International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 12 – No. 9, December 2017 – www.ijais.org

# Modeling Security Requirements: Extending SysML with Security Requirements Engineering Concepts

Ilham Maskani
Team Architecture of Systems LISER-Laboratory
ENSEM, Hassan II University BP 8118
Oasis, Casablanca, Morocco

Jaouad Boutahar, Souhaïl El Ghazi El Houssaïni
Systems, Architectures and Networks Team,
EHTP BP 8108
Casablanca, Morocco

## ABSTRACT
In Security Requirements Engineering, many approaches offer different ways to model security requirements. This paper presents a model that can be used in conjunction with any of the former approaches. The model is an extension of SysML requirements diagrams that adds concepts from Security Requirements Engineering: Stakeholder, Goal, Asset and Risk. The proposed model is illustrated by applying it to a telemedicine system.

## General Terms
Requirements engineering; Security; Security Requirements; Software modeling

## Keywords
Requirements modeling; Security Requirements Engineering; SysML Extension

## 1. INTRODUCTION
Security Requirements Engineering (SRE) is a subject gaining more and more interest in research. In previous work[1], we did a thorough analysis of existing security requirements engineering approaches. We proposed the outline for a comprehensive approach (hereafter referred to as CompASRE), incorporating the strengths and best practices found in existing approaches, and filling the gaps between them. One of those gaps was the modeling of security requirements. It was found that SRE approaches are highly heterogeneous in the way they go about eliciting, modeling and validating requirements. Goal Oriented approaches use graphs containing goals, agents and relationships between these elements. Others, inspired by Model Driven Engineering (MDE) approaches, use models based such as UML (Unified Modeling Language) use cases. Some SRE approaches do not offer any modeling activity to represent their requirements and choose to do so textually. Another major drawback of SRE approaches is the lack of a seamless transition between the requirements engineering phase and the design phase, i.e. how to link requirements to design models? This is what motivated the choice of the SysML modeling language, which offers a permanent link with all later phases of the Software Development Life Cycle (SDLC). In this paper, we present an extension of SysML that incorporates the main SRE concepts in order to model security requirements. The aim is not to just add yet another model on top of existing models, that's specific to only our CompASRE approach. In contrast, the aim is to introduce a model that is comprehensive yet flexible enough to be used by other SRE approaches. As a combination of SysML requirements diagrams and SRE concepts, the presented model makes the most of their benefits. It offers traceability of requirements (to the higher concepts such as stakeholders and goals) and creates a bridge between the requirement engineering phase and later phases. This paper is structured as follows: Section 2 introduces standards and common models in the field of requirements modeling in general, especially SysML. Section 3 focuses on security by explaining the main SRE concepts, their importance and how they are used when modeling security requirements. In section 4, the SysML extension is presented after explaining its rationale. The presented model is illustrated with a Telemedicine case study. Finally, related work is presented in section 5.

## 2. REQUIREMENTS MODELING
In this section, existing standards and common languages are examined for requirements modeling in general.

## 2.1 SysML
SysML[2] (System Modeling Language) is a modeling language whose purpose is to model the system as a whole, in contrary to modeling only the software part. It's an extension of UML, so both are based on the same meta-model, the OMG Meta Object Facility[3]. As a result, SysML kept (with or without a change) the principal diagrams such as use cases, sequence and activity diagrams, and eliminated some diagrams that were too specific to software such as objects diagrams (the part of UML kept in SysML is called UML4SysML). The most notable additions to UML are the requirements diagrams and the parametric diagrams, the later aimed at modeling quantitative elements such as performance. Since March 2017, SysML became an ISO standard ISO/IEC 19514:2017[4].

### *2.1.1 SysML Requirements Diagrams*
SysML has a strong focus on requirements. Requirements diagrams allow modeling functional and nonfunctional system requirements, along with relationships between requirements. In SysML, a requirement specifies a capability or condition that must (or should) be satisfied. It's expressed by an identifier and a descriptive test. There are operators used to model relationships between requirements. The ***"contain "*** relationship is to decompose a requirement into one or many sub-requirements. "***Derive"*** is to express the dependency of a requirement on the completion of another requirement. A very important feature of SysML is that a requirement can be linked to other elements used by SysML during later phases of the development lifecycle, mainly the design and implementation phases. The **"satisfy"** relationship can link a requirement with a design element (such as a use case, sequence diagram...) showing how that requirement will be satisfied. ***"Refine"*** links a requirement to one or more model elements that describe the requirement in further detail. ***"Verify"*** links a requirement to a test case.

## 2.2 Goal-Oriented models

Goal-Oriented Requirements Engineering (GORE) has been a shift from traditional requirements engineering as it introduces concepts such as goals and agents to deal with more and more complex software systems and their environment. They have the advantage of capturing software specifications for the system and its environment. In GORE, a requirement is "a goal whose achievement is the responsibility of a single software agent". Thus, goals are criteria to achieve requirements completeness and pertinence. Goal modeling and refinement allow requirements traceability. Literature offers many GORE approaches, each offering a set of graphs to model requirements.

### *2.2.1 KAOS*

KAOS (Keep All Goals Satisfied)[5] is a GORE approach where requirements are documented using a goal tree, with strategic goals as the root and requirements assigned to a specific agent as terminal leaves. Once an initial set of goals and agents are identified, goals are refined using AND/OR decomposition into soft goals until each terminal goal is realizable by an agent. KAOS offers 4 graphs, the main two graphs being (1) the goal model that represents hierarchical goals and (2) the responsibility model the represents agents and the goals assigned to each.

### *2.2.2 I* framework / Tropos / GRL / URN*

The i* modeling framework[6] is an agent-oriented and goal-oriented modeling framework. It relates goals to the organization context (agents). Models offered by i* are (1) the strategic dependency model which shows relationships between actors, and how they rely on each to achieve goals, and (2) the strategic rationale model that shows the interests of the agents and links between tasks and goals. That second is a main feature of i* as it allows to captures tradeoffs, which means it allows reasoning between two alternative solutions for the system to be. Requirements are expressed in form of actors' properties and relations.

I* is adopted by Tropos [7], an agent-driven software development methodology, with strong focus on the early phases of the SDLC as it includes an early requirements phase and a late requirements phase.

GRL (Goal-oriented Requirement Language)[8] is a language for supporting goal-oriented modeling and reasoning of requirements, based on the i* framework. GRL uses the concepts of intentional elements (goal, softgoal, resource and task), links and actors.

URN (User Requirements Notation) [9] is an ITU standard (Z.151 (10/12)) intended for the elicitation, analysis, specification, and validation of requirements. It's composed of two complementary languages: one for capturing goals provided by GRL (by transitivity based on i*) and one for capturing scenarios provided by the Use Case Map (UCM) notation.

### *2.2.3 Techne*

Techne is a formal abstract language to model requirements[10]. It means that it's a basis upon which new requirements modeling languages can be built. Techne creates r-nets (Requirements nets). A Techne-based language will then visually translate these nets into graphs, with a chosen notation. An advantage of Techne is that it can better capture priorities between requirements. Additionally, it offers transformation patterns between i* and Techne.

## 3. SECURITY REQUIREMENTS ENGINEERING

This section presents the main SRE concepts, their definition and use in a SRE. It also presents how security requirements are modeled in SRE approaches.

## 3.1 SRE concepts

Prior to any modeling activity, security requirements have to be elicited. From our previous study (9 approaches were studied, referenced in the introduction for further details), it was found that in order to elicit requirements, any SRE approach uses a different set of the same concepts. These concepts are drawn from both the fields of security and requirements engineering. All 9 approaches include identifying "goals", 7 of them identify threats, 6 of them identify stakeholders and 4 of them identify assets and risks. It seems logical that any given model to represent security requirements will have to use these concepts. Table 1 offers a definition of these concepts, which is based on the ISO/IEC 27000:2016 vocabulary[11]. Those are the definitions that CompASRE and the presented model are based on.

**Table 1. SRE concepts definitions**

| Concept | Definition | Alternate labels |
|---|---|---|
| Stakeholder | Person or organization that can affect, be affected by, or perceive themselves to be affected by a decision or activity. Some approaches include other systems that have an interest in the IS. We include also internal software agents to whom a goal will be assigned. | Actor, client, agent |
| Asset | Anything that has value to the organization, its business operations and their continuity, including Information resources that support the organization's mission (Data). | Information, Resource, Object |
| Goal | A Security objective that must be achieved by the system to be | Objective |
| Risk | Potential that threats will exploit vulnerabilities of an information asset or group of information assets and thereby cause harm to an organization | |
| Requirement | Need or expectation that is stated, generally implied or obligatory. Requirements are low level details of goals. | Goal, objective |

As an illustration of how these concepts are used in SRE, figure 1 shows the outline of CompASRE. The scope of this article is limited to the modeling activity (referred to below as 'Format requirements'). It supposes that the previous steps of

CompASRE or any other SRE approach have already been conducted to extract security requirements.

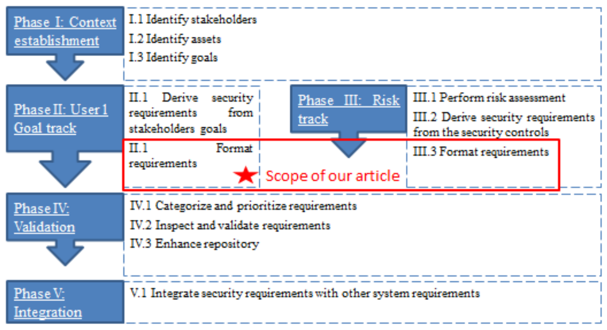
Phase I: Context establishment
I.1 Identify stakeholders
I.2 Identify assets
I.3 Identify goals
Phase II: User1 Goal track
II.1 Derive security requirements from stakeholders goals
II.1 Format requirements
Phase III: Risk track
III.1 Perform risk assessment
III.2 Derive security requirements from the security controls
III.3 Format requirements
Scope of our article
Phase IV: Validation
IV.1 Categorize and prioritize requirements
IV.2 Inspect and validate requirements
IV.3 Enhance repository
Phase V: Integration
V.1 Integrate security requirements with other system requirements

**Fig 1: Outline of CompASRE**

## 3.2 Requirements modeling in SRE

This section describes the models used in SRE for the requirements modeling activity. Proposed models for security requirements are adaptations of common requirements modeling languages to better suit the field of security. It is to note that not every SRE approach offers requirements modeling (i.e. SQUARE [12], MSRA[13] , SREF[14]), and choose to describe requirements textually. A modeling activity can be included without specifying the model itself (Holistic SRE Framework [15]). As for the approaches that do define a specific model for security requirements, the models used are either based on GORE or based on UML extensions.

### *3.2.1 GORE Based*

Many SRE approach take inspiration from GORE approaches. As a result, their models also use GORE models as described in section 2.2. Van Lamsweerde suggests elaborating security requirements using KAOS intentional anti-models based on threats[16]. First, security goals are modeled. Then, from the former model, an anti-model is derived based on threats. Finally, from both former models countermeasures are derived and security requirements are defined. The i* framework is used by Secure Tropos[17], an extension of Tropos that deals with security. The STS approach [18] also uses the i* modeling framework. Security requirements are specified, via the STS-ml requirements modeling language, as contracts that constrain the interactions among the actors.

### *3.2.2 UML Based:*

In the Security Requirements Engineering Process (SREP)[19], elicited security requirements are modeled using UMLSec[20], or expressed as security use/misuse cases or as plain text. But not all requirements are modeled, and not in a unified way. Security requirements are then documented in a *Security Requirements Specification Document* which will be refined in subsequent iterations of the SREP process. In the MOSRE process proposed in [21], UMLsec and SecureUML[22] are suggested for security based modeling of the functional requirements. UML based models have not been considered in our aim to model requirements. They are better suited for modeling at the design phase considering the advanced stereotypes used by UMLSec (fair exchange, secure links, no down-flow…) that already discuss architectural elements or mechanisms. As for SecureUML, it focuses only on modeling Role Based Access Control (RBAC) and authorization constraints.

## 4. ENHANCING SYSML WITH SRE CONCEPTS

### 4.1 Motivation

The purpose is to enhance SysML requirements diagrams with the concepts taken form GORE and other SRE approaches, which are goals, stakeholders, assets and risks. The motivation for such a combination is:

- SRE concepts, mainly drawn from GORE approaches offer great traceability. Each requirement would be matched with the stakeholder who expressed it, the security goal and asset it covers and the associated risk. This will help when categorizing, prioritizing and validating requirements, and at later phases of the SDLC when managing requirements.
- SysML requirements diagrams can be linked to later diagrams and model elements used by SysML during later phases of the SDLC, mainly the design and implementation phases. For example, a requirement can have a « verify » relationship with a use case that shows how that use case satisfies that requirement. Such a link isn't guaranteed when using GORE approaches to model requirements. This assures that the goals expressed by stakeholders will be without a fault taken into account at later phases.
- SysML being an UML extension, it will be easier to adopt by practitioners due to its familiarity. This is even more relevant for existing SRE approaches already using UML.

### 4.2 Extension mechanism

In order to achieve the purpose, the stereotype extension mechanism offered by UML is used. As an example, SysML requirements diagram is itself a stereotype of the class diagram, adding the properties 'Id' and 'Text', as shown in figure 2.

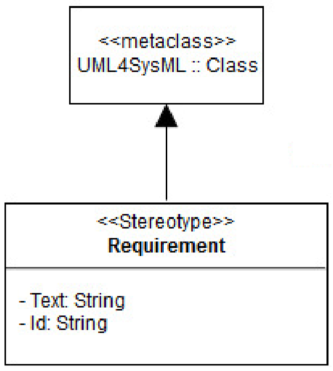

**Fig 2: Basic SysML requirements diagram meta-model**

A stereotype can extend a metaclass or another stereotype. So, to create our model, the "Requirement" stereotype is extended with a new stereotype « Security Requirement », and SRE relevant properties are added to it.

### 4.3 Proposed extension

Figure 3 shows the stereotype created for security requirements. SRE concepts were added as properties. The values of these properties are the stakeholders, goals, assets and risks as identified during the early phases of requirement elicitation.

- ExpressedBy: the stakeholder 'S' who expressed the goal that the requirement covers. Stakeholders can be an internal or external entity such as users, a single part of the system or an external entity
- GoalCovered: the goal 'G' that the requirement covers.
- AssetCovered: The asset 'A' that the requirement covers.
- RiskCovered: the risk 'R' that is mitigated through this requirement

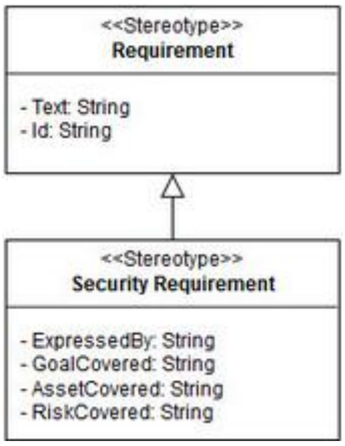

**Fig 3: Security Requirement stereotype**

Consequently, after eliciting security requirements, each requirement will be uniquely identifiable and expressed as follows: SecReq = {ID; S; G; A; R}

The text property can still be used for the sake of ease of reading but is not necessary to identify and express a requirement.

## 4.4 Case study

### *4.4.1 Telemedicine*

In this section, in order to illustrate our model the example of a Telemedicine System is used. A Telemedicine System is a web or mobile application that allows patients, among other features, to have a virtual consultation with a doctor via a video-telephony technology. When first logging in, the patient creates his profile providing a brief medical history. Then, he can make an appointment with one of the registered doctors. A record is created for each consultation containing doctor's consultation notes, images uploaded by the patients, prescriptions, lab results, etc … Such personal and medical data is very security sensitive. Research in telemedicine stretches the importance of security to ensure the adoption of such systems and the quality of the care provided [23]. Furthermore, a systematic review reported security concerns such as the use of poor encryption standards and proprietary protocols for communication without proof of security[24]. For this particular example, we will focus on the integrity of the patient's health records.

### *4.4.2 Modeling Integrity*

When eliciting security requirements, stakeholders, assets, goals and risks are identified. Below is a subset of the identified elements relevant to health records integrity.

- Stakeholders:
  - Patient = P
  - Doctor = D
  - System's Editor = E
  - Software = S
  - Software's Administrator = A
- Assets:
  - Health records = HR
  - Patient's personal information = HR1
  - Patient's initial medical history = HR2
  - Patient's current or future medical information= HR3
- Goals:
  - G1: Guarantee 'HR' updates integrity
  - G1.1: An 'HR' update can only be done by the patient or an authorized doctor
  - G1.1.1: An authorized doctor is a doctor who conducted a consultation on a patient, or a doctor appointed by the patient as the primary physician.
  - G1.1.2: A patient can only update his personal information 'HR1' and initial medical history 'HR2' of his own HR.
  - G1.1.3 A doctor cannot update the 'personal information' and 'initial medical history' information, and can update all other information 'HR3'.
  - G1.2: HR is to be archived after each update for later retrieval.
  - G1.3: For each HR update, the system should record who did the update, and when it was done.
- Risks:
  - R1: update of the HR by unauthorized third party using an authorized doctor identity
  - R2: update of the HR by unauthorized third party using the patient's identity.

Figure 4 shows the resulting requirements diagram. Relationships between requirements used are the "contain" and the "satisfy" relationships, explained in section 2.1.1.

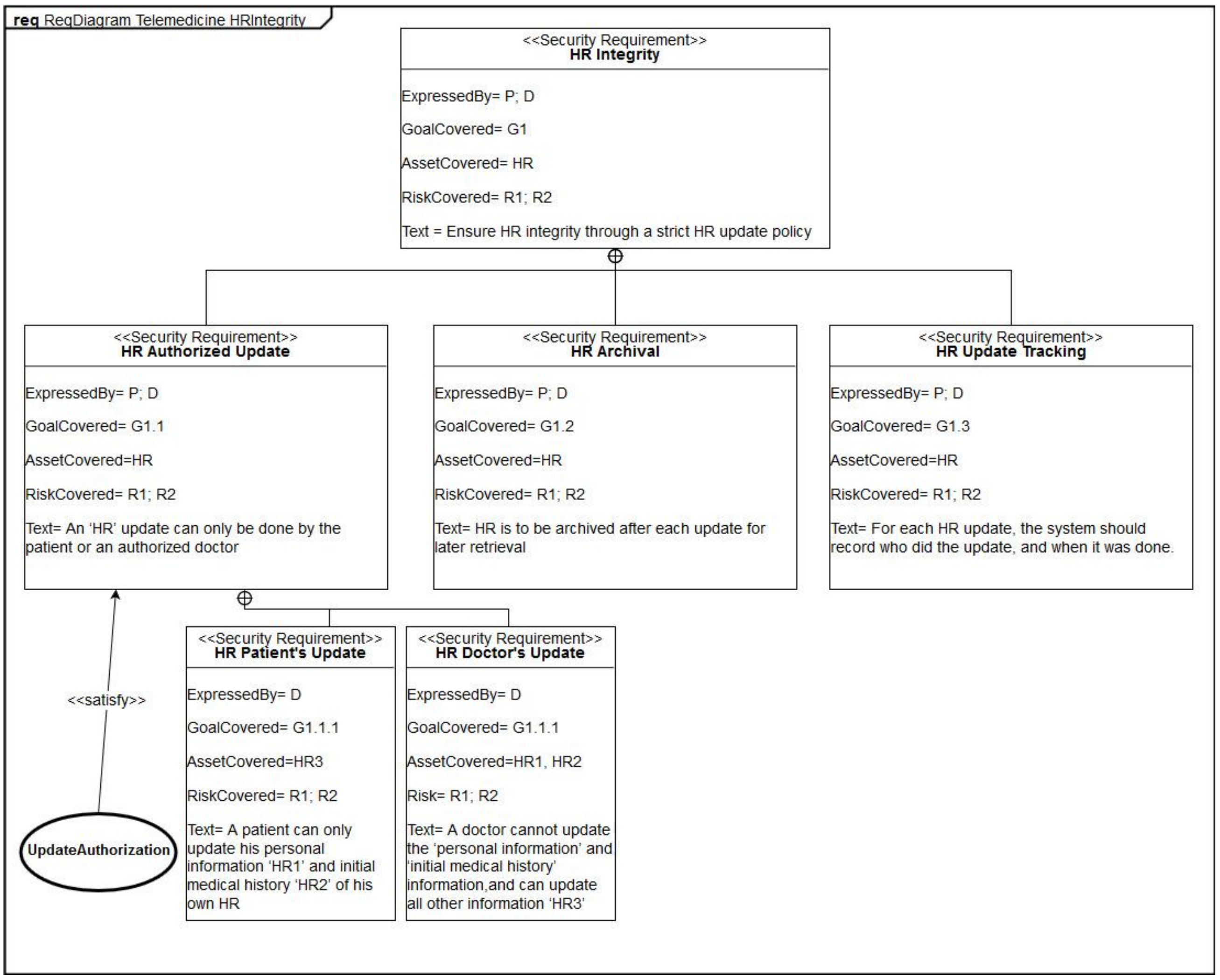

**Fig 4: Extended requirements diagram for HR integrity**

Another useful feature of SysML requirements diagrams is that they can be shown in a graphical, tabular or tree format. A tabular format has the advantage of being familiar to non-model users, when dealing with stakeholders for example. The requirements properties are shown as columns, so it will be highly useful during categorization and validation steps of requirements engineering where we can focus on requirements by stakeholder or risk for example.

### 4.5 Compatibility with other SRE approaches

A strong point of the proposed model is that it can be used with any other SRE approach that offers no specific way to model its requirements. For it to be usable by other SRE approaches, some requirements properties are kept optional as not every approach will cover all these elements. For example, SQUARE methodology does not identify stakeholders, so the property 'ExpressedBy' cannot be used in conjunction with SQUARE. As for the SRE approaches that already have their own models, our model can be complimentary to theirs, especially the ones that already use UML models such as use case or misuse case diagrams.

## 5. RELATED WORK

This section examines other works that use SysML with other SRE concepts. Laleau et al. [25]combined SysML with the B formal method. They extended SysML with the KAOS goal model and gave rules to derive a formal B specification from this goal model. This combination itself was not the objective of the paper. Their objective was to derive automatically a formal specification from an informal one. Moreover, they do not address security requirements. In [26], the authors present SysML-Sec, a Framework for design and development of secure embedded systems. It offers an extension of SysML for requirements (system analysis), design, and system validation. It's specific to embedded systems as it highly takes into account the correlation between software and hardware requirements. They elicit security requirements by identifying threats and vulnerabilities and doing a risk assessment. But they do not keep track of stakeholders or goals expressed by them. Another drawback is that it has a too strong emphasis on vehicular embedded systems, and the coupling between hardware and software engineering. [S08] used SysML requirements diagrams, requirements tables and use cases to model requirements of a road traffic management system.

## 6. CONCLUSION & FUTURE DIRECTIONS

In order to better model security requirements, the purpose of this paper was to present an extension of the SysML requirements diagram by adding concepts drawn form Security Requirements Engineering. This choice was dictated by the benefits offered by both SysML requirements diagrams and the use of SRE concepts. As a result, the presented model maintains the main advantages of SRE such as requirements completeness and traceability to goals and stakeholders. It enables to visualize risk coverage by linking requirements to the related risks. As an extension of SysML requirements diagram, our model guarantees to fill the gap between the requirements engineering phase and later phases such as design, implementation and test. It also benefits from UML familiarity to encourage its use. Last, but not least, it has has the major advantage of being compatible with other SRE approaches or complementary to other SRE models.

Future directions for our research are to validate the proposed CompaSRE approach and model by applying them to a concrete security sensitive system. Their benefits will have to be evaluated by comparing them to other SRE approaches. The model's usability and comprehensiveness will have to be proved by using it with a different approach. It's also planned to develop a tool to support CompASRE, especially the modeling step.

## 7. REFERENCES

[1] I. Maskani, J. Boutahar, and S. EL Ghazi El Houssaïni, 2016, "Analysis of Security Requirements Engineering : Towards a Comprehensive Approach," *IJACSA Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 11, pp. 39–45, Nov. 2016.

[2] "What is SysML? | OMG SysML." [Online]. Available: http://www.omgsysml.org/what-is-sysml.htm. [Accessed: 14-Nov-2017].

[3] "About the OMG System Modeling Language Specification Version 1.5." [Online]. Available: http://www.omg.org/spec/SysML/1.5/. [Accessed: 14-Nov-2017].

[4] "ISO/IEC 19514:2017 - Information technology -- Object management group systems modeling language (OMG SysML)." [Online]. Available: https://www.iso.org/standard/65231.html. [Accessed: 14-Nov-2017].

[5] A. Van Lamsweerde and E. Letier, 2004, "From object orientation to goal orientation: A paradigm shift for requirements engineering," in *Radical Innovations of Software and Systems Engineering in the Future*, Springer, 2004, pp. 325–340.

[6] "i* Intentional STrategic Actor Relationships modelling - istar." [Online]. Available: http://www.cs.toronto.edu/km/istar/. [Accessed: 30-Oct-2017].

[7] "Tropos |." [Online]. Available: http://www.troposproject.eu/. [Accessed: 09-Nov-2017].

[8] "GRL." [Online]. Available: http://www.cs.toronto.edu/km/GRL/. [Accessed: 09-Nov-2017].

[9] "Z.151 : User Requirements Notation (URN) - Language definition." [Online]. Available: https://www.itu.int/rec/T-REC-Z.151-201210-I/en. [Accessed: 09-Nov-2017].

[10] N. A. Qureshi, I. J. Jureta, and A. Perini, 2012, "Towards a Requirements Modeling Language for Self-Adaptive Systems," in *Requirements Engineering: Foundation for Software Quality*, 2012, pp. 263–279.

[11] "ISO/IEC 27000:2016 - Information technology -- Security techniques -- Information security management systems -- Overview and vocabulary," *ISO*. [Online]. Available: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=66435. [Accessed: 20-Oct-2016].

[12] Mead N, Hough E, Stehney T , 2005, Security quality requirements engineering (SQUARE) methodology. Carnegie Mellon Software Engineering Institute, Technical report CMU/SEI-2005-TR-009.

[13] S. F. Gürses and T. Santen, 2006, "Contextualizing Security Goals: A Method for Multilateral Security Requirements Elicitation.," in *ResearchGate*, 2006, pp. 42–53.

[14] C. B. Haley, R. Laney, J. D. Moffett, and B. Nuseibeh, 2008, "Security Requirements Engineering: A Framework for Representation and Analysis," *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 133–153, Jan. 2008.

[15] A. Zuccato, 2007, "Holistic security management framework applied in electronic commerce," *Comput. Secur.*, vol. 26, no. 3, pp. 256–265, May 2007.

[16] A. van Lamsweerde, 2004, "Elaborating Security Requirements by Construction of Intentional Anti-Models," in *Proceedings of the 26th International Conference on Software Engineering*, Washington, DC, USA, 2004, pp. 148–157.

[17] P. Giorgini, F. Massacci, J. Mylopoulos, and N. Zannone, 2006, "Requirements engineering for trust management: model, methodology, and reasoning," *Int. J. Inf. Secur.*, vol. 5, no. 4, pp. 257–274, Aug. 2006.

[18] E. Paja, F. Dalpiaz, and P. Giorgini, 2015, "Modelling and reasoning about security requirements in socio-technical systems," *Data Knowl. Eng.*, vol. 98, pp. 123–143, Jul. 2015.

[19] D. Mellado, E. Fernández-Medina, and M. Piattini, 2007, "A common criteria based security requirements engineering process for the development of secure information systems," *Comput. Stand. Interfaces*, vol. 29, no. 2, pp. 244–253, Feb. 2007.

[20] J. Jurjens, 2010, *Secure Systems Development with UML*. Berlin, Heidelberg: Springer-Verlag, 2010.

[21] P.Salini and S. Kanmani, 2012, "Security Requirements Engineering Process for Web Applications," *Procedia Eng.*, vol. 38, pp. 2799–2807, 2012.

[22] T. Lodderstedt, D. Basin, and J. Doser, 2002, "SecureUML: A UML-based modeling language for model-driven security," *«UML» 2002— Unified Model. Lang.*, pp. 426–441, 2002.

[23] T. M. Hale and J. C. Kvedar, 2014, "Privacy and Security Concerns in Telehealth," *Virtual Mentor*, vol. 16, no. 12, p. 981, Jan. 2014.

[24] V. Garg and J. Brewer, 2011, "Telemedicine Security: A Systematic Review," *J. Diabetes Sci. Technol.*, vol. 5, no. 3, p. 768, May 2011.

[25] R. Laleau, F. Semmak, A. Matoussi, D. Petit, A. Hammad, and B. Tatibouet, 2010, "A first attempt to combine SysML requirements diagrams and B," *Innov. Syst. Softw. Eng.*, vol. 6, no. 1–2, pp. 47–54, Mar. 2010.

[26] L. Apvrille and Y. Roudier, 2013, "SysML-Sec: A SysML environment for the design and development of secure embedded systems," *APCOSEC Asia-Pac. Counc. Syst. Eng.*, pp. 8–11, 2013.