# A Systematic Review of Model-Driven Security

Phu H. Nguyen, Jacques Klein, and Yves Le Traon
*Interdisciplinary Centre for Security, Reliability and Trust (SnT)*
*University of Luxembourg*
*4 rue Alphonse Weicker, L-2721 Luxembourg*
*Email: {phuhong.nguyen, jacques.klein, yves.letraon}@uni.lu*

Max E. Kramer
*Software Design and Quality Group*
*Karlsruhe Institute of Technology*
*Am Fasanengarten 5, D-76131 Karlsruhe, Germany*
*Email: max.e.kramer@kit.edu*

***Abstract*—To face continuously growing security threats and requirements, sound methodologies for constructing secure systems are required. In this context, *Model-Driven Security* (MDS) has emerged since more than a decade ago as a specialized *Model-Driven Engineering* approach for supporting the development of secure systems. MDS aims at improving the *productivity* of the development process and *quality* of the resulting secure systems, with *models* as the main artifact.**

**This paper presents how we systematically examined existing published work in MDS and its results. The systematic review process, which is based on a formally designed review protocol, allowed us to identify, classify, and evaluate different MDS approaches. To be more specific, from thousands of relevant papers found, a final set of the most relevant MDS publications has been identified, strictly selected, and reviewed. We present a taxonomy for MDS, which is used to synthesize data in order to classify and evaluate the selected MDS approaches. The results draw a wide picture of existing MDS research showing the current status of the key aspects in MDS as well as the identified most relevant MDS approaches. We discuss the main limitations of the existing MDS approaches and suggest some potential research directions based on these insights.**

***Keywords*-systematic review; survey; model-driven security; model-driven; security; model; model transformations;**

## I. INTRODUCTION

The further the digital age progresses, the more significant role that security engineering would play, i.e. to protect the increasing amount of sensitive/important information being used and stored in electronic format. While security threats are becoming more dangerous, varied, and evolving, security requirements must be changed accordingly, and thus often getting more complex. In fact, taking into account security concerns while developing (already complex) systems makes the development process more stressful, error-prone and difficult. Security requirements are often tangled between functional requirements. Therefore, it is hard to integrate them properly into the traditional software development process. However, they are rarely taken into account at early stages of the development process [16]. Even though the complexity of systems (especially including security concerns) that has to be produced and maintained is continuously increasing, economic pressure reduces the development time and increases the frequency of demanded modifications. As a consequence, many security weaknesses, which have been exploited in practice, already made the headlines of the newspapers. The lack of efficiency of traditional methods for constructing secure systems are evident. All these issues urge for more timely, innovative, and sound methodologies for better supporting the development and maintenance of reliable secure systems.

*Model-Driven Security* (MDS) has emerged over more than a decade ago as a specialized *Model-Driven Engineering* (MDE) approach for supporting the development of secure systems. MDE has been considered by some researchers as a solution to the handling of complex and evolving software systems [10]. MDE leverages *models* and *transformations* as main artifacts at every development stage. By *modeling* the desired system and manipulating *models*, the level of abstraction is higher than code that brings several significant benefits, especially w.r.t. security engineering. *Firstly*, security concerns are considered (modeled, manipulated) together with the business logic from the very beginning and throughout the MDS development lifecycle. In this way, security requirements are considered early and implemented more properly in the resulting complete secure system implementation. *Secondly*, reasoning about the desired systems at the model level enables the application of formal methods such as model checking and model-based analysis to verifying security properties. Moreover, using models at a higher-level than the final target platform and independently from business functionality enables platform independence as well as cross-platform interoperability. *Thirdly*, MDS is productive, and less error-prone by leveraging on MDE automation provided by automated *model-to-model transformations* (MMTS) and *model-to-text transformations* (MTTs, code generation).

For more than a decade since MDS first appeared, there has been a considerable number of research papers published in this area. The main goal of this review is to examine existing literature work in MDS in a systematic way, classify and compare different approaches, underscore open issues, and suggest some potential research directions. The main contributions of this paper are: 1) identifying so far considerable main approaches in MDS; 2) revealing the current status of the key aspects in MDS like its application domains, its addressed security concerns, etc.; and 3) pointing out the

main limitations of current MDS approaches, and suggesting some potential MDS research directions.

The remainder of this paper is structured as follows. The objective of this systematic review, its research questions, review protocol, search strategy, and selection process are described in Section II. The evaluation criteria used to evaluate selected approaches are presented in Section III. Section IV discusses the detailed results after synthesizing data extracted from the selected approaches. Possible limitations to the validity of the review are pointed out in Section V. A comparison of our work with related work is given in Section VI. Section VII concludes the paper by summarizing the results, highlighting open issues, and giving some thoughts of future work.

## II. Our systematic review method

A systematic literature review (SLR) differentiates itself from traditional reviews by following a well-defined method to systematically identify, examine, synthesize, evaluate, and compare all available literature work relevant to a specific research topic [27]. Three main phases of a SLR process with their associated stages are briefly recalled from [27] as follows. In the description of each phase, we refer to the section in where we show how the stages of that phase have been actually planned and conducted in our review process.
*i) Planning the review*: Identification of the need for a review (presented in Section I); Commissioning a review (optional, skipped); Specifying the research question(s) (Section II-A); Developing a review protocol and evaluating it (Section II-B).
*ii) Conducting the review*: Identification of as many relevant publications as possible (Section II-C); Selection of primary studies and Study quality assessment (Section II-D1); Data extraction and monitoring (Section III); Data synthesis (the results of this review is presented and discussed in Section IV).
*iii) Reporting the review* in a technical report/publication.

### A. Research Questions

This SLR aims to answer the following research questions:

***RQ1: How are the existing* MDS *approaches supporting the development of secure systems?***

This question is further divided into the following sub-questions:
RQ1.1: What kinds of *security mechanisms/concerns* are addressed by these MDS approaches?
RQ1.2 : How do the MDS approaches *specify/model* security requirements together with functional requirements? Is there any tool that supports the *modeling* process?
RQ1.3 : How *model-to-model transformations* (MMTS) are leveraged and which MMT engines are used? Is there any tool support for the transformation process?
RQ1.4 : How *model-to-text transformations* (MTTS) are leveraged to generate code, including complete, configured security infrastructures? Which tools are used for the code and/or security infrastructures generation process?
RQ1.5 : Have any *case studies* been performed to evaluate the approaches? If yes, what results have been obtained? What other evaluation methods (other than case studies) have been applied to evaluate these approaches?
RQ1.6 : Which *application domains* have been addressed in MDS approaches?

***RQ2 : What are the current limitations of the existing* MDS *approach?***

***RQ3 : What are the open issues to be further investigated?***

After having the research questions, the most suitable search strategy can then be employed to identify relevant studies and extract the data required to answer the questions [14]. All the research questions and its next steps for conducting the full review are defined clearly in our review protocol.

### B. Review Protocol

One of the most important aspects for the success of a SLR is its well-designed review protocol. The review protocol serves as a concrete and formal scheme for conducting the SLR. By being well-designed and predefined, the review protocol ensures to reduce the possibility of reviewers' bias.

Our review protocol has been formally defined and some key parts of it (e.g. the search string) were piloted for several times before included in the final protocol. The protocol was initiated by one author and reviewed by the other authors. We strictly followed the protocol while conducting the real review. Most of the contents of our review protocol have been partially presented in Sections I, II, and III.

### C. Search Strategy

In this section, we present the search strategy that we applied to search for relevant MDS papers.

*1) Identification of a Search String :* Based on the research questions (Sect. II-A), we identified the search terms that can be used to form the search string, e.g. *model-driven*, *model-based*, *security*, etc. In fact, we divided our search terms into these categories: MDE (model-driven, model-based, model*), modeling (specify*, design*), transformations (transform, transformation, "code generation") and security.

To form the search string, we used a disjunction of the keywords of each term group and then used the conjunction of all the groups of terms. In order to make sure that our set of selected papers includes all papers that are referred to and relevant for this review, we had to refine our search string for several times. To be more specific, the search string is formulated as follows (but needs to be adapted for each search engine):
*( "model-driven" OR "model based" OR MDA OR* MDE *OR model* OR UML) AND ( specify* OR design* ) AND ( transform* OR "code generation" ) AND security*

*2) Online Databases for Scientific Literature:* We performed automatic search within five electronic databases (range of publication year: 2000-2012): IEEE Xplore[1], ACM Digital Library[2], Web of Knowledge (ISI)[3], ScienceDirect (Elsevier)[4], and SpringerLink (MetaPress)[5], using the search string we described earlier. Each time, the search string may need to be modified to fit the format requirements of the electronic database before applying it.

### D. Inclusion (Exclusion) Criteria and Selection Process

*1) Inclusion (Exclusion) Criteria:* MDS approaches for developing secure system vary a great deal as different security concerns are dealt with, and/or different models/model-driven techniques are leveraged. Therefore, it is absolutely necessary that we define thorough inclusion/exclusion criteria to select the primary studies that can answer our research questions. The following inclusion/exclusion criteria are used:

1. Papers not written in English are excluded. In fact, this criteria was already embedded in our search process where papers not written in English are filtered out.
2. Short papers, i.e. papers with less than 5 pages in IEEE (double-column) format or less than 7 pages in LNCS (single-column) format are excluded.
3. Papers irrelevant to MDE are excluded. For example, the papers addressing security problems without using MDE techniques are excluded.
4. Papers proposing model-driven approaches without dealing with any security concerns are excluded. For example, model-driven approaches for performance analysis are excluded.
5. When a single approach is presented in more than one paper describing different parts of the approach, we include all these papers, but still consider them as a single approach.
6. Papers with insufficient technical information regarding their approaches are excluded. For example, the papers that do not provide a detailed description on secure models, intermediate models (if any), blurry security notion, and transformation/composition techniques, are considered incomplete and are excluded.
7. Only papers using MDE with a "generative" perspective are selected, i.e. papers in which models are central artifacts through out the development. Papers using model-based techniques for only verifying/analyzing security mechanisms without a view for implementation code are excluded.

*2) Primary Studies Selection Procedures:* Table I shows the statistic of our selection process that is explained as follows. The papers found from each repository were divided among reviewers according to the repositories. 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 the paper could not help us to decide to include or exclude it, we further checked the paper's abstract. If the abstract of the paper could not help us to make a decision, we further skimming (and scanning 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 found. Those duplicated papers were included in the final set of selected papers for sure. For the other candidate papers, each was discussed by at least two reviewers. For some "border-line" papers that were not easy to decide by two reviewers, they are checked by all reviewers. We maintained a list candidate papers that are rejected, with reasons for the rejection, after discussion among reviewers.

Table I
SUMMARY OF SEARCH RESULTS AND THE SELECTION PROCESS

| 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 removing duplicates | | | | | | **93** |
| Finally selected | | | | | | **80** |

## III. EVALUATION CRITERIA & DATA EXTRACTION STRATEGY

In this section, we describe a set of key aspects of MDS that forms a so-called evaluation taxonomy of MDS. We derived our evaluation taxonomy from our research questions, and also based on the synthesis of evaluation criteria described in [25] and the evaluation taxonomy proposed in [24]. This taxonomy makes it easier to 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 discussed to show how we extracted data from the primary studies in order to answer our research questions.

**Security concerns/mechanisms**: In this dimension, we classify primary studies according to the security concerns/mechanisms that the MDS approaches are dealing with. Range of security concerns is broad, e.g. authorization, authenticity, availability, confidentiality, integrity, etc. We will count the number of papers addressing each security concern. So, it is possible to identify whether any specific

[1]http://ieeexplore.ieee.org/Xplore/home.jsp
[2]http://dl.acm.org/
[3]http://apps.webofknowledge.com
[4]http://www.sciencedirect.com/
[5]http://link.springer.com/

security topic areas that addressed by a relatively large number of MDS approaches.

**Modeling approaches**: Security concerns can be modeled separately or not from the business logic. In this work, we are interested in finding how the current approaches model security concerns that can be eventually enforced into the system. Primary studies can be classified by the paradigms of modeling, i.e. *Aspect-Oriented Modeling* (AOM) or non-AOM. In AOM approaches, security concerns are modeled in separate *aspect models* to be eventually woven (integrated) into the *primary model(s)*. Vice versa, in non-AOM approaches, security concerns are not modeled as aspects. That means security concerns can be modeled together with business logic in every place where they are needed. But, we also classify as non-AOM approaches where security concerns modeled separately (*separation of concerns*) from the business logic that can be integrated later into the system. For example, a non-AOM approach could (separately) specify an access control policy using a *Domain-Specific Language* (DSL)[6], and then transform and/or generate XACML[7] standard file for enforcing the access control policy. In other words, we would like to know relatively the percentage of non-AOM approaches compared to the percentage of "full" AOM/*Aspect-Oriented Software Development* (AOSD) approaches where security concerns are really modeled as AOM aspects. Furthermore, approaches are also classified by the modeling 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, behavioral, 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 will extract data w.r.t. MMTS in order to answer the following questions: How well-defined are the MMTS rules? How MMTS are implemented? Using which MMT engines (e.g. ATL[8], QVT[9], KERMETA[10], 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?

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

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

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

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

[10]www.kermeta.org

**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 (half) complete, configured security infrastructures. Data should be extracted to see the main purposes of using code generation techniques. Whether the whole system including security infrastructure are generated or just the security configuration, or just the skeleton of the system? Which tools are used for the code generation process?

**Application domains**: Approaches are also classified on the target application domains of the secure systems. Examples of application domains are information systems, Web applications, e-commerce systems, secure smart-card systems, embedded systems, distributed systems, etc.

**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[11] 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.

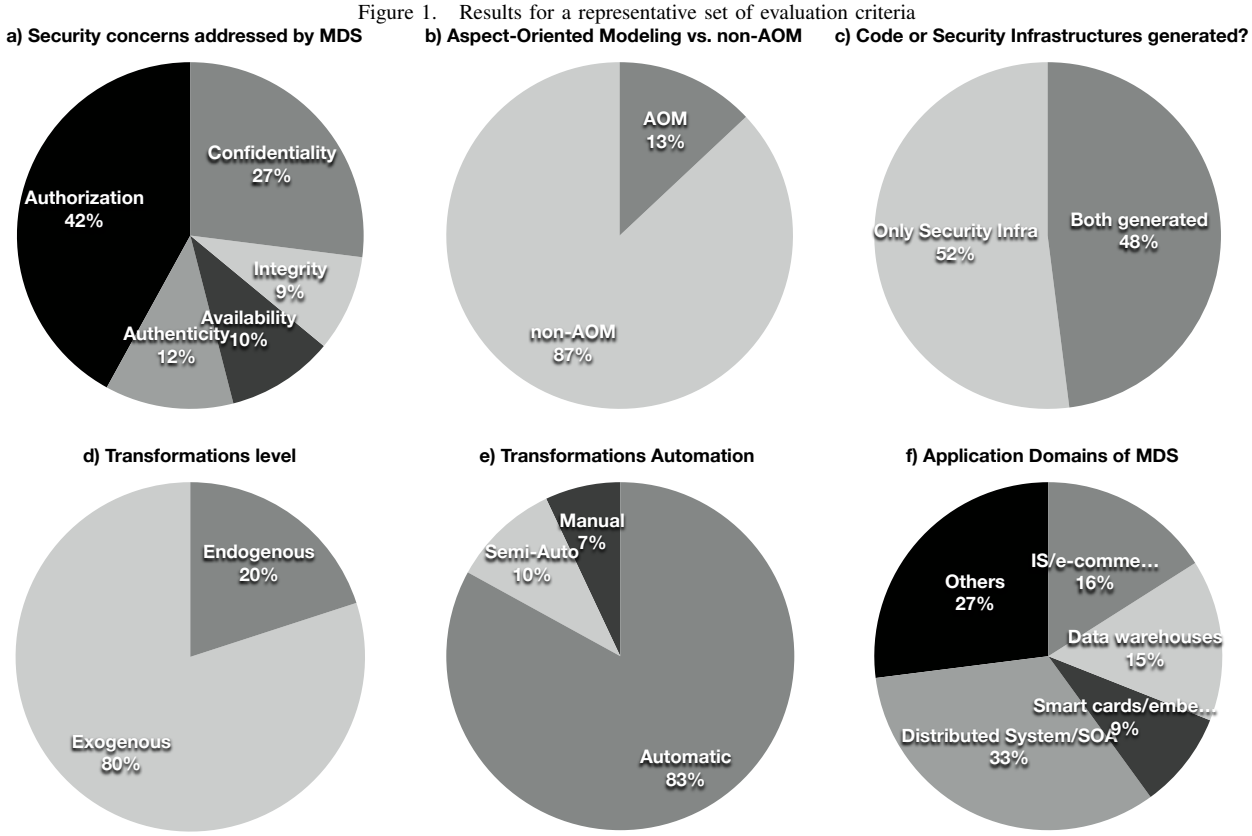
In order to answer the last two research questions, we reviewed the range of security topics, the scope of MDS research work and the quality of MDS research results to determine whether there are any observable limitations and open issues.

## IV. RESULTS

By extracting and synthesizing data according to the evaluation criteria presented in Section III, the research questions we raised from the beginning of this paper (in Section II) can now be answered. This section discusses the results after synthesizing data extracted from the final set of selected papers and provides different views on the results. Fig. 1 visualizes key results for a representative set of evaluation criteria. Table II shows absolute and relative values for all evaluation criteria, e.g. number of papers/criterion.

[11]http://www.mendeley.com/

Figure 1. Results for a representative set of evaluation criteria




### A. Results per Evaluation Criterion

**Security concerns/mechanisms**: Our SLR shows that *authorization*, especially access control (AC), is the security concern that is most addressed by MDS approaches with 42%. The second security concern in terms of attention is *confidentiality* (*secrecy*), with 27 %. Other security concerns, like *integrity*, *availability*, and *authentication* are, however, only rarely considered with about 10% for each concern. Fig. 1a shows these results in a pie chart. These first results are very interesting. Indeed, an open question is "why in MDS *authorization* and *confidentiality* got more attention?". A possible answer could be that MDS is a relatively young research area. It may also be that researchers that mainly work with MDE techniques first address *authorization* (e.g. AC) because it is closer to application logic and functional requirements than other security concerns. 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 communities in order 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. In this way, more research initiatives can focus on dealing with security concerns like *integrity*, *availability*, and *authentication*.

**Modeling approaches**: 13% of the papers discuss approaches that are based on AOM (Fig. 1b) so that security concerns are specified as aspects and eventually woven into primary models. Even though the remaining 87% 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. 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.

Table II shows that 79% of the reviewed papers used standard UML models and defined DSLs for security concerns using the profile and stereotype mechanism of the UML. 21% used other DSLs (e.g. [36]). Thus, it can be seen that defining DSLs is very popular to leverage MDE techniques for secure systems development. In other words, DSL plays a key role in MDS.

Security concerns are often modeled and analyzed with a DSL that is concern-specific. But, only a few of the 80 final papers have well-defined semantics for their languages so that these languages can be used for formal analysis. Only papers related to the UMLSEC approach (see Section IV-B) provide a formal basis for security analyses. This shows that further efforts are required to mature security-specific

modeling languages to foster analyses. Moreover, very few approaches propose to deal with multiple security concerns together like [37]. Most of them are specific to address only one security concern solely.

**Model-to-model transformations (MMTS) & tools**: Table II shows that 55% of the papers clearly mentioned MMTS while 31% did not use transformations, for example, because of a manual integration of security. Within the 55% of papers that described MMTS, 80% of them are exogenous transformations and most of these were used to transform PIMs to PSMs (Fig. 1d). Security concerns were modeled using DSLs for each concern to obtain PIMs that were transformed into PSMs, which can be refined into code. 83% of the MMTS described in the papers are automatic, 10% are semi-automatic (interactive) and only 7% are manual (Fig. 1e). Having automated MMTS is one of the key success factors of MDE [19] so this may also be the case for MDS. Within the papers that clearly describe MMTS, 25% of them also describe their implementation using standard transformation languages like ATL and QVT. 75% of the papers only describe the transformation rules without implementation details, or use other transformation languages like graph-based transformations or some specific compilers.

**Model-to-text transformations (MTTs) & tools**: Table II shows that 60% of the papers describe MTTs or the generation of code or security infrastructures. 40% of the papers did 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 only mentioned in future work. Comparing the purposes of MTTs, we can see in Fig. 1c that there are more approaches (52%) that only generate security infrastructure, such as XACML or security aspects code, than approaches that generate both code and security infrastructure (48%). A reason might be that some enforcement frameworks for authorization are platform-independent (e.g. using XACML) and those approaches only focus on generating XACML. Another reason could be that only few approaches support forward or even round-trip engineering for the whole development cycle of secure systems so that both functional code and security infrastructure can be derived. The results in Table II also shows that even if well-known MTT engines like XPAND were used in 40% of the papers that mentioned MTT, there are still many cases (60%) where other 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. POLARIS [29], for example, performs policy automata analysis and compilation. It includes a graphical interface for editing the automata, an analysis engine that checks for policy conflicts, and a code-generator that creates Java Card applets that implement the policy automata.

**Application domains**: Fig. 1f shows that the main application domains that have been secured by MDS approaches are distributed systems or SOA (33%), information systems or e-commerce (13%), and data warehouses (12%). The remaining papers do not clearly state a domain or claim to be generically applicable for different application domains, e.g. [37], [26].

Table II
RESULTS CLASSIFIED BY THE EVALUATION CRITERIA

| Evaluation criteria | | # papers | % |
|---|---|---|---|
| Security concerns | Confidentiality | **30** | **27** |
| | Integrity | **10** | **9** |
| | Availability | **11** | **10** |
| | Authenticity | **13** | **12** |
| | Authorization | **47** | **42** |
| Aspect-Oriented Modeling/AOSD | Yes | **10** | **13** |
| | No | **70** | **87** |
| Standard models | Yes(UML/UML profiles) | **63** | **79** |
| | Other DSLs | **17** | **21** |
| Type of models | Structural | **53** | **61** |
| | Behavioral | **26** | **30** |
| | Others | **8** | **9** |
| Transformations used | Yes | **44** | **55** |
| | No | **26** | **32** |
| | Unknown | **10** | **13** |
| Transformations level | Endogenous | **10** | **20** |
| | Exogenous | **39** | **80** |
| Transformations automation | Automatic | **24** | **83** |
| | Semi-automatic | **3** | **10** |
| | Manual | **2** | **7** |
| Standard Transformations | ATL/QVT | **20** | **25** |
| | Others/not mentioned | **60** | **75** |
| Code generation mentioned | Yes | **48** | **60** |
| | No | **32** | **40** |
| Code + Security Infrastructures | Yes | **23** | **48** |
| | Only Security Infrastructure | **25** | **52** |
| Code generation tools | Xpand/oAW | **14** | **40** |
| | Others | **21** | **60** |
| Application Domains | IS/e-commerce | **13** | **16** |
| | Data warehouses | **12** | **15** |
| | Smart cards/ embedded systems | **7** | **9** |
| | Distributed Systems/SOA | **26** | **33** |
| | Others | **22** | **27** |
| Type of validation | Case studies | **53** | **66** |
| | Others/not provided | **27** | **34** |

**Evaluation methods**: Most of the papers (66%) describe case studies mainly to illustrate the approaches. There are very few papers that provide an in-depth evaluation like [15], [41], and [9]. Therefore, we suggest that more effort should be put in evaluating MDS approaches, for example with empirical studies or benchmarks.

### *B. Primary* MDS *Approaches*

Altogether, the results show that there are currently several MDS approaches that have been used and discussed in multi-

ple publications. As we cannot discuss all MDS approaches of the 80 papers in detail we identified those approaches with the most publications. For the rest of this paper we will call an MDS approach a primary approach if there are at least 5 primary papers in our final set that discuss this approach. The primary MDS approaches are summarized as follows.

**SECUREUML** (e.g. [28], [6], [7], [8], [13]) is the approach aims at bridging the gap between security modeling languages and design modeling languages. The authors propose a UML-based language (UML profiles) with different dialects, which forms modeling languages (such as SECUREUML+COMPONENTUML) for designing secure systems. Their work mainly focuses on access control constraints based on RBAC in design models. Based on this approach, Clavel et al. show and discuss their practical experience of applying SECUREUML in [15]. The main limitations of SECUREUML are its sole focus on access control and its lack of support for formal analysis.

**UMLSEC** (e.g. [23], [22], [17], [18], [21]) is another well-known UML-based approach in MDS proposed by Juerjens et al. Security requirements, threat scenarios, security concepts, security mechanisms, security primitives can be modeled by using security-related stereotypes (UML profiles), tags, and security constraints. Thus, it is possible to formally analyze UMLSEC diagrams against security requirements w.r.t. their dynamic behaviors. Not like SECUREUML only focusing on authorization (e.g. access control), UMLSEC addresses multiple security concerns such as *confidentiality*, *integrity*, etc. But 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 big miss in UMLSEC. So far, its application domains are for developing secure information systems and secure embedded systems.

**SECTET** (e.g. [3], [4], [2], [12], [1]) is the work by Alam et al. that firstly aimed at securing web services by leveraging the Object Constraint Language (OCL) for specifying RBAC. 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. SECTET-PL is also used for modeling restricted (RBAC-based) delegation in Service Oriented Architecture. MMT and MTT are both carried out in a complete model-driven framework. SECTET mainly addresses RBAC as its security concern and focuses on generating security infrastructure (XACML), not all the source code.

**SECUREMDD** (e.g. [35], [31], [32], [33], [34]) is proposed for facilitating the development of smart card applications based on UML models. In SECUREMDD, UML class diagrams are used for modeling static aspects while UML sequence and activity diagrams are used for modeling dynamic aspects of a system. 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 w.r.t. the security properties of the system. The main limitations of SECUREMDD are its specific application domain and the lack of analysis for consistency between the UML models and the ASM model.

**Secure data warehouses (DWs)** are the motivation for the work of Villarroel, Soler et al. (e.g. [42], [39], [40], [38], [11]). Their approach also uses UML profiles for modeling security enriched PIMs as inputs for a model-driven framework to create secure DW solutions. Secure PIMs can be transformed to secure PSMs by a set of formally defined QVT rules. These PSMs can then be used for generating code with security properties. More recently, the above mentioned techniques for secure DW development are also leveraged in a reverse engineering style to modernize legacy DWs. One of the main limitations is that this approach is very specific for developing secure DWs.

### C. Result Summary and Discussion

The results suggest that more research work should focus on particular security concerns like *integrity*, *availability*, and *authentication*. There are considerable less papers tackling these security concerns than papers dealing with *authorization* and *confidentiality*. Very few selected papers propose a full AOM approach in which security concerns are specified as *aspects* and eventually *woven* into the primary models. However, many approaches still use *separation-of-concerns* methodology to specify security concerns separately from business logic and then enforced into the system at code level, e.g. security enforcement via generated XACML. It also can be seen from the results that UML are used popularly in MDS because of its standard and also UML profiles can be used to define DSLs for specifying security concerns. DSLs are necessary in MDS because the security concerns/mechanisms are often specific. Thus DSLs which are not UML profiles are also recommended, especially DSLs that can deal with multiple security concerns in the same system. An important remark is that more work should be done to have (DSLs) models with well-defined semantics of various security concerns. These models must be extensively, formally defined in order to enable the integration with automated analysis tools (based on well-established formal methods) and/or program synthesis tools. On the other hand, a tool chain (based on MMTS and MTTs) to derive from models to implementation code is also an important piece of future work.

MMTS and MTTs are widely used in MDS in order to improve the productivity of the development process. 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 (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. Last but not least, there is a lack of empirical studies for MDS approaches. More empirical studies should be conducted to fully evaluate MDS approaches.

## V. THREATS TO VALIDITY

There are several threats to validity that may affect the results of this review. In order to maximize 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 on full text. This limitation could affect the search results returned by ISI. To make our review more complete, we should have also conducted the manual search on relevant journals and proceedings of relevant conferences. Even though this step would lead to most of the papers that already found in the automatic search. To minimize 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). 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 from Springer Link repository that even got into our final selected papers for data extraction, e.g. [30], [22].

## VI. RELATED WORK

There are some related surveys in MDS, and only one systematic review [20] in this research area. In [24], the authors present a survey on MDS. They propose an evaluation based on the work of Khwaja and Urban [25]. The study revealed that approaches that analyze implementations of modeled 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 [24] is that it is not a systematic review. As a result, there are some well-known approaches that are missing in [24], such as SECUREUML [7].

In [5], 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 [5] is that it only considers the work around SECUREUML.

[20] is closer to our paper. It is also a systematic review on MDS. 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 standardization is key to achieve the objectives of MDD/MDA (which are increased portability and interoperability). However, [20] presents several drawbacks and differences from our paper. First, concerning the systematic review protocol, no evaluation criteria and data extraction strategy are given. Moreover, exclusion criteria are very specific to the research questions. Consequently, the authors exclude papers which do not support automatic code generation, e.g. UMLSEC papers. Finally, the authors exclude AOM approaches, because they consider that AOM does not consider security aspects as specific aspects (i.e. different from other aspects).

## VII. CONCLUSIONS AND FUTURE WORK

In this paper we presented a systematic literature review on model-driven approaches for developing secure systems. We described a rigorous review protocol as well as a search strategy providing an exhaustive list of relevant papers on MDS. We have picked out a final set of 80 papers from 10633 relevant papers after an extensive (5-step) selection and review process. From the final papers we extracted and summarized the data in order to answer our research questions.

Our results show that most approaches focus on *authorization* and *confidentiality* while only few publications address further security concerns like *integrity*, *availability*, and *authentication*. Most of the approaches try to separate security concerns from core business logic, but only a few weave security aspects into primary models. The UML profile mechanism is often used for the definition of security-oriented DSLs. This may also be one of the reasons why most security modeling languages lack a thorough semantic foundation, which is need not only for automated formal analyses. The reviewed papers provide only incomplete MDE tool-chains as we did not identify an integrated approach for the generation of functional code and security infrastructures. Most approaches discuss illustrative examples but lack in-depth evaluations, for example using common benchmarks or empirical studies. Altogether, our literature review shows that many MDS approaches are successful

for specific, isolated security concerns, but lack formality, automation, process-integration and evaluation.

To make our review more systematic and complete, we will manually search in journals and conference proceedings for MDS papers. The journals and conferences will be chosen based on relevance and impact in terms of the high impact index[12] and conferences rankings. Cross references and the latest publications from the main authors will be checked manually. To this end we will perform an additional backward search on the references in the identified papers in order to identify thematically related topics and publications. Finally, we will adapt our future search strategy according to the "Snowballing search strategy" [43].

## ACKNOWLEDGMENT

This work is supported by the Fonds National de la Recherche (FNR), Luxembourg, under the MITER project C10/IS/783852.

## REFERENCES

[1] B. Agreiter and R. Breu. "Model-Driven Configuration of SELinux Policies". In: *On the Move to Meaningful Internet Systems: OTM 2009*. Vol. 5871. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 887–904.

[2] M. Alam, J.-P. Seifert, and X. Zhang. "A Model-Driven Framework for Trusted Computing Based Systems". In: *Enterprise Distributed Object Computing Conference, 2007. EDOC 2007. 11th IEEE International*. 2007, pp. 75–75.

[3] M. Alam, R. Breu, and M. Breu. "Model driven security for Web services (MDS4WS)". In: *Multitopic Conference, 2004. Proceedings of INMIC 2004. 8th International*. 2004, pp. 498–505.

[4] R. Alam MAlam2006eu and M. Hafner. "Modeling permissions in a (U/X)ML world". In: *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*. 2006, pages.

[5] D. Basin, M. Clavel, and M. Egea. "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.

[6] D. Basin, J. Doser, and T. Lodderstedt. "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.

[7] D. Basin, J. Doser, and T. Lodderstedt. "Model driven security: From UML models to access control infrastructures". In: *ACM Trans. Softw. Eng. Methodol.* 15.1 (Jan. 2006), pp. 39–91.

[8] 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.

[9] B. Best, J. Jürjens, and B. Nuseibeh. "Model-Based Security Engineering of Distributed Information Systems Using UMLsec". In: *29th International Conference on Software Engineering, 2007. ICSE 2007.* 2007, pp. 581–590.

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

[11] C. Blanco et al. "Applying an MDA-Based Approach to Consider Security Rules in the Development of Secure DWs". In: *International Conference on Availability, Reliability and Security, 2009. ARES '09*. 2009, pp. 528–533.

[12] R. Breu et al. "Model-Driven Security Engineering of Service Oriented Systems". English. In: *Information Systems and e-Business Technologies*. Vol. 5. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, 2008, pp. 59–71.

[13] A. Brucker, J. Doser, and B. Wolff. "A Model Transformation Semantics and Analysis Methodology for SecureUML". In: *Model Driven Engineering Languages and Systems*. Vol. 4199. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, pp. 306–320.

[14] K.-Y. Cai and D. Card. "An analysis of research topics in software engineering - 2006". In: *Journal of Systems and Software* 81.6 (2008). Agile Product Line Engineering, pp. 1051–1058.

[15] M. Clavel et al. "Model-Driven Security in Practice: An Industrial Experience". In: *Model Driven Architecture – Foundations and Applications*. Vol. 5095. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 326–337.

[16] L. Cysneiros and J. Sampaio do Prado Leite. "Non-functional requirements: from elicitation to modelling languages". In: *Proceedings of the 24th International Conference on Software Engineering, 2002. ICSE 2002*. 2002, pp. 699–700.

[17] D. Hatebur et al. "Systematic Development of UMLsec Design Models Based on Security Requirements". In: *Fundamental Approaches to Software Engineering*. Vol. 6603. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, pp. 232–246.

[18] S. Houmb and J. Jürjens. "Developing secure networked Web-based systems using model-based risk assessment and UMLsec". In: *Tenth Asia-Pacific Software Engineering Conference, 2003*. 2003, pp. 488–497.

[12] Journal Citation Reports 2011

[19] 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.

[20] 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.

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

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

[23] J. Jürjens. “UMLsec: Extending UML for secure systems development”. In: *«UML»2002 – The Unified Modeling Language* (2002).

[24] K. Kasal, J. Heurix, and T. Neubauer. “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.

[25] 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.

[26] 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*. Vol. 1. 2006, pp. 317–324.

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

[28] T. Lodderstedt, D. Basin, and J. Doser. “SecureUML: A UML-Based Modeling Language for Model-Driven Security”. English. In: *«UML»2002 – The Unified Modeling Language*. Vol. 2460. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002, pp. 426–441.

[29] M. McDougall, R. Alur, and C. Gunter. “A model-based approach to integrating security policies for embedded devices”. In: *Proceedings of the fourth ACM international conference on Embedded software - EMSOFT ’04* (2004), p. 211.

[30] “Modeling Security Critical SOA Applications”. English. In: *Security Engineering for Service-Oriented Architectures*. Springer Berlin Heidelberg, 2009, pp. 93–119.

[31] N. Moebius, K. Stenzel, and W. Reif. “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.

[32] N. Moebius and K. Stenzel. “Model-Driven Code Generation for Secure Smart Card Applications”. In: *The 20th Australian Software Engineering Conference* (2009), pp. 44–53.

[33] N. Moebius, K. Stenzel, and W. Reif. “Formal Verification of Application-Specific Security Properties in a Model-Driven Approach Example : A Copycard Application”. In: (2010), pp. 166–181.

[34] N. Moebius et al. “Incremental Development of large, secure Smart Card Applications”. In: *md-sec2012.pst.ifi.lmu.de* (2012), pp. 1–6.

[35] 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.

[36] 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.

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

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

[39] 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.

[40] 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.

[41] 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*. 2007, pp. 829–836.

[42] R. Villarroel et al. “Using UML Packages for Designing Secure Data Warehouses”. In: *Computational Science and Its Applications - ICCSA 2006*. Vol. 3982. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, pp. 1024–1034.

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