An Exploratory Study of Why UMLsec Is Not Adopted

Shouki A. Ebad
*Department of Computer Science,
Faculty of Science, Northern Border University,
Saudi Arabia*

Keywords: UMLsec, Software Security, Model-based Software Engineering, Adoption.

Abstract: UMLsec is an extended UML-based secure modelling profile. It has been applied at the phase of the software design and architecture. Although it appeared over two decades ago and been integrated into some tools, how extensively it has been adopted or used by the software security community is questionable. This paper employs social science methodologies to fill this gap. The contribution of this study is to find the reasons affecting the UMLsec adoption by software practitioners and researchers and their proposals to increase this adoption. As a result, only 13% of the sample uses UMLsec. In addition, four problems preventing the use of UMLsec, (1) using a pattern-driven security methodology rather than UMLsec (2) agile supportability; agile process reduces the design and architecture documentation including UML diagram (3) UMLsec standardization and tooling is still questionable (4) the awareness and training on use UMLsec are weak. The study also presented proposals for UMLsec improvement, in particular (1) simplifying the notations to apply UMLsec in many fields (2) raising awareness (e.g., demonstrating practical examples to the interested people). The paper discussed the threats to the validity of the study and suggested open issues for future research.

1 INTRODUCTION

A number of methodologies have been proposed for secure software engineering such as SQUARE, Secure Tropos, and CLASP (Khan et al., 2021; Mohammed et al., 2017; Gopal et al, 2014). The ones that are based on the Unified Modelling Language (UML) have been the most frequently used at design and architecture development phase (Mohammed et al., 2017; Sebastian et al., 2020). This UML-based category includes different methodologies such as UMLsec, CORAS, SecureUML, and Gomaa-UML. The kind of category allows system developers to express and reason about security policies using UML. The point is to build a security-annotated architecture model first; the implementation is then derived from the model. Accordingly, UML-based security methodologies have been applied at the phase of design and architecture. A well-known UML-based security approach in the literature is a UMLsec, which is a UML profile that extends the existing UML diagrams with security requirements features Jürjens and Shabalin (2007). In particular, it introduces three extensions for security aspect: *stereotypes, tagged values, and constraints*. Such extensions add, in case of attaching to UML elements, three types of security-relevant information (1) Requirements on the system physical structure (e.g., «Internet» and «encrypted») (2) Requirements on the system logical structure (e.g., «secrecy», «integrity», and «critical») (3) Policies that system components are expected to follow, for instance:

- «secure links» to ensure that the physical layer follows security requirements on the communication
- «secure dependency» to ensure that a dependent component in the system architecture model keeps the security requirements to the component(s) it depends on.
- «abac» to define RABAC's elements (Role Attribute-Based Access Control) including roles and permissions and verify them against the system structure. Accordingly, UMLsec defines 21 stereotypes; Figure 1 (borrowed from Jürjens and Shabalin (2007) shows some of them together with associated tags and informal descriptions.

UMLsec was first proposed over two decades ago by Jan Jürjens, an active researcher in software security area (Mohammed et al., 2017). It has already been validated in several industrial applications such as

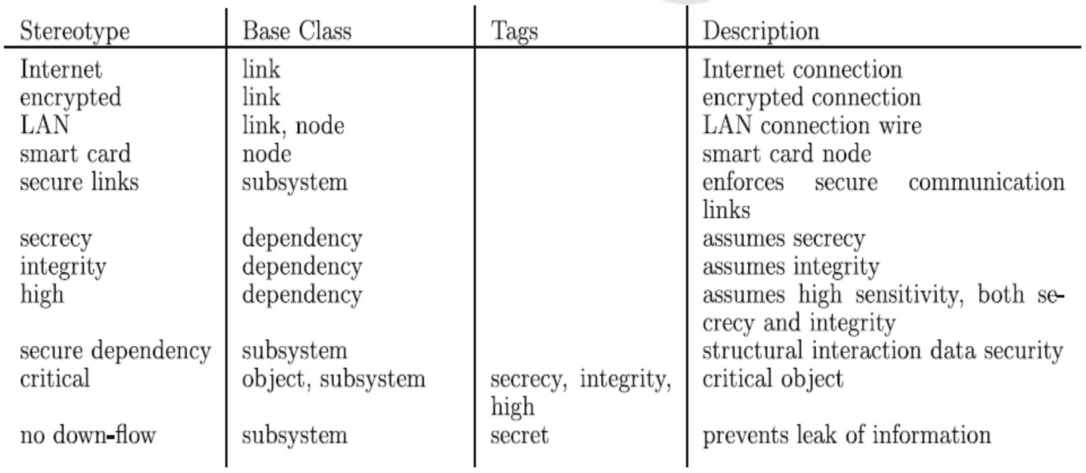
Ebad, S.
An Exploratory Study of Why UMLsec Is Not Adopted.
DOI: 10.5220/0010821400003120
In *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP 2022)*, pages 357-364
ISBN: 978-989-758-553-1; ISSN: 2184-4356

payment systems (Jürjens 2001) and telecommunication systems (Jurjens et al., 2008, Schneider et al., 2012, Sklavos et al. 2014). However, UMLsec adoption by the software practitioners and researchers is limited compared to those methodologies used in the coding phase (e.g., static analysis and dynamic analysis) (Mohammed et al., 2017). This study investigates the factors about why software security community may or may not adopt UMLsec. Particularly, the major contribution of this study is to identify the reasons affecting the UMLsec adoption by secure software practitioners and researchers and their proposals to increase this adoption.

2 LITERATURE REVIEW

Jürjens and Shabalin (2007) described a UML verification scheme supporting the development of automated requirements analysis tools for UML diagrams. They connected their scheme to CASE tools using XMI and allowed convenient access to this data and to the ordinary user. They presented plugins for verifying models defined using UMLsec. The framework allowed advanced users of the UMLsec approach to themselves implement verification procedures for the constraints of self-defined stereotypes. Gopal et al (2014) made a comparison among the popular security requirements methodologies: SQUARE, UMLSec, Secure Tropos and CORAS based on five criteria: coverage of CIA triad (confidentiality, integrity and availability), applicability to systems, stakeholders' views, the concept of asset identification, and software development phase. They then introduced a methodology (MAR(S)2) that incorporates all the important functions including requirements validation to produce profound and well-defined security requirements for critical infrastructure industrial systems like SCADA. Burger et al. (2015) used UMLsec and graph transformation to present a technique to support semi-automatic system co-evolution which responds to environmental knowledge evolution. The goal was to enable practitioners to interact more reliably to environmental changes and to ensure lifelong compliance of systems. To assess their approach, they used an open-source project, iTrust, as a case study. Sedaghatbaf and Azgomi (2016) used the Dempster-Shafer theory of evidence to formulate the uncertainties in input parameters of software security evaluation process and determine their effects on output measures. They specified parameters using the SecAM profile while attacks using UML diagrams (i.e., misuse case and mal-activity diagrams). UML/SecAM models were then converted into attack trees to estimate the probability of security breaches. The method was validated by a case study on an online marketing system.

Mohammed et al. (2017) conducted a systematic mapping study to determine the primary studies on the use of software security approaches in software development life cycle (SDLC). They identified 52 security approaches categorized into five categories, namely- ordered by the most frequency, 'vulnerability identification, adaption and mitigation', 'extended UML-based secure modelling profiles', 'software security focused process', 'non UML-based secure modelling notations', and 'secure requirements modelling'. Ramadan et al. (2017) presented a framework to manage security



Stereotype	Base Class	Tags	Description
Internet	link		Internet connection
encrypted	link		encrypted connection
LAN	link, node		LAN connection wire
smart card	node		smart card node
secure links	subsystem		enforces secure communication links
secrecy	dependency		assumes secrecy
integrity	dependency		assumes integrity
high	dependency		assumes high sensitivity, both secrecy and integrity
secure dependency	subsystem		structural interaction data security
critical	object, subsystem	secrecy, integrity, high	critical object
no down-flow	subsystem	secret	prevents leak of information

Figure 1: UMLsec stereotypes.

requirements from the stakeholder's perspective. They integrated two methodologies, SecBPMN2 and UMLsec via model transformation. The proposed framework was appropriate to render the early development phases of a case study, an air traffic management system, less error-prone and more systematic. Rehman et al (2018) combined the best aspects of four methods (UMLsec, CLASP, SQUARE, SREP) to develop a security requirements engineering framework for cyber-physical systems (CPS). The framework was evaluated using a case study of medical video chat. The framework is expected to help in determining the security requirements for CPS. Muneer et al. (2020) reviewed the exiting proposed notations to represent security aspects within a system including The Nine Principles and UMLsec. Based on theory of "how visual notations communicate?", they evaluated five notations (encryption, session maintained communication, content verification using hash, anonymous access, and authentication using sensory challenge).

Hu et al (2020) used semi-formal methods (i.e., UML/MARTE) and formal methods (i.e., Z/FSA) to propose a security modelling and verification framework of embedded software. They presented an extensible security model ZMsec (Z-MARTE security model), which extends Z with elements of MARTE and FSA to describe three aspects of software: security use-cases, static structures and dynamic behaviours. Sebastian et al. (2020) conducted a systematic literature review to investigate model-driven architecture on model-based methodologies in software engineering. As a result, UML was the most widely used modelling language. Arogundade et al. (2021) built an integrated UMLsec-based modelled system that converts UML diagrams to source code. The system incorporates four tools, Eclipse Mars with Papyrus modelling plug-ins, Eclipse Kepler, Java EE, and CARiSMA plug-ins. The integration was done by an application built with NetBeans. The proposed system was validated by modelling an e-government application from the class diagram to analysis and code generation. Khan et al. (2021) performed a systematic review about software security, different techniques, and models that have been proposed and designed in the context of the development of secure software. A relevant result is that "SecureUML and UMLsec" is still among the most cited secure software engineering methodologies.

Most works did not give the adoption of UMLsec much attention. Most researchers focused on construction a security model or framework for systems and automatically or semi-automatically implement/code it from that model (Burger et al. 2015; Ramadan et al., 2017; Jürjens and Shabalin, 2007; Arogundade et al., 2021). Some reviewed the existing approaches in software security (Khan et al. 2021; Mohammed et al., 2017; Muneer et al., 2020; Sebastian et al., 2020), and introduced a new approach (Gopal et al, 2014). Others performed the software security evaluation process from uncertainty analysis point of view (Sedaghatbaf and Azgomi, 2016). The others used UML to propose a new security framework for embedded software (Hu et al, 2020).

3 RESEARCH APPROACH

In this study, a questionnaire-based survey is developed to get information required for the research. The questionnaires are answered by the participants from software security community. The information collected are then organized in a form that can be processed in a qualitative way. Such approach is suitable in studying, for example, how a development technique or process has improved or degraded a specific entity (human, infrastructure, etc.), or why some designers prefer this pattern while others prefer another (Wohlin et al., 2012). Herein, we aim at studying why/why not UMLsec is adopted by secure software people.

3.1 Data Collection

The target people herein are practitioners and researchers working in several organizations. The emails of the participants were allocated through the articles published on different aspects of software security. Because many researchers use their personal email (e.g., Yahoo, Gmail, Outlook, etc.) or the email of their organization (e.g., Microsoft, HPE, etc.), it was difficult to find the countries of all participants. The URL of the questionnaire, together with a cover letter explaining the objective of the study, were emailed to the targeted sample. Invitations were sent to 31 arbitrarily selected practitioners and researchers in the area of software and computer security. Following the data collection, the responses were coded to enable them to be processed using computer. Microsoft Excel was used for the analysis purpose.

3.2 Survey Instrument

In spite of its suitability for our aim, surveys with several questions are boring for participants to fill out, and the data quality may decline (Wohlin et al., 2012). Therefore, the questionnaire consists of only two sections. The first section includes the participant's population profiles such as age, educational level, and occupation. The second section includes an open-ended question about using of UMLsec, problems preventing the adoption of UMLsec, and proposals to raise the degree of adoption as shown in Table 1.

4 RESULTS AND DISCUSSION

4.1 Descriptive Analysis

A total of 31 responses were received. The majority of participants' age is 36 and above years old, with 83.8%, this is followed by those who range between 26 and 35 years old with 16.1%, and no participants were below 26 years old (Figure 2(a)). Furthermore, most of the participants are Ph.D. holders with 87.1%, followed by M.S. holders with 12.9%, and no participants were undergraduate or diploma degrees' holder (Figure 2(b)). It is worth emphasizing that most of the participants (77.4%) have good experience; more than 10 years, 16.1% of them have 7-10 experience years, and 6.5% have 4-6 experience years (Figure 2(c)). In terms of occupation, 90.3% of the participants are working in academia, whereas 9.7% of them are working in industry (Figure 2(d)). Accordingly, 71.0% of the respondents were working on academic software projects, e.g. undergraduate projects, M.S. projects, and Ph.D. projects, 12.9% of them were working on industrial software projects, and the rest (16.1%) were working on other types of software projects (Figure 2(e)).

4.2 Challenges and Opportunities for UMLsec Adoption

In this section, we qualitatively analysed the participants' answers to the following open-ended question (Part 2 in the survey):

"Did you use UMLsec? If 'yes', (a) what are the problems preventing the adoption of UMLsec? (b) what are the proposals to increase the level of UMLsec adoption?"

Despite all participants have 4-11 experience years in working on different types of software projects as shown in Figure 2 (c and e), most of them (87%) indicated that they did not use UMLsec in their academic/research work because they never knew it before (Figure 2(f)). Only 13% of software practitioners and researchers adopt UMLsec in their work. This finding is shocking or particularly enlightening because UMLsec appeared over two decades ago (Jürjens, 2001) and several studies confirmed that it is a well-known method.

4.2.1 Problems Preventing the Use of UMLsec

(1) Agile Supportability:
Some respondents indicated that the new software processes do not require UMLsec; for example, agile methodology that is suitable for fast-moving business environment which needs rapid software processes differ from the traditional ones. The following is a sample of answers provided by different participants:

"Most of the agile methodologies don't focus on documentation anymore"

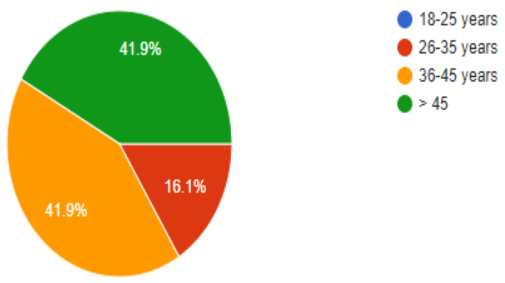
"Support agile approaches, be able to cope with systems which have not been designed based on models, but, most important…"

The common feature of agile methods is that design documentation is reduced or generated automatically by the programming environment used to implement the software (Sommerville, 2015). Figure 4 describes the agile process. It is clearly that specification, design and implementation activities are interleaved.

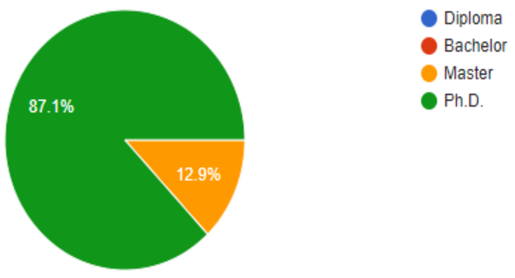
This design documentation minimization might come from the nature of communication among the software people; informal communications rather than formal meetings with written documents.

Table 1: Survey sections.

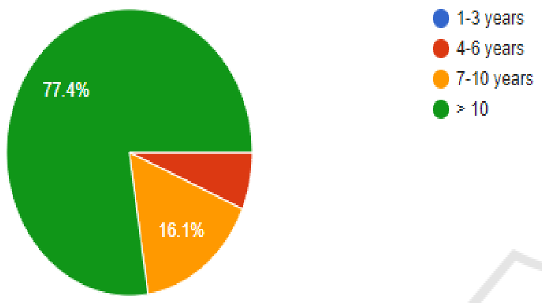
Part I Personal information		Part II: Open-ended question
Age	18-25, 26-35, 36-45, > 45	Did you use UMLsec? (Yes/No). If "Yes", then a) what are the problems preventing the adoption of UMLsec? b) what are the proposals to raise the level of UMLsec adoption?
Educational Level	Diploma, Bachelor, Master, Doctorate	
No. of Experience Years	1-3, 4-6, 7-10, >10	
Occupation	Academia/ Industry	
Type of Software Projects Working on	Industrial projects, academic Projects (undergraduate projects, master projects, and PhD projects.), and other research Projects	



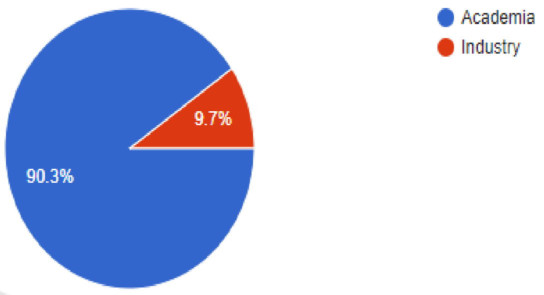
(a) Age.



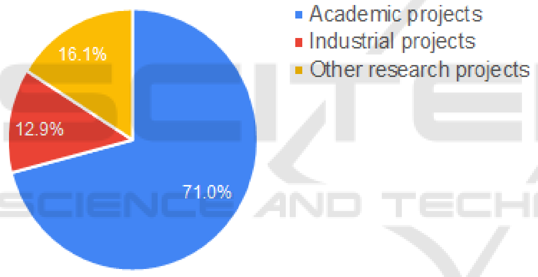
(b) Education Level.



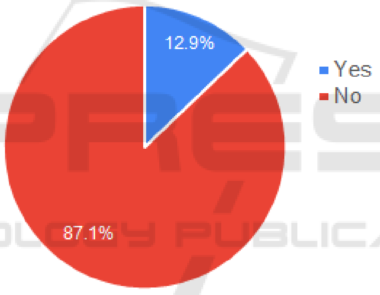
(c) Experience Years.



(d) Occupation.



(e) Types of Software Projects.



(f) Percentage of people who use UMLsec

Figure 2: (a-e) Respondents of Part I (personal information), (f) Respondents of Part II (open-ended question).

(2) ASE Adoption:
Some prefer to use other methodologies than UMLsec such as ASE, a pattern-driven security methodology (Uzunov et al., 2015):

“We have a better security methodology: ASE … “

The underlying idea behind the ASE method is taking encapsulation concept more steps further, by using patterns for the incorporation of security features and the threats modelling, and even as part of its process. ASE is particularly developed for distributed software (e.g., file sharing and collaborative editing) whereas UMLsec is generic in a way. However, UMLsec does not focus on requirements engineering activities such as elicitation, completeness, and validation (Gopal et al., 2014).

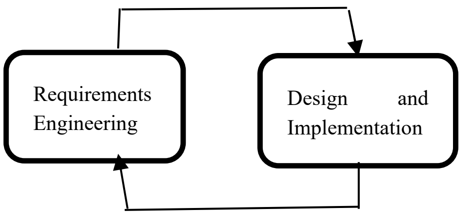


Figure 4: Agile development (Sommerville, 2015).

(3) Standardization and Tooling:
The adequacy of UMLsec notations seems to be questionable. According to the participants’ answers, this might come from different sources:

- The lack of tool-support for the analysis of UMLsec against difficult system security requirements especially those relevant to behavioural features (Muneer et al, 2020).

- The abstraction level of UML is very high; this causes issues in depiction of security requirements while designing software artefacts (Doan et al., 2004).
- The lack of theoretical bases for the use; this limits the UMLsec contribution (Muneer et al., 2020).

With no standardized rules and tooling applying to different areas, any methodology would not be dependable specifically for especially for certain systems such as health-related CPS (Rehman et al., 2018) because there is a need for tooling in creating a system architecture, and implementing it in terms of executable code. The following answers show how the people have a lack of trust to existing tools:

"UMLsec is not fully standard and not fully incorporated in existing tools. Hardly usable in practice. No concrete link with security control frameworks, risk models and existing security assessment practices"

"Also, there are threat modelling tools which are now being use more commonly"

"Stable tooling, general acceptance of UML and esp. model-based software engineering in industry"

"We need …tool to apply this new model across IT and Security from the beginning of software development"

Others stated that Google Cloud platform was enough for them to secure their applications:

"I'm using the tools and practices of the Google Cloud Platform [to secure a source code; part of my PhD work]"

Like similar platforms, Google Cloud Platform helps in protecting IT assets including software from evolving cyber threats by residing them in a safe place such as data centres around the world. This is insufficient to develop secure software as UMLsec aims; because residing assets in a safe place (from data centre to the device as the clouds do) may prevent from unauthorized access, but not stop other breaches such as illegitimate exportation or cost of the considerable inaccuracy in input parameters which may lead to misleading predictions.

Although UMLsec was applied with Eclipse Papyrus, a participant stated that this tool was not good enough:

"Papyrus is really slow and prone to errors"

The UML profile and Eclipse-based CARiSMA must be installed and integrated into the model in the case of applying the security concepts, and an automated security analysis, respectively.

(4) Lack of Experience:
Some participants suffered from human-related problems including absence of awareness and lack of training on using UMLsec. The following is a sample of their answers that show this factor:

"lack of experience and time [is a problem to prevent UMLsec acceptance]"

"We need the right people … to apply this new model across IT and Security from the beginning of software development"

"The awareness [is needed]"

This little widespread use of UMLsec comes from the development phase whose UMLsec works at. Other methodologies, which are used throughout SDLC (e.g., CLASP and SREP) or mainly based on requirements engineering process (e.g., SQUARE), but UMLsec works around the design/architecture phase. For this, UMLsec may not be a good choice to develop secure today's software like CPS (Rehman et al (2018). Because UMLsec works at the design/architecture level, conflicts between software quality attributes could occur, i.e., any software architecture improvement with respect to a quality attribute (e.g., security) may degrade another attribute (e.g., performance) (Sedaghatbaf et al 2016).

4.2.2 Proposals for Improving UMLsec Adoption

Correspondingly with the software security obstacles, the respondents suggested various proposals to improve the adoption of UMLsec which can be generalized into two groups. First, modifications on the current UMLsec notational features to be easier and more flexible. Second, awareness. The following is a sample of such proposals from their answers:

"Improve standardization and tool support. Relate the model to existing security assessment practices"

"make [notations] easily available recipes for use"

"[Let UMLsec] be part of sustaining any business for long run"

"I suggest an awareness campaign to folks in the College of Engineering"

"Application in different fields"

Representation of the constraints for security requirements plays a vital role in standardization because they are measured in terms of number and complexity of interactions among the requirements (i.e., the constraints become tighter). Additionally, achieving trust in UMLsec tools is not minor; the advice here is not only to make the notations simpler and more flexible but also easily applied in different areas including Internet of Things devices, which in turn may require adding new notations in an explicit manner. The awareness proposal can be achieved through different ways, for example (1) the development of UMLsec empirical examples and presenting them to the software community (2) investigating the possible of UMLsec solutions reuse in other projects.

4.2.3 Summary

Figure 5 summarizes the above discussion i.e., the problems and proposals relevant to UMLsec.

5 LIMITATIONS AND FUTURE WORK

The main threat to the study's validity is the ability to generalize and results. Although the sample size is relatively acceptable, the sample of individuals, heavily slanted towards academia, might not bring all important insights to the table. In general, findings are probably not easily generalizable because the participants were not really representative of software companies. In general, the literature lacks research in using a case study research strategy in software security due to confidentiality issues (Ebad et al., 2021). Accordingly, it would be useful to know just where UMLsec has succeeded and failed (e.g., types of applications that have benefited from such modelling). This would help to inform suggestions for increasing UMLsec adoption.

6 CONCLUSIONS

The study employs social science methods to investigate why software practitioners and researchers may or may not adopt UMLsec. The data were collected from 31 experts having a good experience in working on different software projects at academia or industry. The study analysed the obstacles of UMLsec adoption and provided different proposals for its improvement. The results indicated that there are four problems preventing the adoption of UMLsec, (1) use other security approaches such as ASE, a pattern-driven security methodology (2) supporting agile process minimizes design documentation including UMLsec (3) the lack of UMLsec standardization and tooling (4) the awareness/training related to UMLsec is weak. Together with obstacles of UMLsec acceptance, several proposals for its improvement are provided. The simplifying of security notations and raising awareness (e.g., demonstrating practical examples to the interested people) are the most critical proposals that need to be considered. Although the present paper yielded important findings, it also posits a limitation that should be taken in future attempts; most of the entire collected data were academic people. Thus, it is very imperative if future trials would take this limitation into account.

REFERENCES

Khan, R.A., Khan, S.U., Khan, H.U., Ilyas, M. (2021). Systematic mapping study on security approaches in secure software engineering, IEEE Access, Vol. 9

Mohammed, N.M., Niazi, M., Alshayeb, M., Mahmood, S. (2017). Exploring software security approaches in software development lifecycle: a systematic mapping study, Computer Standards & Interfaces, 50, 107–115

Gopal, T., Subbaraju, M., Joshi, R.V., Dey, S. (2014). MAR(S)2: methodology to articulate the requirements for security in SCADA. In *International Conference on the Innovative Computing Technology (INTECH 2014)*, 13-15 Aug. 2014, Luton, UK

Sebastián, G., Gallud, J. A., Tesoriero, R. (2020). Code

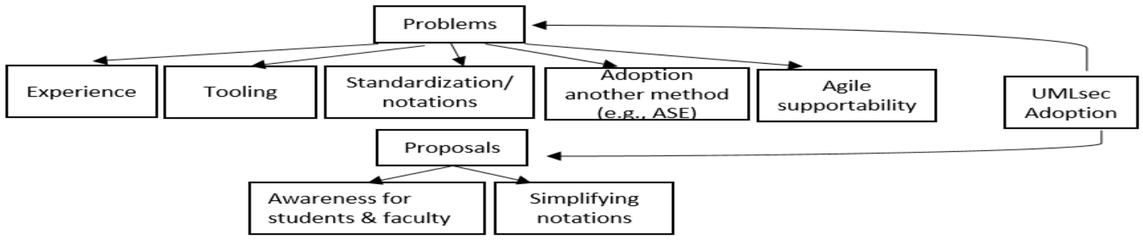

Problems
Experience
Tooling
Standardization/ notations
Adoption another method (e.g., ASE)
Agile supportability
UMLsec Adoption
Proposals
Awareness for students & faculty
Simplifying notations

Figure 5: Summary of problems and proposals.

generation using model driven architecture: a systematic mapping study. Journal of Computer Languages, 56, 100935.

Jürjens J., Shabalin, P. (2007). Tools for secure systems development with UML, International Journal on Software Tools for Technology Transfer, 9:527–544.

Jürjens J. (2001). Modelling audit security for smart-card payment schemes with UMLsec, In *16th International Conference on Information Security (IFIP/SEC"01)*, IFIP. Kluwer Academic Publishers, 2001, pp. 93–108.

Jurjens J., Schreck J., Bartmann P. (2008). Model-based security analysis for mobile communications. In *30th International Conference on Software Engineering (ICSE 2008)*. 10-18 May, 2008, Leipzig, Germany

Sklavos N., Hubner M., Goehringer D., Kitsos P. (2014). *System-level design methodologies for telecommunication*, Springer

Schneider, K., Knauss, E., Houmb, S., Islam, S., Jürjens, J. (2012). Enhancing security requirements engineering by organisational learning, Requirements Engineering Journal (REJ), vol. 17, no. 1, 35–56.

Ramadan, Q., Salnitri, M., Strüber, D., Jürjens, J., Giorgini, P. (2017). From secure business process modeling to design-level security verification. In *ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*, 17-22 Sept. 2017, Austin, TX, USA.

Jürjens, J. (2001). Towards development of secure systems using UMLsec. In *International Conference on Fundamental Approaches to Software Engineering*, LNCS 2029, pp. 187-200, Springer, Berlin, Heidelberg

Burger, J., Gartner, S., Ruhroth, T., Zweihoff, J., Jurjens, J., Schneider, K. (2015). Restoring security of long-living systems by co-evolution. In *IEEE 39th Annual International Computers, Software & Applications Conference*, 1-5 July 2015, Taichung, Taiwan

Sedaghatbaf, A., Azgomi, M.A. (2016). Quantitative evaluation of software security: an approach based on UML/SecAM and evidence theory, The ISC Int'l Journal of Information Security, Vol. 8, No. 2, 143-155.

Rehman, S., Allgaier, C., Gruhn, V. (2018). Security requirements engineering: a framework for cyber-physical systems. In *International Conference on Frontiers of Information Technology*, 17-19 Dec. 2018, Islamabad, Pakistan.

Muneer, S., Nadeem, M., Kasi, B., Yousaf, M.H. (2020). Evaluating the effectiveness of notations for designing security aspects. In *International Conference on Advanced Communications Technology (ICACT)*, 16-19 Feb. 2020, Phoenix Park, South Korea.

Hu, X., Zhuang, Y., Zhang, F. (2020). A security modeling and verification method of embedded software based on Z and MARTE, Computers & Security, 88 (2020) 101615.

Arogundade O.T., Onilede O., Misra S., Abayomi-Alli O., Odusami M., Oluranti J. (2021) From modeling to code generation: an enhanced and integrated approach. In Singh P.K., Polkowski Z., Tanwar S., Pandey S.K., Matei G., Pirvu D. (eds) Innovations in Information and Communication Technologies (IICT-2020). Advances in Science, Technology & Innovation (IEREK Interdisciplinary Series for Sustainable Development). Springer, Cham. https://doi.org/10.1007/978-3-030-66218-9_50.

Wohlin C, Runeson P,HostM, OhlssonMC, Regnell B,Wesslen A. (2012). *Experimentation in software engineering*, Springer, Berlin/Heidelberg, Germany.

Anton V. Uzunov, Eduardo B. Fernandez, Katrina Falkner (2015). ASE: a comprehensive pattern-driven security methodology for distributed systems, Computer Standards & Interfaces, 41, 112–137.

Sommerville, I. (2015). *Software engineering*, Pearson, England, 10th edition.

Doan, T. Demurjian, S. Ting, T., Ketterl, A. (2004). Mac and uml for secure software design. In *Proceedings of the 2004 ACM workshop on Formal methods in security engineering*. ACM, 75–85.

Ebad, S.A., Darem A.A., Abawajy J.H. (2021). Measuring software obfuscation quality– a systematic literature review, IEEE Access, vol. 9, pp. 99024-99038.