development process, e.g. for composing security models into business models and/or transforming *platform-independent models* (PIMs) to *platform-specific models* (PSMs). We extract data related to MMTs for answering the following questions: How well-defined are the MMTs rules? How MMTs are implemented? Using which MMT engines (e.g. ATL[7], QVT[8], Kermeta[9], Graph-based MMTs, etc.)? Is there any tool support for the transformation process? What is the automation level of MMTs: *automatic* (if entire process of creating the target model can be done automatically), *semi-automatic*, and *manual*. Some information about the classification of MMTs should also be extracted to see if it supports well for the security mechanisms? E.g., *endogenous* MMTs or *exogenous* MMTs used? Here, *endogenous* MMTs are transformations within one metamodel whilst *exogenous* MMTs are transformations between different metamodels.

Model-to-text transformations (MTTs, code and/or security infrastructure generation) & tools: MDE also supports the development of secure systems by automatically generating code, including (partial) complete, configured security infrastructures. Data should be extracted to see the main purposes of using code generation techniques.

[5] http://martinfowler.com/books/dsl.html

[6] *extensible Access Control Markup Language*, a XML-based declarative access control policy language

[7] http://www.eclipse.org/atl/

[8] http://projects.eclipse.org/projects/modeling.mmt

[9] www.kermeta.org

Is the whole system including security infrastructure generated? Or just the security infrastructure (configuration) is generated? Can fully code and/or security infrastructure be generated? Or just the (code) skeleton of the system is generated? Which tools are used for the code generation process?

Application domains: MDS approaches are also classified on the target application domains of the secure systems. Some MDS approaches might target only a specific application domain. Some might explicitly be applicable to different application domains in general. Others might implicitly be applicable to different application domains. Some examples of application domains are information systems, web applications, databases, secure smart-card systems, embedded systems, distributed systems, etc. The application domains might be overlapping but could show relatively the intended application domain(s) of a specific MDS approach.

Evaluation methods: To point out the limitations of each approach, we check again how the approach has been evaluated. How many case studies have been performed? What results have been obtained? What other evaluation methods (other than case studies) have been applied to evaluate these approaches? This can be answered by extracting data from the validation section of each paper.

To make the data extraction consistent among the reviewers, we all tried to extract the relevant data from a small set of prospective primary papers. We then discussed to ensure a common understanding of all the extracted data items and refined the data extraction procedure. Excel files were used for storing the extracted data while a tool called Mendeley[10] was used in reviewing and controlling the selected papers. The final set of primary studies (selected papers) was divided among reviewers. Each reviewer examined again the allocated papers and enriched the Excel files to ensure detailed data according to the taxonomy has been extracted from the selected papers. The data extraction forms of each reviewer were read and discussed by two other reviewers. All ambiguities were clarified by discussion among the reviewers.

To answer the last two research questions, we reviewed the range of security topics, the scope of MDS research work and the quality of MDS research results to determine

[10] `http://www.mendeley.com/`

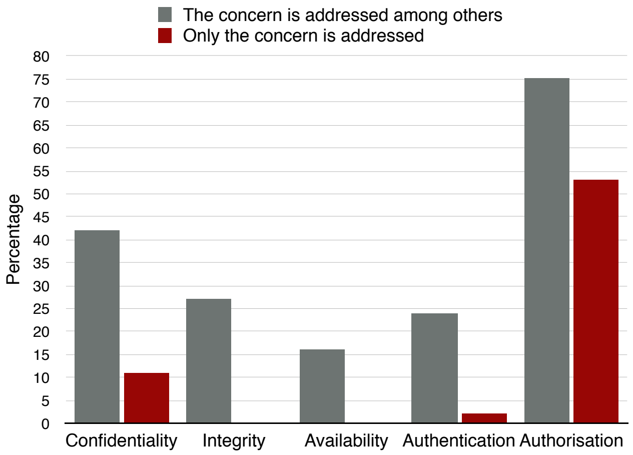


Figure 6: How much each concern is addressed in MDS?

whether there are any observable limitations and open issues.

5. Results

This section presents the main results of our SLR and how our research questions are answered. First, in Section 5.1 we report on some statistic results according to the evaluation criteria. Then, the significant MDS approaches and other emerging/less common MDS approaches are revealed and described in Sections 5.2, 5.3 respectively. Finally, Section 5.4 analyses the trends of some key factors in MDS.

5.1. Results per Evaluation Criterion

An overview of the results can be seen in Figures 6, 7, 8 and Tables 4, 5. Fig. 6 shows the statistics about how each security concern has been addressed by the primary MDS approaches. Fig. 8 visualises other key results for a representative set of evaluation criteria. Tables 4, 5 summarise all the values for all evaluation criteria. We present the results for each evaluation criterion as follows.

Security concerns/mechanisms: *RQ1.1: What is the statistic of* security concerns *addressed by the* MDS *approaches?* To answer this question, we analysed the data regarding the security concerns addressed by the reviewed MDS approaches. Fig. 6

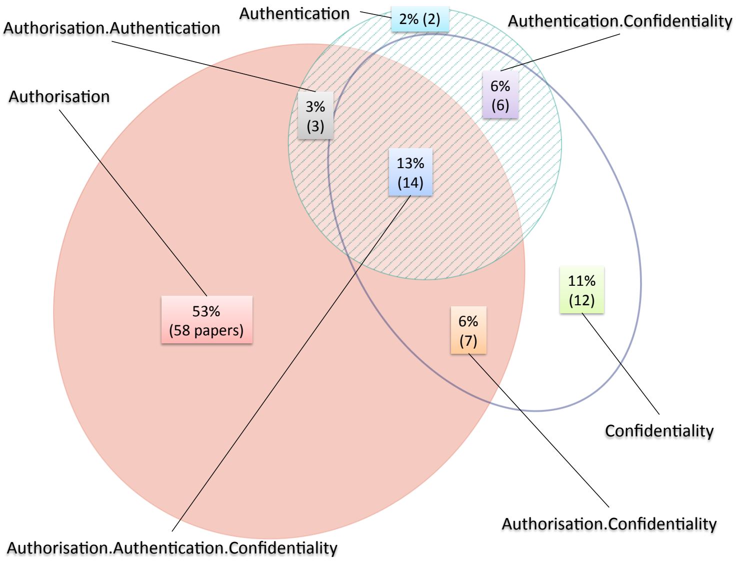

Authorisation.Authentication
Authentication
2% (2)
Authentication.Confidentiality
Authorisation
3%
(3)
6%
(6)
13%
(14)
11%
(12)
53%
(58 papers)
6%
(7)
Confidentiality
Authorisation.Confidentiality
Authorisation.Authentication.Confidentiality

Figure 7: Intersection of Authentication, Authorisation, and Confidentiality

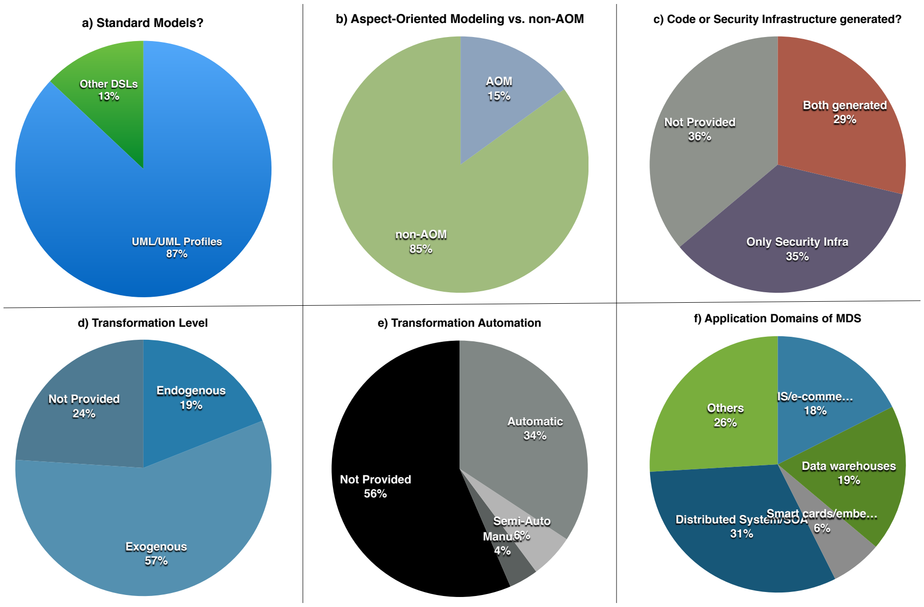


Figure 8: Statistics of some key MDS artefacts

shows the statistic of security concerns tackled by the reviewed MDS approaches. We can see that *authorisation* is addressed the most, by 75% of the examined MDS papers. Moreover, more than half of the MDS papers (53%) deal with *authorisation* only (Fig. 6). The second security concern in terms of receiving attention is *confidentiality* addressed by 42% of the examined MDS papers. 11% of the examined MDS papers tackle *confidentiality* solely (Fig. 6). Other security concerns, like *integrity*, *authentication*, and *availability* are, however, less tackled with 27%, 24%, and 16% correspondingly. These results suggest that more MDS research work should focus on particular security concerns like *integrity*, *availability*, and *authentication*.

We also would like to know how much multiple security concerns are tackled at the same time by the MDS approaches. Fig. 7 displays the statistic about how much three key security concerns (Authentication, Authorisation, and Confidentiality) are tackled solely and simultaneously. Only 13% of the examined MDS papers propose methodologies to tackle all three together. About 15% of the examined MDS papers deal with two concerns simultaneously: Authentication and Authorisation (3%), Authentication and Confidentiality (6%), Confidentiality and Authorisation (6%). Not only multiple security concerns are less tackled, but also rarely the inter-relations among multiple security concerns are formally taken into account in the reviewed MDS approaches. Future MDS approaches should address multiple security concerns simultaneously, systematically by formally specifying inter-security concern relations. The inter-relation among security concerns have to be taken into account while developing DSLs for specifying security requirements.

These first results are very interesting. Indeed, an open question is “why in MDS *authorisation* and *confidentiality* got more attention?”. A possible answer could be that MDS is a relatively young research area with more “model-driven” than “security”. MDS is the common name of the MDE approaches specifically focusing on secure systems development. Thus, among the authors of the published MDS papers, there are significantly more researchers with MDE background than security engineering background. Researchers that mainly work with MDE techniques may first address *authorisation* (e.g. AC) because it is closer to application logic and functional requirements than other security concerns. This could be linked to the nature of security concerns. MDE researcher might not be familiar with security concerns to be addressed at the network layer. Given the background of the authors of the most renowned MDS approaches, it might be that we need more interest in MDE from the security engineering community to see more MDS approaches dealing with security concerns like *integrity*, *availability*, and *authentication*. Therefore, we suggest that more effort should be put into communicating MDE techniques as well as MDS approaches to the security engineering community.

Modeling approaches: *RQ1.2: How do the* MDS *approaches* specify *or* model *security requirements together with functional requirements? Is there any tool that supports the* modelling *process?* Fig. 8a shows that 87% of the examined papers used standard UML models and defined DSLs for security concerns using the profile and stereotype mechanisms of the UML. 13% used other DSLs (e.g. [92], [83], or [93]). Thus, we understand that standardised, common UML models are broadly used by MDS approaches. On the other hand, defining DSLs (either UML profiles or other DSLs) is also very popular to leverage MDE techniques for secure systems development. UML profiles and other kinds of DSLs have been developed to better capture the specific semantics of security concerns. In other words, defining DSLs plays a key role in MDS because that way allows expressing security concepts/elements more easily. However, using UML profiles is not the only way for developing DSLs in MDS approaches. DSLs which are not UML profiles are also recommended, especially DSLs that can deal with multiple security concerns in the same system.

15% of the papers discuss approaches that are based on AOM (Fig. 8b) where security concerns are specified as aspects and eventually woven into primary models. Even though the remaining 85% are not really aspect-oriented, most of them still follow the *separation of concerns* principle and really separate security concerns from the main business logic [11]. In most of the cases, security concerns were specified separately from the business logic in PIMs and transformed into PSMs that can be refined into security infrastructures (e.g. *XACML*) integrated with the systems.

[11] Note that in this paper we only classified a modelling approach as AOM if a concern is modelled as an aspect model that can be woven into a primary model. We explained this point in Section 4.

Security concerns are often modelled and analysed with a DSL that is concern-specific. But, few MDS papers have well-defined semantics for their languages so that these languages can be used for formal analysis. Only some papers related to the UMLsec, SecureUML approaches (see Section 5.2) provide some formal basis for security analyses. This shows that further efforts are required to mature security-specific modelling languages to foster analyses. Most (89%) of the MDS papers use structural models. Behavioural models are used in 31% of the reviewed MDS papers. Other types of models like domain specific models accounted for 13%. Using solely one type of models could not be enough to be able to express multiple security concerns. Thus, very few modelling approaches propose to deal with multiple security concerns together like [50, 108]. Most of them are specific to address only one security concern solely.

Model-to-model transformations (MMTs) & tools: Table 5 shows that 74% of the papers clearly mentioned MMTs while 26% did not use or mentioned transformations, e.g., because of a manual integration of security. More specifically, 57% of the examined papers use exogenous transformations. Most of these were used to transform PIMs to PSMs (Fig. 8d). Security concerns were modelled using DSLs for each concern to obtain PIMs that were transformed into PSMs, which can be refined into code. 19% define endogenous MMTs that are used to weave/compose security models into base models defined using the same DSLs.

34% of the examined MDS papers implement automatic MMTs, 6% describe semi-automatic (interactive) MMTs, and only 4% are manual (Fig. 8e). But 56% do not specifically provide any implementation information about MMTs, e.g. some simply provide mapping rules for transforming models. Having automated MMTs is one of the key success factors of MDE [60] and MMTs play a crucial role in MDS as well. Especially some important semantics of security mechanisms might be embedded in the MMTs. Providing MMTs implementation details in MDS is important to evaluate the efficiency of each approach. It can be also helpful for other researchers to learn from previous experiences in choosing or developing a suitable transformation engine for their work. 19% of the selected MDS papers describe their MMTs implementation using standard transformation languages like ATL and QVT. 81% of the papers only

Table 4: Results classified by the evaluation criteria

Evaluation criteria		# papers	%
Security concerns (overlapping)	Confidentiality	45	**42**
	Integrity	29	**27**
	Availability	17	**16**
	Authenticity	26	**24**
	Authorisation	81	**75**
Aspect-Oriented Modeling/AOSD	Yes	16	**15**
	No	92	**85**
Standard models	Yes(UML/UML profiles)	94	**87**
	Other DSLs	14	**13**
Type of models (overlapping)	Structural	96	**89**
	Behavioural	33	**31**
	Others	14	**13**

describe the transformation rules without implementation details, or use other transformation languages like graph-based transformations, or specific (Java-based) compilers/tools.

Model-to-text transformations (MTTs) & tools: Table 5 shows that 64% of the papers describe MTTs or the generation of code or security infrastructures. 36% of the papers do not describe MTTs in details. Some mainly used models for verifying or analyzing implemented secure systems, e.g. UMLsec where code/security infrastructure generation is mainly mentioned in future work. Comparing the purposes of MTTs, we can see in Fig. 8c that there are nearly as many MDS papers (34%) that only generate security infrastructure, such as XACML or security aspects code, as the MDS papers that describe generation of both code and security infrastructure (29%).

The tools used for code generation are not shown in Table 5 because there are

Table 5: Results classified by the evaluation criteria

Evaluation criteria		# papers	%
Transformations used	Yes	80	**74**
	No/Unknown	28	**26**
Transformations level	Endogenous	20	**19**
	Exogenous	62	**57**
	Not Provided	26	**24**
Transformations automation	Automatic	37	**34**
	Semi-automatic	6	**6**
	Manual	4	**4**
	Not Provided	61	**56**
Standard Transformations	ATL/QVT	20	**19**
	Others/not mentioned	88	**81**
Code generation mentioned	Yes	69	**64**
	No	39	**36**
Code + Security Infrastructures generated	Yes	31	**29**
	Only Security Infrastructure	37	**34**
	Not Provided	40	**37**
Application Domains	IS/e-commerce	19	**18**
	Data warehouses	20	**19**
	Smart cards/ embedded systems	7	**6**
	Distributed Systems/SOA	34	**31**
	Others	28	**26**
Type of validation	Controlled experiment	2	**2**
	Industry case studies	5	**5**
	Academic case studies	72	**67**
	Example only	23	**21**
	Not Provided	6	**5**

too many different tools. Besides Eclipse-based MTT engines like Xpand[12], there are many cases where ad-hoc self-developed engines (e.g. Java-based tools, parsers, etc.) are used. A reason for that could be that many "ad-hoc" tools are preferred because of their specific support for a specific security domain. Ark [127][13], for example, transforms an input UML model designed with the proposed UML profile into a skeleton of application code (program code and deployment descriptor). More ad-hoc Java-based tools like the one in [32] generates code (XACML policy files) from the constraints specified in SECTET-PL The tool uses Antlr [104], a compiler program for the syntax analysis of the constraints.

In general, MMTs and MTTs are widely used in MDS to improve the productivity of the development process. Most of the primary MDS approaches do mention to leverage MMTs and/or MTTs by describing transformation rules/intentions. However, more than half of the primary MDS approaches did not provide implementation details of MMTs or MTTs. Not many primary MDS approaches use standard transformation languages/tools like ATL or QVT but rather ad-hoc tools like Java-based compiler/tools for engineering security into the system. With the progress in the maturity of standard MMT and MTT tools, they should be leveraged more in the future MDS approaches. Most of the MMTs in the selected studies are *exogenous* used for transforming PIMs to PSMs. The main reason is that there are many approaches (e.g. dealing with access control) generating only security infrastructure. Access control models (PIMs) often used to generate XACML configuration files (PSMs) for enforcing security policy. Another reason could be the lack of *all-round* approaches for the whole development cycle of secure systems which in the end lead to automatic generation of both code and security infrastructure. An *all-round* approach could follow AOM paradigm to fully leverage the automation of MMTs and MTTs for composing, transforming and generating both code and security infrastructure. Developing tool chains (based on MMTs and MTTs) to derive from models to implementation code is also an important piece

[12] `https://www.eclipse.org/modeling/m2t/?project=xpand`

[13] extends the code generation engine of the openArchitectureWare framework that was already migrated into Eclipse as Xpand

of future work. Few complete tool chains to automate (most of) the MDS development process have emerged, but are still rare.

Application domains: Fig. 8f shows the main application domains that have been secured by MDS approaches. In general, these are distributed systems or SOA (31%), information systems or e-commerce (18%), data warehouses (19%), and smart cards/embedded systems (6%). The remaining MDS papers do not clearly state a domain, or could be generically applicable for different application domains, such as [59, 74, 95, 108].

Evaluation methods: Most of the papers (67%) describe academic case studies used to evaluate their approaches. There are still quite many MDS papers (21%) which only provide “running examples” to illustrate their approaches. Few MDS papers show controlled experiments (2%) and industry case studies (5%) in the evaluation of their approaches. There are very few papers that provide an in-depth evaluation like [38], [118], and [21]. Therefore, we suggest that more effort should be put in evaluating MDS approaches, e.g., with empirical studies or benchmarks.

5.2. Significant MDS *Approaches*

Altogether, the synthesised data show that there are currently several MDS approaches that have been proposed, used, and discussed in multiple publications. We would like to identify the most influential MDS approaches in terms of numbers of publications and citations. In total, five primary MDS approaches, which are called significant MDS approaches, have been identified. They are summarised in Tables 6, 7. Each has at least 7 primary MDS papers in our final set. The details of each approach, except Secure data warehouses, can be found in [79]. Here we briefly present each approach, and then compare some key points among them.

SECTET firstly aimed at securing web services by leveraging the Object Constraint Language (OCL) for specifying RBAC [5]. Based on that, a complete configured security infrastructure (XACML policy files) is generated. Later on, the authors proposed a specification language namely SECTET-PL (OCL-based) which is part of the SECTET framework for model-driven security for B2B workflows. In this framework, Constraint-based RBAC (CRBAC) can be specified and then transformed into low-level web services standard artefacts (XACML policy files) [7]. SECTET-PL is also used for modelling restricted (RBAC-based) delegation of rights in Service Oriented Architecture [8]. Their modelling approach is extended in [53, 54]. MMT and MTT are both carried out in a complete model-driven framework [32, 34, 56]. In general, SECTET mainly addresses RBAC as its security concern and focuses on generating security infrastructure (XACML), not all the source code. Recently, Memon et al. [80] and also Katt et al. [72] propose two pattern refinement approaches based on SECTET framework that allows flexible configurations of SOA security.

Secure data warehouses (DWs) are the motivation for the work of developing MDS techniques for secure database development. This MDS approach is very specific for developing secure DWs. Fernández-Medina et al. [43, 44] extend OCL and UML for secure database development [45]. Their approach also uses UML profiles for modelling security enriched PIMs as inputs for a model-driven framework to create secure DW solutions [46, 115]. Secure PIMs can be transformed to secure PSMs by a set of formally defined QVT rules [116, 117, 118]. These PSMs can then be used for generating code with security properties. A similar MDS approach for developing secure XML data warehouses is presented in [122, 123, 124, 125] More recently, the above mentioned techniques for secure DW development are also leveraged in a reverse engineering style to modernise legacy DWs [27].

SecureMDD is proposed for facilitating the development of smart card applications based on UML models. In SecureMDD, UML class diagrams are used for modelling static aspects while UML sequence and activity diagrams are used for modelling dynamic aspects of a system [90]. From platform-independent UML models (PIMs) of a system, its formal abstract state machine (ASM) specification and Java Card code are generated. The generated abstract state machine specification is used for formally proving the correctness of the generated code regarding the security properties of the system. Thus, their MDS approach integrates MDE techniques with semi-formal and formal methods for verification as well as the implementation of security-critical applications [85, 86, 88]. The authors illustrated that SecureMDD is applicable for the development of large and complex secure Smart Card applications as well [87]. The main limitations of SecureMDD are its specific application domain and the lack of

Table 6: Summary of the Significant MDS Approaches

	Total Citations	Security Concerns	Modeling Approach				MMT		MTT		Verification	Application Domains	Validation
			Language	AOM	SOC	Type	Level	Impl	Both	Impl			
Sectet [5, 6, 7, 8, 9, 32, 33, 34, 52, 53, 54, 55, 56, 72, 80]	304	mainly authorisation (access control, delegation), integrity, confidentiality, non-repudiation,	UML profiles	X	✓	S	Exo	Qvt	X	Xpand	X	e-government, e-health, e-education, web services, SOA	ACS
SecureDWs [24, 25, 26, 27, 43, 44, 45, 46, 115, 118, 119, 120, 122, 123, 124, 125, 126]	336	privacy, integrity, authentication, availability, non-repudiation, auditing, access control	UML profiles	X	✓	S	Exo	QVT	X	MOF, CASE tool	X	web applications, databases	IE, ACS

Note: Supported (✓); Partially supported (o); Not supported (X); Controlled experiment (CE); Industrial case study (ICS); Academic case study (ACS); Illustrative example (IE); Not provided (NP); Non-restrictive (NR); Self-developed (Self-); Endogenous (Endo); Exogenous (Exo); Structural (S); Behavioural (B); Others (O)

All the total citations numbers are calculated based on the citations reported by Google Scholar when we were selecting primary MDS papers in 2014.

Table 7: Summary of the Significant MDS Approaches (continue)

	Total Citations	Security Concerns	Modeling Approach				MMT		MTT		Verification	Application Domains	Validation
			Language	AOM	SOC	Type	Level	Impl	Both	Impl			
SecureMDD [28, 85, 86, 87, 88, 89, 90]	72	cryptography (secrecy, integrity, confidentiality), application-specific security properties	UML profiles	X	✓	S, B	Endo, Exo	Qvt	✓	Xpand	Kiv theorem prover, test cases from UML specifications	smart card and service applications	IE, ACS, ICS
SecureUML [14, 15, 16, 17, 18, 19, 30, 35, 38, 40, 78]	1230	access control	UML profiles	o	✓	S	Endo	o	o	ArcStyler, ActionGUI, compiler (self-)	SecureMova model-checker	web applications	IE, ACS, ICS
UMLsec [48, 63, 64, 65, 66, 67, 68, 69, 97, 98]	778	confidentiality, integrity, authenticity, authorisation, freshness, information flow, non-repudiation, fair exchange	UML profiles	X	o	S, B	Endo ([48, 67])	X	X	compiler (self-)	aiCall theorem prover	web applications, embedded systems, distributed systems	IE, ACS, ICS

Note: Supported (✓); Partially supported (o); Not supported (X); Controlled experiment (CE); Industrial case study (ICS); Academic case study (ACS); Illustrative example (IE); Not provided (NP); Non-restrictive (NR); Self-developed (Self-); Endogenous (Endo); Exogenous (Exo); Structural (S); Behavioural (B); Others (O)

analysis for consistency between the UML models and the ASM model.

SecureUML is the approach which aims at bridging the gap between security modelling languages and design modelling languages. First, UML and UML profile are used for modelling application with role-based access control that can lead to generated complete access control infrastructures [78]. Then, Basin et al. [18] propose a UML-based language (UML profiles) with different dialects, which forms modelling languages (such as SecureUML + ComponentUML) for designing secure systems. Access control infrastructures for server-based applications can be generated automatically from models. Their work mainly focuses on access control constraints based on RBAC in design models. Semantics of SecureUML (and ComponentUML) are provided by Brucker et al. [35] and Basin et al. [14, 16] which enable formal analysis of security-design models. Based on this work, Clavel et al. show and discuss their practical experience of applying SecureUML in industrial settings [38]. Recently, the work on SecureUML has been continued by combining SecureUML + ComponentUML with a language for graphical user interfaces (GUI), namely ActionGUI [15, 19]. These modelling languages with MMT enable the full generation of security-aware GUIs from models for data-centric applications with access control policies. Another recent work by Dios et al. [40] makes use of ActionGUI for model driven development of a secure eHealth application. The main limitation of SecureUML is its sole focus on access control.

UMLsec is one of the most well-known UML-based approaches in MDS proposed early by Jürjens [65, 66]. Security requirements, threat scenarios, security concepts, security mechanisms, security primitives can be modeled by using security-related stereotypes (UML profiles), tags, goal trees. and security constraints. Thus, it is possible to formally analyse UMLsec diagrams against security requirements regarding their dynamic behaviours. Not like SecureUML only focusing on authorisation (e.g. access control), UMLsec addresses multiple security concerns such as *confidentiality*, *integrity* [64]. Not to a great extent but AOM is also used in the UMLsec approach [67]. Later on, UMLsec is deployed by Best et al. [21] in an industrial context for designing and analysing designs of distributed information systems. On the other hand, relevant tools support for UMLsec are presented in [68]. To tackle also social challenges in security, UMLsec was combined with Secure Tropos [96] to take on security from requirement engineering phase [98]. This work is then extended and applied to two different industrial case studies [97]. A more recent work related to UMLsec is by Jürjens et al. [69] for incremental security verification for evolving UMLsec models. However, UMLsec lacks support for improving productivity of the development process in terms of automated model transformations. Even having a view from models to code but the lack of automated transformation(s) from models to implementation code is a miss in UMLsec. Other than that, UMLsec could be considered as the most complete and mature MDS approach that deals with multiple security concerns, from very early at the requirement engineering level, with transformations, formal analysis possibility, tools support, industrial case studies.

In general, the most common point among the significant MDS approaches is that they all propose to use UML profiles in their modeling phase. Even though not following truly AOM, defining UML profiles as DSLs for modeling security concerns still allows these significant MDS approaches to have separation of concerns. Except SecureUML which only addresses access control, other approaches are able to touch multiple security concerns. Structural models are mainly used in all five approaches. SecureMDD and UMLsec have also used behavioral models. Exogenous MMTs are defined in SECTET and SecureDWs to transform PIMs (UML models) to PSMs. SecureUML and UMLsec integrate security into systems specified in UML using endogenous MMTs. SecureMDD combine both kinds of MMTs in their development process. Some standard transformation tools are used (e.g. QVT and XPAND) among other self-developed tools (java-based compilers). With their formal background, SecureMDD, SecureUML and UMLsec provide some tools for formal verification of security properties. These three also have industrial case studies while SECTET and SecureDWs have not. Generally, each approach is quite specific to a application domain, e.g. SecureDWs for secure database development, or SecureMDD for secure smart card development.

5.3. Less common/emerging MDS *Approaches*

It would not be fair to only discuss about the above-mentioned significant MDS approaches. There are other less common or emerging MDS approaches that are also worth to get noticed and analysed. We discuss some representative ones here. For the full list, readers are referred to Tables 9 and 11. The less common or emerging MDS approaches here are simply classified into several groups as follows.

Pattern-based MDS: Based on domain-independent, time-proven security knowledge and expertise, security patterns can guide security at each stage of the development process. Some MDS approaches that leverage security patterns are remarkable. Abramov et al. [1, 2, 3] propose an MDS framework for integrating access control policies into database development. At the pre-development stage, organisational policies are specified as security patterns. Then, the specified security patterns guide the definition and implementation of the security requirements which are defined as part of the data model. The database code can be generated automatically after the correct implementation of the security patterns has been verified at the design stage. Their approach has been evaluated in a controlled experiment [2]. Also using security patterns but at a different level of abstraction, Kim et al. [74, 75] develop a pattern-based technique for systematic, model-driven development of secure systems focusing on access control. Because this work mainly focuses on the design stage, access control is specified as design pattern. Bouaziz et al. [29] introduce a security pattern integration process for component-based models. With this process, security patterns can be integrated in the whole development process, from UML component modelling until aspect code generation. Another pattern-driven approach is proposed by Schnjakin et al. [112] for facilitating the configuration of security modules for service-based systems. The proposed security advisor enables the transformation from the general security goals, via security patterns at different abstraction level, to concrete security configurations. Menzel [82] uses the security configuration patterns to operate the transformation of architecture models annotated with security intentions to security policies. The patterns that provide expert knowledge on Web Service security can be specified using a DSL. As using cloud services provided by cloud providers is getting more popular, Moral-García et al. [91] recently propose an enterprise security pattern for securing Software as a Service.

The security solution provided by the pattern can be driven by making design decisions whilst performing the transformation between the solution models. Specifically, from a Computation Independent Model (CIM), different PIMs can be derived based on different design decisions with security patterns. Those PIMs are transformed into PSMs which are then transformed into Product Dependent Models (PDMS).

MDS for SECURITY@RUNTIME: Many modern applications such as cloud-based software-as-a-service (SaaS) applications require the dynamic adaptation or even evolution of both security and service at runtime. More and more (MDS) approaches have been being proposed in this area. Almorsy et al. [12] introduce an approach called Model Driven Security Engineering at Runtime (MDSE@R). MDSE@R is based on a UML profile with tool supports for separately specifying base system and security, and then merging those models into a joint system-security model. Because security and system models are separated and loosely coupled, they can evolve more easily. Security controls are enforced dynamically into the target system at the code level. After that, in [11] the same authors leverage the MDSE@R approach for multi-tenant, cloud-hosted SaaS applications. This allows dynamically engineering security for multi-tenant SaaS applications at runtime. Recently, Almorsy et al. [10] develop a new DSL called SECDVSL for specifying visually a variety of security concepts like objectives, threats, requirement, architecture, and enforcement controls. SECDVSL also allows maintaining traceability among these security concepts. Not specifically for SaaS applications but component-based architecture, Morin et al. [92] leverage the notion of model@run.time to enable dynamically enforcing role-based access control policies into component-based systems. In the follow-up work, Nguyen et al. [103] deal with not only access control policies but also the more complex, but essential, delegation of rights mechanism. The propose MDS framework allows dynamically enforcing/weaving access control policies with various delegation features into security-critical systems. This is done with a flexibly dynamic adaptation strategy. Another runtime-update of security policy-based approach is presented by Elrakaiby et al. [42]. The introduced DSL called *Security@Runtime* covers many of the security requirements of modern applications such as authorisation, obligation, and reaction policies. Xiao [130]'s work is on adaptive and secure multi-agent systems. The authors adopting the adaptive agent model to put forward a security-aware model-driven mechanism by using an extension of RBAC model.

MDS for Secure SOA: Many MDS approaches focus on securing service-oriented systems (SOSs). Gilmore et al. [51] show how services, service compositions, and non-functional properties can be modeled using their self-developed UML profile and its extension. They address non-functional properties in general where security is considered with performance and reliable messaging. The models are the input for the framework VIATRA[14] to derive deployment mechanisms using MMT and MTT. Wada et al. [127] also address non-functional aspects in SOA with a MDD framework and tool support. Their work is empirically evaluated to show the improvement in the reusability and maintainability of service-oriented applications. More specifically to integrate security-related non-functional aspects in the development of services, Gallino et al. [49] present their MDS solution using multiple domain-specific models independently addressing security aspects. Hoisl et al. [57, 58] propose an MDS approach based on SoaML for specification and the enforcement of secure object flows in process-driven SOA. [83, 84] introduce a security metamodel for SOA. This metamodel is the base for their MDS framework that allows modelling of security requirements in system design models. Going further than modelling, Nakamura et al. [99] propose an MDS tooling framework to generate Web services security configurations. In the same line, intermediate model structure is introduced by Satoh et al. [109, 110] to simplify the transformation rules for transforming a security policy written in WebService-SecurityPolicy into platform-specific configuration files.

Aspect-Oriented Modelling in MDS: AOM techniques would be ideal for MDS with fully separation of concerns support. With AOM, security concerns can be modelled separately, and then automatically composed into primary models. All of the reviewed MDS approaches in this category except [106, 131] tackle multiple security concerns. These approaches aim at dealing with multiple security concerns as one would expect from any AOM approach. Georg et al. [50] propose a methodology that allows not only security mechanisms but also attacks to be modelled as aspect models.

[14] `http://www.eclipse.org/viatra/`

Table 8: Summary of the less common/emerging MDS Approaches - Part 1

	Security Concerns	Modeling Approach				MMT		MTT		Verification	Application Domains	Validation
		Language	AOM	SOC	Type	Level	Impl	Both	Impl			
Pattern-Based by Abramov et al. [1, 2, 3]	access control	UML	X	✓	S	Exo	ATL	✓	SMT (self-)	✓	database	ACS; CE
Pattern-Based by Bouaziz et al. [29]	access control	UML	X	✓	S	Endo	ATL	✓	NP	X	component-based architecture	ACS
Pattern-Based by Kim et al. [74, 75]	access control	UML	✓	✓	S, B	Endo, Exo	NP	X	X	✓	NR	ACS
Pattern-Based by Schnjakin et al. [112]	integrity, confidentiality, authentication, authorisation	BPMN	X	✓	O	NP	X	X	NP	X	service-oriented architectures	IE
Pattern-Based by Moral-García et al. [91]	integrity, confidentiality, availability, authentication, authorisation	DSL	X	✓	O	Exo	X	X	NP	X	secure cloud computing	IE
Sec@runtime by Almorsy et al. [10, 11, 12]	integrity, confidentiality, availability, authentication, authorisation	UML	✓	✓	S, B	Endo	NP	o	NP	testing	cloud-based applications	ACS, CE
Sec@runtime by Elrakaiby et al. [42]	authorisation	UML, DSL	✓	✓	S, DSM	Exo	NP	X	NP	X	NR	ACS
Sec@runtime by Morin et al. [92]	authorisation	DSL	X	✓	DSM	Exo	Kermeta	o	Kermeta	X	component-based architecture	ACS

Note: Supported (✓); Partially supported (o); Not supported (X); Controlled experiment (CE); Industrial case study (ICS); Academic case study (ACS); Illustrative example (IE); Not provided (NP); Non-restrictive (NR); Self-developed (Self-); Endogenous (Endo); Exogenous (Exo); Structural (S); Behavioural (B); Others (O)

Table 9: Summary of the less common/emerging MDS Approaches - Part 1

	Security Concerns	Modeling Approach				MMT		MTT		Verification	Application Domains	Validation
		Language	AOM	SOC	Type	Level	Impl	Both	Impl			
Sec@runtime by Nguyen et al. [103]	authorisation (access control, delegation)	DSL	X	✓	DSM	Exo	Kermeta	o	Kermeta	X	component-based architecture	ACS
Sec@runtime by Xiao [130]	authorisation	DSL	DSM	X	✓	Exo	NP	✓	JADE(self-)	X	NR	ACS
SecureSOA by Gallino et al. [49]	authorisation	UML	X	✓	S	Exo	NP	o	NP	X	SOA	ACS
SecureSOA by Gilmore et al. [51]	non-functional aspects	UML profiles	X	o	S, B, O	Exo	VIATRA	✓	VIATRA2	✓	distributed systems	ACS
SecureSOA by Hoisl et al. [57, 58]	confidentiality, integrity	UML	X	✓	S, B	Exo	NP	NP	NP	X	SOA	ACS
SecureSOA by Menzel [82] and Menzel et al. [83, 84]	confidentiality, integrity, authentication, authorisation	DSL	X	✓	DSM	Exo	NP	o	NP	X	SOA	ACS
SecureSOA by Nakamura et al. [99]	confidentiality, integrity, availability, authentication	UML profile	X	✓	S	Exo	NP	o	NP	X	SOA	IE
SecureSOA by Satoh et al. [109, 110]	authentication	DSL	X	✓	DSM	Endo	NP	o	NP	X	SOA	IE
SecureSOA by Wada et al. [127]	confidentiality, integrity, authorisation	UML profile	X	✓	S	Exo	ark	✓	ark (self-)	X	SOA	CE
SecureSOA by Wolter et al. [129]	integrity, confidentiality, availability, authentication, authorisation	DSL	X	✓	DSM	Exo	NP	o	NP	X	SOA	IE

Note: Supported (✓); Partially supported (o); Not supported (X); Controlled experiment (CE); Industrial case study (ICS); Academic case study (ACS); Illustrative example (IE); Not provided (NP); Non-restrictive (NR); Self-developed (Self-); Endogenous (Endo); Exogenous (Exo); Structural (S); Behavioural (B); Others (O)

Table 10: Summary of the less common/emerging MDS Approaches - Part 2

	Security Concerns	Modeling Approach				MMT		MTT		Verification	Application Domains	Validation
		Language	AOM	SOC	Type	Level	Impl	Both	Impl			
AOMsec by Georg et al. [50]	integrity, confidentiality, availability, authentication, authorisation	UML	✓	✓	S, B	Endo	NP	NP	NP	✓	NR	IE
AOMsec by Mouheb et al. [94] and Mouheb et al. [95]	confidentiality, authorisation	UML	✓	✓	S, B, O	Endo	QVT	✓	RSA	X	NR	ACS
AOMsec by Ray et al. [106]	authorisation (AC)	UML	✓	✓	S, O	Endo	NP	NP	NP	X	NR	IE
AOMsec by Sánchez et al. [108]	confidentiality, integrity, authorisation	UML profile	✓	✓	S, B	Exo	QVT	o	NP	X	NR	ACS
AOMsec by Zhu et al. [131]	confidentiality, integrity, availability	UML profile	✓	✓	S, B	Exo	NP	o	aspect code gen (self-)	✓	NR	ICS
Access Control oriented by Ahn et al. [4]	authorisation	UML	X	X	S, O	NP	NP	o	Octopus + Dresden OCL toolkit	X	NR	CE
Access Control oriented by Burt et al. [36]	authorisation	DSL	X	✓	DSM	Exo	NP	NP	NP	X	NR	IE
Access Control oriented by Fink et al. [47]	authorisation	DSL	X	✓	DSM	Exo	Graph Transformation	NP	NP	X	NR	ACS
Access Control oriented by Kim et al. [76]	authorisation (AC)	UML	✓	✓	S, B	Endo	IBM RSA	✓	IBM RSA	o	NR	ACS
Access Control oriented by Mouelhi et al. [93]	authorisation	DSL	X	✓	DSM	Exo	NP	o	NP	✓(testing)	NR	ACS

Note: Supported (✓); Partially supported (o); Not supported (X); Controlled experiment (CE); Industrial case study (ICS); Academic case study (ACS); Illustrative example (IE); Not provided (NP); Non-restrictive (NR); Self-developed (Self-); Endogenous (Endo); Exogenous (Exo); Structural (S); Behavioural (B); Others (O)

Table 11: Summary of the less common/emerging MDS Approaches - Part 2

	Security Concerns	Modeling Approach				MMT		MTT		Verification	Application Domains	Validation
		Language	AOM	SOC	Type	Level	Impl	Both	Impl			
Access Control oriented by Kaddani et al. [70]	authorisation (Or-BAC)	UML	X	✓	S	Exo	NP	o	NP	X	electrical grid	IE
Access Control oriented by Pavlich-Mariscal et al. [105]	authorisation, authentication	UML profile	X	✓	S	Endo	NP	✓	self-	o	NR	ACS
Access Control oriented by Sohr et al. [114]	authorisation	UML	X	✓	S, B, O	NP	NP	NP	NP	X	distributed systems	ACS
Access Control oriented by Schefer-Wenzl et al. [111]	authorisation	UML profile	X	✓	S, B, O	Exo	NP	NP	NP	X	NR	ACS
Access Control oriented by Bertolino et al. [20]	authorisation	DSL	X	✓	S, B	Exo	Kermeta	o	NP	model-based testing	NR	ACS
Usage Control by Neisse et al. [100]	authorisation (UCON)	DSL	X	✓	DSM, S, B, O	Exo	Java-based tool	o	Java-based tool (self-)	X	NR	ACS
ModelSec by Sánchez et al. [107]	integrity, confidentiality, availability, authentication, authorisation	DSL (SecML)	X	✓	DSM	Exo	RubyTL	o	MOFScript	X	NR	ACS
Secure Web Apps by Busch et al. [37]	integrity, confidentiality, availability, authentication, authorisation	UML profile	X	✓	DSM	Exo	NP	o	XPand	testing possible	web applications	ACS
SecEmbedded by Eby et al. [41]	confidentiality, availability	DSL	X	✓	DSM	Exo	NP	NP	NP	X	embedded systems	ACS

Note: Supported (✓); Partially supported (o); Not supported (X); Controlled experiment (CE); Industrial case study (ICS); Academic case study (ACS); Illustrative example (IE); Not provided (NP); Non-restrictive (NR); Self-developed (Self-); Endogenous (Endo); Exogenous (Exo); Structural (S); Behavioural (B); Others (O)

been taken into account.

MDS for Access Control: Section 5.1 shows that access control problem got the most attention from the MDS community. We discuss here some representative MDS approaches that specifically address access control. Ahn et al. [4] propose a framework for representing security model, specifying and validating security policy, and automatically generating security enforcement codes. This framework leverages the MDD approach together with a systematic tool to build secure systems. Also presenting a MDD approach for access control, Fink et al. [47] aim at developing access control policies for distributed systems using MOF and UML profiles. However, this approach does not work well with module-based system like systems based on SOAP [16]. Kim et al. [76] present a feature-based approach that enables systematic configuration of RBAC features for developing customisable access control-based enterprise systems. Feature modelling is used for effectively capturing the variabilities of the RBAC. UML models are used for specifying the static and behavioural properties of RBAC features. The composition method in their approach is used for building RBAC configuration, which also serves as a verification point for correctness of composition. Aiming at a full design-to-testing MDD process, Mouelhi et al. [93] introduce a generic access control metamodel. The generic access control policy model specified by the metamodel is automatically transformed into security policy for the XACML platform, and integrated in the target application using aspect-oriented programming. Model-based mutation testing makes the access control enforcement quantitatively testable. Pavlich-Mariscal et al. [105] propose a MD framework with a set of composable access control features that can be tightly integrated into the UML. At the code level, access control is map to the policy code which realises access control diagrams and features, and the enforcement code, to restrict access to methods based on information of the policy code. The degree of traceability of mappings is assessed. Recently, Schefer-Wenzl et al. [111] propose a full MDD approach for specifying and enforcing break-glass policies in process-aware information systems. By tackling a complex security exception handling mechanism like break-glass policies with MDS, this work shows developing

[16] `http://www.w3.org/TR/soap/`

DSLs for specific security concerns are a good way to capture well the semantics of these concerns. Based on that, a typical MDD process can be developed for derive security from specification to enforcement with tools support. Bertolino et al. [20] even go further in terms of tools support by providing a toolchain for designing, generating. and testing access control policies. This toolchain is the result of integrating specific tools for specific stages of the development cycle that have been developed in a collaborative research network. The research around UMLsec has also resulted in various tools support but not yet systematically formed a tool chain.

Miscellaneous: Neisse et al. [100] present one of few MDS approaches about usage control, the next generation of access control. Consisting of authorisations and obligations, high-level usage control policies are specified considering an abstract system model and automatically refined with the help of policy refinement rules to implementation-level policies. The work by Elrakaiby et al. [42] mentioned above can also be categorised as usage control. In the domain of securing embedded systems, the approach we reviewed is by Eby et al. [41]. The authors propose a framework to incorporate security modelling into embedded system design. Their security analysis tool is capable of analysing the flow of data objects through a system and identifying points that are vulnerable to attack. Not restricted to a particular application domain, ModelSec by Sánchez et al. [107] can deal with multiple security concerns in an integrated fashion, including privacy, integrity, access control, authentication, availability, non-repudiation, and auditing. ModelSec supports defining and managing security requirements by building security requirements models for an application from which operational security models can then be generated. Recently, Busch et al. [37] present an MDS approach specific for securing web applications, tackling multiple security concerns. The graphical, UML-based Web Engineering (UWE) language is extended for specifying security concerns in web applications. Moreover, the approach is mapped to an iterative development cycle from requirement specification to testing and deployment with tools support.

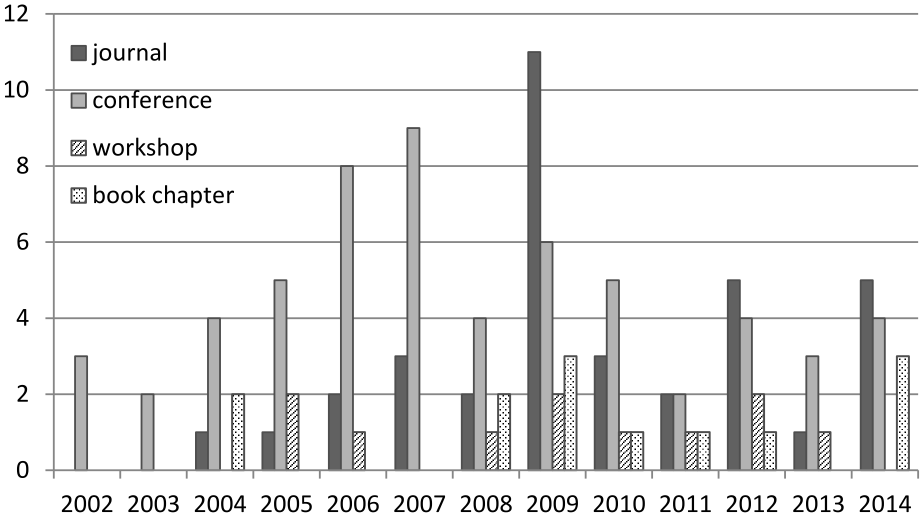


Figure 9: Trend of MDS publication

5.4. Trend analysis of MDS *approaches*

In terms of publication, we can see in Fig. 9 there was a peak time for primary MDS publications in 2009. As we mentioned, the primary MDS approaches were first introduced from 2002. From 2002-2008, more primary MDS papers were published at conferences than journals. The number of primary MDS papers published at conferences were going up until 2007. In 2008, the number of primary MDS papers published at conferences decreased. One of the reasons could be primary MDS papers were under submission to journals. In 2009, there was a peak number of primary MDS papers published in journals. After the peak in 2009, the trend of primary MDS publications looks more stable for the period 2010-2014. From 2010 to 2014, less primary MDS papers were published than the previous 5-year period (2005-2009). However, the trend of publishing primary MDS papers in the period 2010-2014 seems more stable.

Similarly to the trend of publications, the trend of how security concerns have been addressed also has a peak time in 2009. Fig. 10 shows that, nearly all the time reviewed, authorisation is the concern that has been addressed the most. Only in 2009, confidentiality was tackled by more primary MDS papers than authorisation. The other concerns were always less focused than authorisation and confidentiality all the time

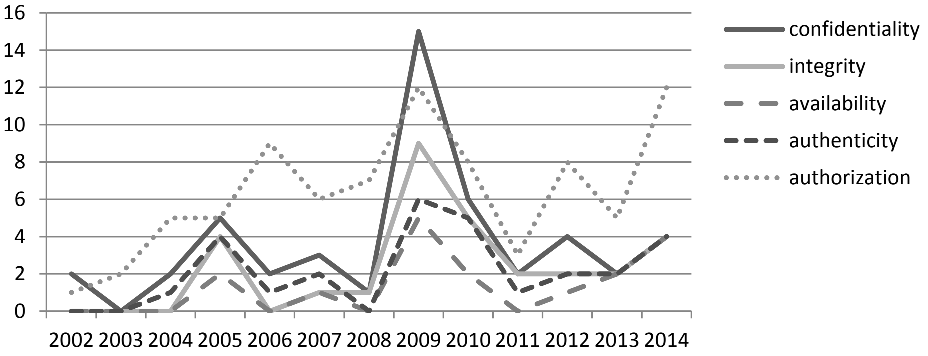


Figure 10: Trend of security concerns addressed by MDS studies

reviewed. Until 2014, authorisation looks like still being addressed the most by the MDS research community. MDS researchers should pay more attention to the less tackled security concerns, and should aim at a solution addressing multiple security concerns simultaneously.

The trends of how MDE artefacts leveraged in the primary MDS approaches look well coupled with the number of primary MDS publications. The line of each artefact is very close to the others (see Fig. 11). This means that most primary MDS approaches did leverage the key artefacts of MDE in secure systems development. It is easily understandable that as long as we clearly define how an approach can be considered an MDS approach, most of the key MDE artefacts have to be leveraged in an MDS approach. This trend should hold in the future as well.

In terms of publication venues, Information and Software Technology (IST) journal and ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS) are so far the most popular venues for publication of primary MDS papers. Fig. 13 shows that at least 10 primary MDS publications have been found in each of these two venues. The next two attractive venues for primary MDS papers are ARES (security conference), and SoSym (MDE journal). Primary MDS papers were also published at some other general journals (Journal of Universal Computer Science) or domain specific conferences (IEEE International Conference on Web Services). The proceedings of Tutorial Lectures on Foundations of Security Analysis and Design (FOSAD) contains some significant primary MDS approaches as well. In

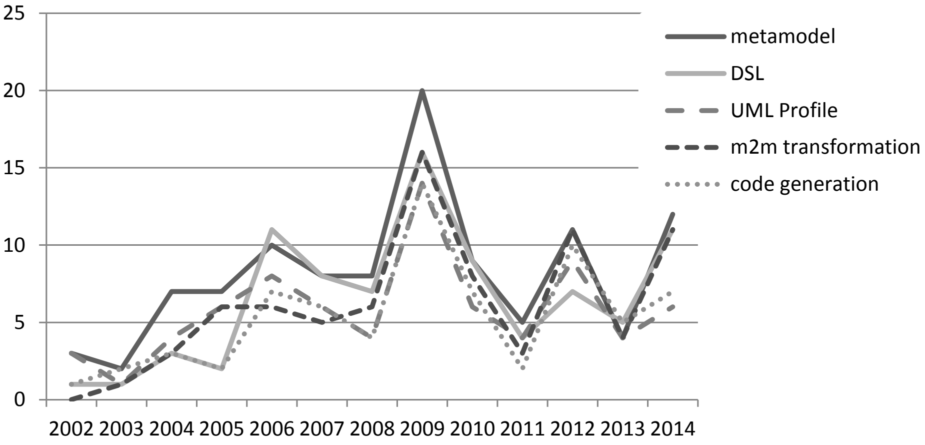


Figure 11: Trend of MDE artefacts leveraged by MDS studies

general, except ARES and CSJ, conferences and journals specific for security do not seem to be the common venues for MDS publications yet.

6. Threats to validity

We discuss the threats to validity of this SLR according to the lessons learned on validity in SLRs [77] and our own experience.

6.1. The search process

To maximise the relevant articles returned by the search engines, we kept the search string not too specific but still reflecting what we wanted to search for. Moreover, the search string was used for searching not only in the titles, abstracts but also in the full text of an article. Only the search engine of Web of Knowledge (ISI) does not provide the option for searching in full text. This limitation could affect the search results returned by ISI. To minimise the possibility of missing relevant papers, we kept our search string generic so that we cover as many relevant papers as possible (more than 10 thousands relevant papers found). To complement for the automatic search, we have also conducted the manual search on relevant journals and proceedings of relevant conferences. Then, to mitigate the limitations of automatic and manual searches, we have adopted the snowballing strategy. Even though only three out of five steps of

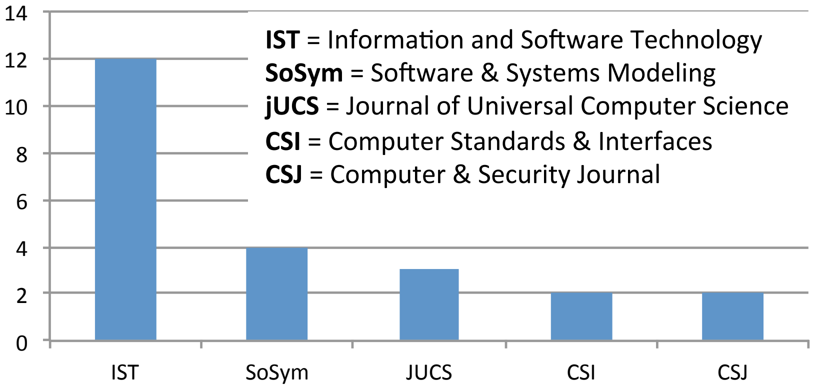
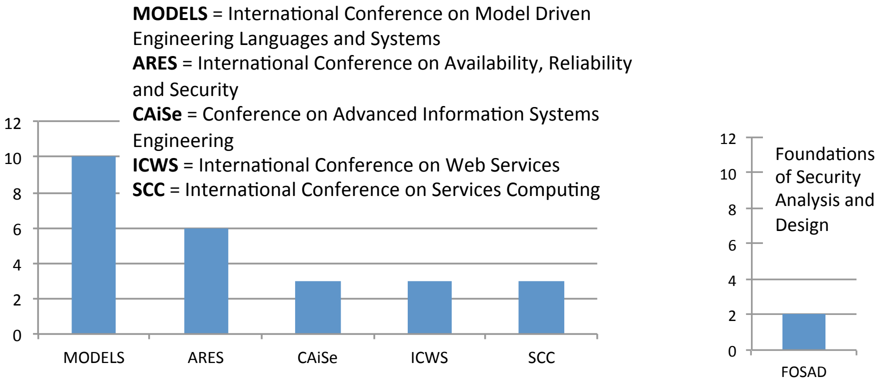


Figure 12: Number of papers for the journals with the most MDS papers found in this review



(a) conferences

(b) workshops/others

Figure 13: Number of papers per conference, workshop/other with the most MDS papers found in this review

the snowballing strategy were adopted, those are the key steps. Moreover, we already conducted the extensive automatic and manual searches which covered thousands of relevant publications, and resulted in a large set of primary MDS papers. That is why conducting only three key steps of snowballing strategy would be fair enough. Another possible threat is that we did not extensively search for books related to MDS. However, we did include the option to also search for book chapters while performing automatic search. In fact, we found out some book chapters that got into our final selected papers for data extraction, e.g. [54], [64].

6.2. Selection of primary studies

A large part of the search and selection process was conducted by the first author. Some publications might have been missed. To mitigate this risk, every doubtful or ”borderline” publication was not dismissed in the first place but rather being cross-checked and discussed by all the reviewers. Additionally, our clearly predefined review protocol with inclusion and exclusion criteria helped to reduce the reviewers' bias in the selection of primary studies.

The results of this SLR papers are based on the data extracted and synthesised from the selected MDS studies. Note that we have applied the citation criterion to estimate the quality and impact factors of the selected primary MDS studies. Even though this criterion is not too strict, applying it caused a number of MDS papers not to be included. We realized that some of the excluded MDS papers are related to the included primary MDS studies. To mitigate the risk of missing some important data of the primary MDS studies, we put back the excluded MDS papers that are related to the primary MDS studies. In total, we re-selected 15 MDS papers as the ”sidekick” papers to be included in the final set for data extraction.

Some key selection criteria in this SLR are time-bound. The citation criterion for selecting primary MDS papers is based on the numbers of citations provided by Google Scholar engine. The selection of venues for conducting manual search is based on Microsoft Research ranking website. Google citations will change from time to time. Similarly, rankings of conferences and journals will change. Those time-bound metrics influence the reproduction of this SLR. So, some papers which were not selected as primary MDS papers because of the citation criterion would satisfy this criterion later on.

7. Related work

In [71], the authors present a survey on MDS. They propose an evaluation based on the work of Khwaja and Urban [73]. The study revealed that approaches that analyse implementations of modelled systems are still missing. Due to the fact that implementations are not generated automatically from formal specifications, verification of running code is reasonable. The main drawback of [71] is that it is not a SLR. As a result, there are some well-known approaches that are missing in [71], such as SecureUML [18].

In [13], Basin et al. went through a "Decade of Model-Driven Security" by presenting a survey focusing on their specific MDS approach called SecureUML. The authors claim that MDS has enormous potential, mainly because Security-Design Models provide a clear, declarative, high-level language for specifying security details. The potential is even more, when the security models rely on a well-defined semantics. The main drawback of [13] is that it only considers the work around SecureUML.

[121] is a survey of model-based security methodologies for distributed systems. The papers surveyed in [121] are not only about model-driven methodologies but also architecture-driven methodologies, pattern-driven methodologies, and agent-driven methodologies. Thus the focus is not specifically MDS but rather security engineering for distributed systems in general. Our paper explicitly targets MDS methodologies as described in the previous sections.

In [79], five well-known MDS approaches, i.e. UMLsec, SecureUML, Sectet, ModelSec, and SecureMDD, are summarised, evaluated, and discussed. These five MDS approaches are also confirmed in this paper. It can be seen that our SLR results are complementary to the contributions of the normal survey papers, e.g. [79], [121]. Those survey papers perform in depth analysis of some significant MDS approaches by elaborating one after another. But our SLR performs a SLR in both width and depth of MDS research which result in not only (evidently) significant MDS approaches but also emerging considerable MDS approaches. It is the first MDS literature review that systematically considers all relevant publications using explicit evaluation and extraction criteria. Furthermore, our SLR provides a detailed look at all the key artefacts of any MDS approaches such as modelling techniques, security concerns, how model transformations employed, how verification and validation methods used, and case studies, and application domains. We also provide a trend analysis for the development of MDS research area.

[62] is closer to our SLR. The authors propose three research questions with the goal to determine if the current MDS approaches focus on code generation and/or having empirical studies. The study shows that there is a need for more empirical studies on MDS (none exists), and that standardisation is key to achieve the objectives of MDD/MDS (which are increased portability and interoperability). However, [62] presents several drawbacks and differences from our paper. First, their search strategy is very limited compared to our three-pronged search strategy. Second, concerning the SLR protocol, no evaluation criteria and data extraction strategy are given. Moreover, their exclusion criteria are very narrow. Consequently, the authors exclude significant papers in the field, e.g. UMLsec papers. Also, the authors exclude AOM approaches, because they consider that AOM does not consider security aspects as specific aspects (i.e. different from other aspects). Our work covers all the limitations of [62] and provides much more extensive SLR on the topic.

8. Conclusions

We have presented an extensive systematic literature review on model-driven approaches for developing secure systems. The SLR is based on a rigorous three-pronged search process, which combined automatic search and manual search with snowballing strategy. Using 9 clearly predefined selection criteria, 108 MDS papers have been strictly selected, and then reviewed. From these primary MDS papers, we extracted and synthesised the data to answer three research questions: How do these approaches support the development of secure systems? What are the limitations? What are open issues to be further investigated?

The results show that most MDS papers focus on *authorisation* (75 %) and *confidentiality* (42 %) while only few publications address further security concerns like *integrity*, *availability*, and *authentication* (RQ 1.1). Moreover, very few MDS papers deal with multiple security concerns simultaneously in a systematic way, e.g. 9 % address authentication, authorisation, and confidentiality together (RQ 1.2). MMTs were mentioned in most of the identified MDS papers (74 %), but more than half of the papers do not provide detailed information on the used languages, tools, or transformation rules (56 %) and only a few mention standard transformation languages (19 %), such as ATL or QVT (RQ 1.4). MTTs were mentioned slightly less often (64 %) than transformations to models and were used almost equally often to generate only security infrastructure (34 %) or also functional code (29 %) (RQ 1.4). Most papers discuss illustrative examples or academic case studies (67 %) but do not mention in-depth evaluations, e.g. industrial case studies (5 %), controlled experiments (2 %) or common benchmarks (RQ 1.5). Although most papers do not mention a specific application domain there are domains that are discussed more frequently, such as distributed or service-oriented systems (31 %) and data warehouses (19 %).

Altogether, our literature review shows that many MDS approaches are limited to specific, isolated security concerns and are specialized to certain application domains. They often show a lack of formality, automation, process-integration and evaluation. Our findings urge for more attention from the MDS research community to less tackled security concerns, such as availability and to approaches that systematically consider inter-relations between multiple security. An important open issue is the lack of rigorous evaluations of claimed benefits and capabilities of MDS approaches.

Independent of our initial research questions, our SLR revealed five significant MDS approaches that can be classified as more mature than the rest. But we also identified various emerging/less common MDS approaches that respond to recent developments, such as cloud-based environments. With trend analyses for the last twelve years we showed that there was a clear peak of publications on MDS in 2009, which mainly results from an increase in journal publications. Finally, our analysis of publication venues showed that the journal on Information and Software Technology and the MODELS conference published most of the identified MDS papers.

In future work, our SLR protocol and the list of finally selected MDS papers could be used for a follow-up SLR of MDS to identify papers that are published after this review. A reviewer would need to check again the citation criterion for those primary MDS papers using up-to-date citation numbers. After obtaining a subset of MDS papers from the original set, only forward snowballing would have to be conducted for this subset as backward snowballing cannot reveal newly published papers in references of old papers. After reviewing and selecting a new set of MDS papers from the result of forward snowballing, the full snowballing process could be performed on it to obtain a new final set. For the newly found papers in this final set data extraction would have to be performed in order to obtain up-to-date results on new MDS publications.

9. Acknowledgments

This work is supported by the Fonds National de la Recherche (FNR), Luxembourg, under the MITER project C10/IS/783852 and the KASTEL project by the German Federal Ministry of Education and Research under grant BMBF 01BY1172.

References

[1] J. Abramov et al. “A methodology for integrating access control policies within database development”. In: *Computers & Security* 31.3 (2012), pp. 299–314.

[2] J. Abramov et al. “Evaluation of the Pattern-based method for Secure Development (PbSD): A controlled experiment”. In: *Information and Software Technology* 54.9 (2012), pp. 1029–1043.

[3] J. Abramov et al. “Tool support for enforcing security policies on databases”. In: *IS Olympics: Information Systems ...* (2012), pp. 126–141.

[4] G. Ahn and H. Hu. “Towards realizing a formal RBAC model in real systems”. In: *Proceedings of the 12th ACM symposium on Access ...* (2007), p. 215.

[5] M. Alam et al. “Model driven security for Web services (MDS4WS)”. In: *Multitopic Conference, 2004. Proceedings of INMIC 2004. 8th International*. 2004, pp. 498–505.

[6] M. Alam. “Model driven security engineering for the realization of dynamic security requirements in collaborative systems”. In: *Models in Software Engineering* 4364 (2007), pp. 278–287.

[7] M. Alam et al. “A constraint based role based access control in the SECTET a model-driven approach”. In: *... of the 2006 International Conference on ...* c (2006), pp. 1–13.

[8] M. Alam et al. “A framework for modeling restricted delegation in service oriented architecture”. In: *Trust and Privacy in Digital ...* (2006), pp. 142–151.

[9] M. Alam et al. “Modeling permissions in a (U/X) ML world”. In: (2006), pages.

[10] M. Almorsy and J. Grundy. “SecDSVL: A Domain-Specific Visual Language To Support Enterprise Security Modelling”. In: *Software Engineering Conference (ASWEC), 2014 23rd Australian*. IEEE. 2014, pp. 152–161.

[11] M. Almorsy et al. “Adaptable, model-driven security engineering for SaaS cloud-based applications”. In: *Automated Software Engineering* 21.2 (2013), pp. 187–224.

[12] M. Almorsy et al. “Mdse@ r: model-driven security engineering at runtime”. In: *Cyberspace Safety and Security*. Springer, 2012, pp. 279–295.

[13] D. Basin et al. “A decade of model-driven security”. In: *Proceedings of the 16th ACM symposium on Access control models and technologies*. SACMAT ’11. ACM, 2011, pp. 1–10.

[14] D. Basin et al. “A Metamodel-Based Approach for Analyzing Security-Design Models”. In: *Model Driven Engineering Languages and Systems*. Vol. 4735. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 420–435.

[15] D. Basin et al. “A model-driven methodology for developing secure data-management applications”. In: *Software Engineering, IEEE Transactions on* 40.4 (2014), pp. 324–337.

[16] D. Basin et al. “Automated analysis of security-design models”. In: *Information and Software Technology* 51.5 (2009), pp. 815–831.

[17] D. Basin et al. “Model driven security for process-oriented systems”. In: *Proceedings of the eighth ACM symposium on Access control models and technologies*. SACMAT ’03. ACM, 2003, pp. 100–109.

[18] D. Basin et al. “Model driven security: From UML models to access control infrastructures”. In: *ACM Trans. Softw. Eng. Methodol.* 15 (2006), pp. 39–91.

[19] D. Basin et al. “Model-driven development of security-aware GUIs for data-centric applications”. In: *Foundations of security . . .* (2011), pp. 101–124.

[20] A. Bertolino et al. “A Toolchain for Designing and Testing Access Control Policies”. In: *Engineering Secure Future Internet Services and Systems*. Springer, 2014, pp. 266–286.

[21] B. Best et al. “Model-Based Security Engineering of Distributed Information Systems Using UMLsec”. In: *29th International Conference on Software Engineering, 2007. ICSE 2007.* 2007, pp. 581–590.

[22] J. Bezivin. “Model Driven Engineering: An Emerging Technical Space”. In: *GTTSE, pp.36-64* (2006).

[23] J. Biolchini et al. “Systematic review in software engineering”. In: *System Engineering and Computer Science Department COPPE/UFRJ, Technical Report ES* 679.05 (2005), p. 45.

[24] C. Blanco. “Applying QVT in order to implement secure data warehouses in SQL Server Analysis Services”. In: *Journal of Research . . .* 41.2 (2009).

[25] C. Blanco and I. de Guzmán. “Showing the Benefits of Applying a Model Driven Architecture for Developing Secure OLAP Applications.” In: *J. UCS* 20.2 (2014), pp. 79–106.

[26] C. Blanco and R. Pérez-Castillo. “Towards a modernization process for Secure Data Warehouses”. In: *Data Warehousing and . . .* Mda 2003 (2009), pp. 24–35.

[27] C. Blanco et al. “Modernizing Secure OLAP Applications with a Model-Driven Approach”. In: *The Computer Journal* (2014).

[28] M. Borek et al. “Model-Driven Development of Secure Service Applications”. In: *2012 35th Annual IEEE Software Engineering Workshop* (2012), pp. 62–71.

[29] R. Bouaziz et al. “An Engineering Process for Security Patterns Application in Component Based Models”. In: *2013 Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*. Ieee, 2013, pp. 231–236.

[30] C. Braga. “A transformation contract to generate aspects from access control policies”. In: *Software and Systems Modeling* 10.3 (2011), pp. 395–409.

[31] M. Brambilla et al. *Model-Driven Software Engineering in Practice*. 1st ed. Synthesis Lectures on Software Engineering. Morgan & Claypool Publishers, 2012.

[32] R. Breu et al. “Model based development of access policies”. In: *International Journal on Software Tools for Technology Transfer* 9.5-6 (2007), pp. 457–470.

[33] R. Breu et al. “Model driven security for inter-organizational workflows in e-government”. In: *E-Government: Towards Electronic . . .* (2005), pp. 122–133.

[34] R. Breu et al. “Model-Driven Security Engineering of Service Oriented Systems”. In: *Information Systems and e-Business Technologies*. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, 2008, pp. 59–71.

[35] A. Brucker et al. “A Model Transformation Semantics and Analysis Methodology for SecureUML”. In: Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, pp. 306–320.

[36] C. Burt et al. “Model driven security: unification of authorization models for fine-grain access control”. In: *Enterprise Distributed . . .* (2003), pp. 159–171.

[37] M. Busch et al. “Modeling Security Features of Web Applications”. In: *Engineering Secure Future Internet Services and Systems*. Springer, 2014, pp. 119–139.

[38] M. Clavel et al. “Model-Driven Security in Practice: An Industrial Experience”. In: *Model Driven Architecture – Foundations and Applications*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 326–337.

[39] I. Crnkovic et al. “A classification framework for software component models”. In: *Software Engineering, IEEE Transactions on* 37.5 (2011), pp. 593–615.

[40] M. de Dios et al. “Model-Driven Development of a Secure eHealth Application”. In: *Engineering Secure Future …* (2014), pp. 97–118.

[41] M. Eby and J. Werner. “Integrating security modeling into embedded system design”. In: *Engineering of Computer- …* (2007), pp. 221–228.

[42] Y. Elrakaiby et al. “Security@ Runtime: A Flexible MDE Approach to Enforce Fine-grained Security Policies”. In: *Engineering Secure Software and …* (2014), pp. 19–34.

[43] E. Fernández-Medina and M. Piattini. “Extending OCL for secure database development”. In: *UML 2004 - The Unified Modeling Language …* (2004), pp. 380–394.

[44] E. Fernández-Medina and J. Trujillo. “Extending UML for designing secure data warehouses”. In: *… Modeling–ER 2004* (2004), pp. 217–230.

[45] E. Fernández-Medina and M. Piattini. “Designing secure databases”. In: *Information and Software Technology* 47.7 (2005), pp. 463–477.

[46] E. Fernández-Medina et al. “Developing secure data warehouses with a UML extension”. In: *Information Systems* 32.6 (2006), pp. 826–856.

[47] T. Fink et al. “An MDA approach to access control specifications using MOF and UML profiles”. In: *Electronic Notes in Theoretical Computer Science* 142 (2006), pp. 161–179.

[48] J. Fox and J. Jurjens. “Introducing security aspects with model transformation”. In: *… of Computer-Based Systems, 2005. ECBS’05. …* (2005).

[49] J. Gallino and M. D. Miguel. “Domain-Specific multi-modeling of security concerns in service-oriented architectures”. In: *Web Services and Formal …* (2012), pp. 128–142.

[50] G. Georg et al. “An aspect-oriented methodology for designing secure applications”. In: *Information and Software Technology* 51.5 (2009), pp. 846–864.

[51] S. Gilmore et al. “Non-functional properties in the model-driven development of service-oriented systems”. In: *Software & Systems Modeling* 10.3 (2010), pp. 287–311.

[52] M. Hafner and R. Breu. “Realizing model driven security for inter-organizational workflows with ws-cdl and uml 2.0”. In: *Model Driven Engineering Languages and Systems* 3713 (2005), pp. 39–53.

[53] M. Hafner and R. Breu. “Extending Sectet : Advanced Security Policy”. In: *Security Engineering for Service-Oriented Architectures*. Springer Berlin Heidelberg, 2009, pp. 159–188.

[54] M. Hafner and R. Breu. “Modeling Security Critical SOA Applications”. In: *Security Engineering for Service-Oriented Architectures*. Springer Berlin Heidelberg, 2009, pp. 93–119.

[55] M. Hafner et al. “Modeling and enforcing advanced access control policies in healthcare systems with sectet”. In: *Models in Software Engineering* (2008), pp. 132–144.

[56] M. Hafner et al. “Towards a MOF/QVT-based domain architecture for model driven security”. In: *Model Driven Engineering Languages and ...* (2006), pp. 275–290.

[57] B. Hoisl and S. Sobernig. “Integrity and confidentiality annotations for service interfaces in SoaML models”. In: *Availability, Reliability and Security (ARES ...* (2011), pp. 673–679.

[58] B. Hoisl et al. “Modeling and enforcing secure object flows in process-driven SOAs: an integrated model-driven approach”. In: *Software & Systems Modeling* 13.2 (2012), pp. 513–548.

[59] J.-M. Horcas et al. “An aspect-oriented model transformation to weave security using CVL”. In: *Model-Driven Engineering and Software Development (MODELSWARD), 2014 2nd International Conference on*. IEEE. 2014, pp. 138–150.

[60] J. Hutchinson et al. “Empirical assessment of MDE in industry”. In: *Proceedings of the 33rd International Conference on Software Engineering*. ICSE ’11. ACM, 2011, pp. 471–480.

[61] S. Jalali and C. Wohlin. “Systematic literature studies: database searches vs. backward snowballing”. In: *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*. ACM. 2012, pp. 29–38.

[62] J. Jensen and M. G. Jaatun. “Security in Model Driven Development: A Survey”. In: *Proceedings of the 2011 Sixth International Conference on Availability, Reliability and Security*. ARES ’11. IEEE Computer Society, 2011, pp. 704–709.

[63] J. Jürjens. “Model-based security engineering for real”. In: *FM 2006: Formal Methods* (2006), pp. 600–606.

[64] J. Jürjens. “Model-based security engineering with UML”. In: *Foundations of Security Analysis and Design III* (2005), pp. 42–77.

[65] J. Jürjens. “UMLsec: Extending UML for secure systems development”. In: *UML 2002 – The Unified Modeling Language* (2002), pp. 1–9.

[66] J. Jürjens. “Using UMLsec and goal trees for secure systems development”. In: *Proceedings of the 2002 ACM symposium on Applied . . .* (2002), pp. 1026–1030.

[67] J. Jürjens and S. Houmb. “Dynamic secure aspect modeling with UML: From models to code”. In: *Model Driven Engineering Languages and Systems* (2005), pp. 142–155.

[68] J. Jürjens and P. Shabalin. “Tools for secure systems development with UML”. In: *. . . Journal on Software Tools for Technology Transfer . . .* 9.5-6 (2007), pp. 527–544.

[69] J. Jürjens et al. “Incremental security verification for evolving UMLsec models”. In: *Modelling Foundations and . . .* (2011), pp. 52–68.

[70] A. Kaddani et al. “Towards a Model Driven Security for critical infrastructures using OrBAC”. In: *Multimedia Computing and Systems (ICMCS), 2014 International Conference on*. IEEE. 2014, pp. 1235–1240.

[71] K. Kasal et al. “Model-Driven Development Meets Security: An Evaluation of Current Approaches”. In: *Proceedings of the 2011 44th Hawaii International Conference on System Sciences*. HICSS ’11. IEEE Computer Society, 2011, pp. 1–9.

[72] B. Katt et al. “Enhancing Model Driven Security through Pattern Refinement Techniques”. In: *Formal Methods for Components . . .* Ffg 822740 (2013), pp. 169–183.

[73] A. A. Khwaja and J. E. Urban. “A Synthesis of Evaluation Criteria for Software Specifications and Specification Techniques”. In: *International Journal of Software Engineering and Knowledge Engineering* 12.05 (2002), pp. 581–599. eprint: `http://www.worldscientific.com/doi/pdf/10.1142/S0218194002001062`.

[74] D.-K. Kim and P. Gokhale. “A Pattern-Based Technique for Developing UML Models of Access Control Systems”. In: *Computer Software and Applications Conference, 2006. COMPSAC ’06. 30th Annual International*. 2006, pp. 317–324.

[75] D.-K. Kim et al. “Modeling Role-Based Access Control Using Parameterized UML Models”. In: *Fundamental Approaches to Software Engineering*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 180–193.

[76] S. Kim et al. “A feature-based approach for modeling role-based access control systems”. In: *Journal of Systems and Software* 84.12 (2011), pp. 2035–2052.

[77] B. Kitchenham. “Guidelines for performing systematic literature reviews in software engineering”. In: *EBSE Technical Report* (2007).

[78] T. Lodderstedt et al. “SecureUML: A UML-Based Modeling Language for Model-Driven Security”. In: *UML 2002 – The Unified Modeling Language*.

Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, pp. 426–441.

[79] L. Lucio et al. *Advances in Model-Driven Security*. Elsevier, 2014.

[80] M. Memon and G. Menghwar. “Security modeling for service-oriented systems using security pattern refinement approach”. In: *... and Systems Modeling* (2012).

[81] T. Mens and P. Van Gorp. “A taxonomy of model transformation”. In: *Electronic Notes in Theoretical Computer Science* 152 (2006), pp. 125–142.

[82] M. Menzel. “A Pattern-driven Generation of Security Policies for Service-oriented Architectures”. In: *Web Services (ICWS), ...* (2010), pp. 243–250.

[83] M. Menzel and C. Meinel. “A Security Meta-model for Service-Oriented Architectures”. In: *2009 IEEE International Conference on Services Computing* (2009), pp. 251–259.

[84] M. Menzel and C. Meinel. “SecureSOA Modelling Security Requirements for Service-Oriented Architectures”. In: *Services Computing (SCC), 2010 IEEE ...* (2010), pp. 146–153.

[85] N. Moebius and K. Stenzel. “Model-Driven Code Generation for Secure Smart Card Applications”. In: *Software Engineering ...* (2009), pp. 44–53.

[86] N. Moebius et al. “Formal Verification of Application-Specific Security Properties in a Model-Driven Approach Example : A Copycard Application”. In: (2010), pp. 166–181.

[87] N. Moebius et al. “Incremental Development of large, secure Smart Card Applications”. In: *mdsec2012.pst.ifi.lmu.de* (2012), pp. 1–6.

[88] N. Moebius et al. “Modeling Security-Critical Applications with UML in the SecureMDD Approach”. In: *International Journal On Advances in ...* 1.1 (2009), pp. 59–79.

[89] N. Moebius et al. “Generating formal specifications for security-critical applications - A model-driven approach”. In: *Software Engineering for Secure Systems, 2009. SESS ’09. ICSE Workshop on*. 2009, pp. 68–74.

[90] N. Moebius et al. “SecureMDD: A Model-Driven Development Method for Secure Smart Card Applications”. In: *Availability, Reliability and Security, 2009. ARES ’09. International Conference on*. 2009, pp. 841–846.

[91] S. Moral-García et al. “Enterprise security pattern: A model-driven architecture instance”. In: *Computer Standards & Interfaces* 36.4 (2014), pp. 748–758.

[92] B. Morin et al. “Security-driven model-based dynamic adaptation”. In: *Proceedings of the IEEE/ACM international conference on Automated software engineering*. ASE ’10. ACM, 2010, pp. 205–214.

[93] T. Mouelhi et al. “A model-based framework for security policy specification, deployment and testing”. In: *Model Driven Engineering …* 1 (2008), pp. 537–552.

[94] D. Mouheb et al. “An Aspect-Oriented Approach for Software Security Hardening: from Design to Implementation.” In: *SoMeT* (2009).

[95] D. Mouheb et al. “Aspect-Oriented Modeling for Representing and Integrating Security Concerns in UML”. In: *Software Engineering Research, Management and Applications 2010*. Springer Berlin Heidelberg, 2010, pp. 197–213.

[96] H. Mouratidis and P. Giorgini. “Secure tropos: a security-oriented extension of the tropos methodology”. In: *International Journal of Software Engineering and Knowledge Engineering* 17.02 (2007), pp. 285–309.

[97] H. Mouratidis et al. “Secure information systems engineering: experiences and lessons learned from two health care projects”. In: *… Information Systems Engineering* (2009), pp. 231–245.

[98] H. Mouratidis et al. “Towards a comprehensive framework for secure systems development”. In: *Advanced information systems engineering* (2006), pp. 48–62.

[99] Y. Nakamura and M. Tatsubori. “Model-driven security based on a web services security architecture”. In: *Services Computing, …* (2005).

[100] R. Neisse and J. Doerr. “Model-based specification and refinement of usage control policies”. In: *2013 Eleventh Annual Conference on Privacy, Security and Trust* (2013), pp. 169–176.

[101] P. H. Nguyen et al. “A Systematic Review of Model Driven Security”. In: *Proceedings of the 20th APSEC*. 2013.

[102] P. H. Nguyen et al. “Model-driven adaptive delegation”. In: *Proceedings of the 12th annual international conference on Aspect-oriented software development*. AOSD ’13. ACM, 2013, pp. 61–72.

[103] P. H. Nguyen et al. “Modularity and Dynamic Adaptation of Flexibly Secure Systems: Model-Driven Adaptive Delegation in Access Control Management”. In: *Transactions on Aspect-Oriented Software Development XI*. Springer, 2014, pp. 109–144.

[104] T. J. Parr and R. W. Quong. “ANTLR: A predicated-LL (k) parser generator”. In: *Software: Practice and Experience* 25.7 (1995), pp. 789–810.

[105] J. Pavlich-Mariscal et al. “A framework of composable access control features: Preserving separation of access control concerns from models to code”. In: *Computers & Security* 29.3 (2010), pp. 350–379.

[106] I. Ray et al. “An aspect-based approach to modeling access control concerns”. In: *Information and Software Technology* 46.9 (2004), pp. 575–587.

[107]

bibinitperiod Sánchez and F. Molina. “ModelSec : A Generative Architecture for Model-Driven Security”. In: *Journal of Universal …* 15.15 (2009), pp. 2957–2980.

[108] P. Sánchez et al. “Model-driven development for early aspects”. In: *Information and Software Technology* 52.3 (2010), pp. 249–273.

[109] F. Satoh and Y. Yamaguchi. “Generic Security Policy Transformation Framework for WS-Security”. In: *IEEE International Conference on Web Services (ICWS 2007)* Icws (2007), pp. 513–520.

[110] F. Satoh et al. “Adding Authentication to model driven security”. In: *Web Services, 2006. ICWS’06. ...* (2006), pp. 585–594.

[111] S. Schefer-Wenzl and M. Strembeck. “Model-driven specification and enforcement of RBAC break-glass policies for process-aware information systems”. In: *Information and Software Technology* 56.10 (2014), pp. 1289–1308.

[112] M. Schnjakin et al. “A pattern-driven security advisor for service-oriented architectures”. In: *Proceedings of the 2009 ACM workshop on Secure web services - SWS ’09* (2009), p. 13.

[113] D. Simmonds et al. “An aspect oriented model driven framework”. In: *EDOC Enterprise Computing Conference, 2005 Ninth IEEE International*. IEEE. 2005, pp. 119–130.

[114] K. Sohr et al. “Enforcing Role-Based Access Control Policies in Web Services with UML and OCL”. In: *2008 Annual Computer Security Applications Conference (ACSAC)* (2008), pp. 257–266.

[115] E. Soler et al. “Designing Secure Data Warehouses by Using MDA and QVT”. In: *J. UCS* 15.8 (2009), pp. 1607–1641.

[116] E. Soler et al. “A Framework for the Development of Secure Data Warehouses based on MDA and QVT”. In: *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007*. 2007, pp. 294–300.

[117] E. Soler et al. “A set of QVT relations to transform PIM to PSM in the Design of Secure Data Warehouses”. In: *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007*. 2007, pp. 644–654.

[118] E. Soler et al. “Application of QVT for the Development of Secure Data Warehouses: A case study”. In: *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007*. Vol. 3. 1. 2007, pp. 829–836.

[119] J. Trujillo et al. “A UML 2.0 profile to define security requirements for Data Warehouses”. In: *Computer Standards & Interfaces* 31.5 (2009), pp. 969–983.

[120] J. Trujillo et al. “An engineering process for developing Secure Data Warehouses”. In: *Information and Software Technology* 51.6 (2009), pp. 1033–1051.

[121] A. V. Uzunov et al. “Engineering Security into Distributed Systems: A Survey of Methodologies”. In: *Journal of Universal Computer Science* 18.20 (1, 2012), pp. 2920–3006.

[122] B. Vela and C. Blanco. “Model driven development of secure XML data warehouses: a case study”. In: *Proceedings of the 2010 . . .* (2010).

[123] B. Vela et al. “A practical application of our MDD approach for modeling secure XML data warehouses”. In: *Decision Support Systems* 52.4 (2012), pp. 899–925.

[124] B. Vela et al. “Model driven development of secure XML databases”. In: *ACM SIGMOD Record* 35.3 (2006), pp. 22–27.

[125] B. Vela et al. “Development of Secure XML Data Warehouses with QVT”. In: *Information and Software Technology* 55.9 (2013), pp. 1651–1677.

[126] R. Villarroel. “A UML 2.0/OCL extension for designing secure data warehouses”. In: *Journal of Research . . .* 38.1 (2006).

[127] H. Wada et al. “A model-driven development framework for non-functional aspects in service oriented architecture”. In: *International Journal of Web Services . . .* X (2008), pp. 1–31.

[128] C. Wohlin and R. Prikladnicki. “Systematic literature reviews in software engineering”. In: *Information and Software Technology* 55.6 (2013), pp. 919–920.

[129] C. Wolter et al. “Model-driven business process security requirement specification”. In: *Journal of Systems . . .* 55.4 (2009), pp. 211–223.

[130] L. Xiao. “An adaptive security model using agent-oriented MDA”. In: *Information and Software Technology* 51.5 (2009), pp. 933–955.

[131] Z. J. Zhu and M. Zulkernine. “A model-based aspect-oriented framework for building intrusion-aware software systems”. In: *Information and Software Technology* 51.5 (2009), pp. 865–875.