Journal of Universal Computer Science, vol. 18, no. 6 (2012), 816-844
submitted: 15/10/11, accepted: 15/2/12, appeared: 28/3/12 © J.UCS

Syntactic and Semantic Extensions to Secure Tropos to Support Security Risk Management

Raimundas Matulevičius
(Institute of Computer Science, University of Tartu, Tartu, Estonia
rma@ut.ee)

Haralambos Mouratidis
(School of Architecture, Computing and Engineering
University of East London, London, UK
haris@uel.ac.uk)

Nicolas Mayer, **Eric Dubois**
(CRP Henri Tudor - CITI, Luxembourg, Luxembourg
{nicolas.mayer, eric.dubois}@tudor.lu)

Patrick Heymans
(PReCISE, Computer Science Faculty, University of Namur, Namur, Belgium
phe@info.fundp.ac.be)

Abstract: The need to consider security from the early stages of the development process of information systems has been argued by academics and industrialists alike, and security risk management has been recognised as one of the most prominent techniques for eliciting security requirements. However, although existing security modelling languages provide some means to model security aspects, they do not contain concrete constructs to address vulnerable system assets, their risks, and risk treatments. Furthermore, security languages do not provide a crosscutting viewpoint relating all three – assets, risks and risk treatments – together. This is problematic since, for a security analyst, it is difficult to detect what the potential security flaws could be, and how they need to be fixed. In this paper, we extend the Secure Tropos language, an agent- and goal-oriented security modelling language to support modelling of security risks. Based on previous work, where we had observed some inadequacies of this language to model security risks, this paper suggests improvements of Secure Tropos semantics and syntax. On the syntax level we extend the concrete and abstract syntax of the language, so that it covers the security risk management domain. On the semantic level, we illustrate how language constructs need to be improved to address the three different levels of security risk management. The suggested improvements are illustrated with the aid of a running example, called eSAP, from the healthcare domain.
Key Words: Risk management, information system, security, Secure Tropos, syntax and semantics of modelling language.
Category: D.2.1, D.2.2, H.1, H.4.2, J.6.

1 Introduction

Information systems (IS) undoubtedly play an important role in todays society and more and more are at the heart of critical infrastructures. As such, developers of IS face an increasing complexity because of issues such as interoperability with other systems and IS's operation in open, distributed and mobile environments. In such context, the need to secure IS and the information stored in them is vital. Security risk management activities are considered as central by IS professionals towards the development of secure IS. These activities do not only support information system developers in the handling of security vulnerabilities but they also provide a framework in terms of which the return on investment of the security solutions is evaluated against the economic and business consequences of not implementing them. Therefore, the motivation of our work comes from the need to facilitate a framework to support IS developers to identify security risks and develop appropriate security requirements.

A central focus of IS development is to consider security issues from the very early phases, a.k.a. Requirements Engineering (RE). The associated scientific literature features a number of modelling languages specifically dedicated to security sensitive contexts. However, although existing security modelling languages provide some means to model security aspects, they do not contain concrete constructs to address vulnerable system assets, their risks, and risk treatments. Furthermore, security languages do not provide a crosscutting viewpoint relating all three – assets, risks and risk treatments – together. This is problematic since, for a security analyst, it is difficult to detect what the potential security flaws could be, and how they need to be fixed. This situation advocates for the design of yet another modelling language. However, defining a complete new notation does not appear to us as viable option from a sustainability perspective for the modelling community. As demonstrated for example with UML in software engineering, a consensus over unified and common notations has been proved to be a big push for the adoption of modelling practices in public and private companies. At RE level we plead for a similar approach and rather than the development of a totally new language we improve existing languages, offering an ontological basis sufficiently closed to the risk management domain.

With respect to the above objective, we have identified Secure Tropos [Mouratidis, 2004], which uses the concept of security constraint and methods such as security attack scenarios to analyse security requirements, as a suitable candidate language. The selection of Secure Tropos results from a detailed analysis [Matulevičius et al., 2008b] of the adequacy of its concepts to the Information System Security Risk Management (ISSRM) domain model [Mayer, 2009], [Mayer et al., 2007]. In this paper, based on our previous work [Matulevičius et al., 2008b], we extend the Secure Tropos language to support modelling of security risks. In particular, this paper suggests improvements of Secure Tropos's semantics and syntax. On the syntax level we extend the concrete and abstract syntax of the language, so that it covers the security risk management domain. On the semantic level, we illustrate how language constructs need to be improved to address the three different levels of security risk management.

This paper is structured as follows. Section 2 describes the ISSRM domain model and process. Section 3 describes the Secure Tropos language and it discusses how the current version of the language is suitable for ISSRM. Section 4 presents our extensions to the Secure Tropos language and Section 5 discusses related work. Section 6 presents areas for future work and it concludes the paper. It is worth noting that a running example from the health and social care domain is employed throughout the paper to demonstrate the applicability and improve understanding of our work.

2 Information System Security Risk Management (ISSRM)

The main objective of ISSRM [Mayer, 2009], [Matulevičius et al., 2008a], [Matulevičius et al., 2008b] is defined as the protection of essential IS constituents against all harm to information security. The domain model (see Fig. 1) is defined after careful survey [Mayer et al., 2007], [Mayer, 2009] of the risk management standards [ISO/IEC Guide 73, 2002], [AS/NZS 4360, 2004], security-related standards [ISO/IEC 27001, 2005], [ISO/IEC 13335-1, 2004], security risk management methods [Alberts and Dorofee, 2001], [Vraalsen et al., 2007], [Insight Consulting, 2003] and software engineering frameworks [Haley et al., 2008], [Firesmith, 2007]. It is important to note that security risk management is a specification of risk management in general, leading to conceptual and semantic differences between the two domains. For example, the standard definition of a risk (in general) coming from ISO Guide 73 [ISO/IEC Guide 73, 2002] and the one of a security risk defined in ISO/IEC 27000 [ISO/IEC 27000, 2009] are different and involve different components, the main difference being the absence of the vulnerability concept in a risk in general. The domain model is structured around three groups of concepts: asset-related concepts, risk-related concepts and risk treatment-related concepts. In this section, we briefly discuss these concepts (see [Mayer, 2009] for more details).

Asset-related concepts describe the elements that need to be protected within the organisation and the criteria that guarantee security of these elements. *Assets* are things that have value to the organisation and that need to be protected (e.g., medical reimbursement process, operating systems, people encoding data). Assets are structured to *business assets* and *IS assets*. Business assets are assets related to the organisation in terms of its business model and are necessary for achieving its objectives (e.g., information, process, capabilities, employees' skills). IS assets are component or part of the IS supporting business assets (e.g., laptop computer, system administrator). *Security criterion* is a property or constraint on business assets characterising their needs for security. It is expressed in terms of *confidentiality*, *integrity*, and *availability* (examples of security criteria

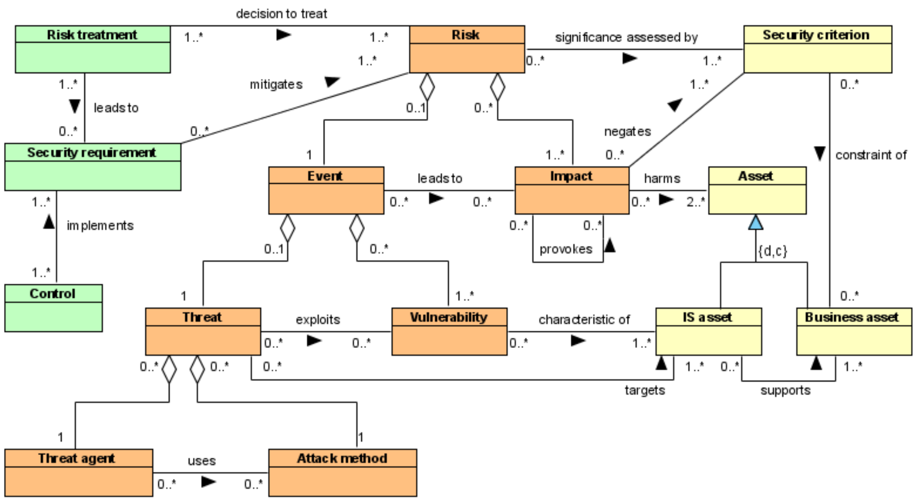


Figure 1: The ISSRM Domain Model [Mayer, 2009], [Matulevičius et al., 2008a], [Matulevičius et al., 2008b]

applied on business assets are confidentiality of the list of names, integrity of the medical reimbursement process). *Risk-related concepts* present how risk itself is defined. A *risk* is the combination of a *threat* with one or more *vulnerabilities* leading to negative *impacts* harming one or more of the *assets*. Example of a risk is a cracker using social engineering on a member of the organisation, because of weak awareness of the staff, leading to non-authorised access on personal computers and loss of confidentiality and integrity of sensitive information. A *threat* and a *vulnerability* are composing a security *event* (e.g., a cracker using social engineering because of lack of awareness). A *threat* (e.g., a cracker using social engineering) is the potential attack targeting *IS assets* that may lead to *harm*. A *vulnerability* (e.g., weak awareness) is a characteristic of an *IS asset* that can constitute a weakness or a flaw in terms of security. A *threat* is composed of a *threat agent* and an *attack method* . A *threat agent* is an agent that can potentially cause harm to assets of the IS (e.g., a cracker with considerable technical ability). An *attack method* is a standard means by which a threat agent carries out a threat (e.g., system intrusion). *Impact* is also a component of risk. It is the potential negative consequence of a risk that may harm assets of a system when a threat is accomplished (e.g., password discovery, loss of names confidentiality). *Risk treatment-related concepts* are the decisions, requirements and controls defined and implemented to mitigate the risks. A *risk treatment* is the treatment decision for a risk. It can include *risk reduction* (e.g., take measurements to avoid network intrusions), *risk avoidance* (e.g., do not connect IS to the Internet), *risk transfer* (e.g., take an insurance for covering the loss of service), and *risk retention* (e.g., accept that the service could be unavailable for one hour) decisions. Based on the taken decision, *security requirements* are defined to mitigate the risks (e.g., back-up copies of information and software shall be taken and tested regularly in accordance with the agreed backup policy). Generally, *risk reduction* decision leads to security requirements. *Risk transfer* decision needs also some security requirements about third parties. Avoiding risk and retaining risk do not need any additional security requirement. Finally, *controls* are the implementation of security requirements (e.g., backup procedure, building guard).

The ISSRM process consists of six steps [Mayer et al., 2007], [Matulevičius et al., 2008a], [Matulevičius et al., 2008b]. The process begins with *(a)* a study of the organisation's *context* and the identification of its *assets*. In this step, the organisation and its environment are described, and an overview of the IS, when already in place, is made. Then, based on the level of protection required for the assets, one needs to determine the *(b) security objectives*. Security objectives are defined in terms of confidentiality, integrity and availability of the assets. The main step of the process is *(c) risk analysis*, that elicits which risks are harming assets and threatening security objectives. It consists of identifying risks and estimating their level in a qualitative or quantitative manner. We speak about *risk assessment* [ISO/IEC Guide 73, 2002] only after the level of analysed risks is evaluated against the security needs, which are determined during *step (b)* of the process. This risk has an estimated level high enough to be considered. It could be necessary at this step to fully review the context and assets identification, if the risk assessment is considered as unsatisfactory. Once risk assessment is performed, decisions about *(d) risk treatment* are taken, like reducing the risk with some controls or transferring the risk to a third party. *Security requirements* on the IS can thus be determined as security solutions to mitigate the risks (*step (e)*). At the end of the risk treatment step, followed by the security requirements definition, if they are considered as unsatisfactory, the risk treatment step can be revised, or all of the preceding steps can be revised from the definition of the context and the assets. Requirements are finally instantiated into *(f) security controls*, i.e. system specific countermeasures, that are implemented within the organisation.

The ISSRM process is iterative. Several iterations need to be performed, until reaching an acceptable level for all risks. Even after reaching an acceptable level for all risks, the ISSRM process should be monitored and regularly reviewed. Risks are obviously not static and should be monitored either automatically or manually by the risk analyst in an organisation. Each modification in the organisation's business, in its context and/or in its IS can produce modifications on risks and its different levels. In an ideal way, the ISSRM process should in fact be continuously performed, in order to keep the organisation's business and its associated security needs aligned with the measures taken and the ensuing security level.

3 Secure Tropos

In this section, we present a summary of Secure Tropos. We also analyse how the current version of the language is suitable for ISSRM.

3.1 Secure Tropos for IS Modelling

Tropos [Bresciani et al., 2004] supports development of IS through four phases: early and late requirements, architectural and detailed design. *Early requirements analysis* aims at defining and understanding a problem by studying its existing organisational setting. *Late requirements analysis* defines the system-to-be in the context of its operational environment. *Architectural design* deals with the definition of the system global architecture in terms of subsystems. *Detailed design* specifies each architectural component in terms of inputs, outputs, control and other relevant information. In this paper, we particularly focus on the early stages of the IS development. This means we will consider how Secure Tropos is applied during early and late requirements analysis.

Secure Tropos [Mouratidis and Giorgini, 2007a], [Mouratidis et al., 2003a] is based on the Tropos methodology [Bresciani et al., 2004] with the scope to address IS security during its development. The key points of the methodology are: (*i*) social issues of security are analysed during the early requirements stage; (*ii*) security is considered simultaneously with the other requirements of the system-to-be; (*iii*) although introduced incrementally during the requirements phases, security is addressed in depth during the system design phases.

Secure Tropos supports the analysis of security considerations during an information systems development based on a number of models. The *security enhanced actor model* (SEAM)identifies and analyses actors of the environment, actors of the system and dependency relationships between them. Here, an *actor* describes an entity that has strategic goals and intentions within the system or within the organisational setting [Bresciani et al., 2004]. In order to deal with the security issues, security constraints are introduced. A *security constraint* represents a restriction related to security that the system must have and actors must respect [Mouratidis and Giorgini, 2007a], [Mouratidis et al., 2005].

A *Dependency* relationship between two actors indicates that one actor (the depender) depends for some reason (dependum) on another actor (the dependee) [Bresciani et al., 2004]. *Secure dependency*, introduces security constraint(s) that must be respected by actors for the dependency to be satisfied [Mouratidis et al., 2003a]. This means that the depender expects from the dependee to satisfy the security constraint(s) and also that the dependee will make an effort to deliver the dependum by satisfying the security constraint(s) [Mouratidis et al., 2005]. The ***security enhanced goal model*** (SEGM), allows a deeper understanding of how the actors reason about goals to be fulfilled, plans to be performed and availability of resources [Bresciani et al., 2004]. It completes the security enhanced actor model with the reasoning that each actor makes about its internal goals, plans and resources. A *hardgoal* hereafter, represents an actor's strategic interest. A *softgoal*, unlike a textit goal, does not have clear criteria for deciding whether it is satisfied or not and, therefore, it is subject to interpretation (goals are said to be *satisfied* while softgoals are said to be *satisficed*). A *plan* represents a way of doing things. A *resource* represents an informational or physical entity. In the security enhanced goal model, elements are linked by the *means-ends*, *decomposition* and *contribution* relationships.

To facilitate security analysis, Secure Tropos introduces the concept of secure goal (and task) [Mouratidis and Giorgini, 2007a]. A *secure goal* represents the strategic interest of an actor with respect to security. Secure goals are mainly introduced to achieve possible security constraints that are imposed to an actor or exist in the system. How secure goals are achieved is described by a *secure task.* A ***Security reference diagram*** assists the identification of security requirements [Mouratidis and Giorgini, 2007a]. In this type of diagram, a threat represents circumstances that have the potential to cause loss or problems, which can put in danger the security features of the system. On the other hand, an actual attack is defined with the aid of a security attack scenario [Mouratidis and Giorgini, 2007a]. Here, developers can identify the goals of the possible attacker and through these identify a set of possible attacks to the system. The *attacks* relationship shows what is the target of an attacker's plan.

3.2 Research Method

The main objective of this paper is to investigate how we can improve Secure Tropos [Mouratidis and Giorgini, 2007a] in order to support management of IS security risk at the early stages of the IS development. To reach this objective, we follow the research method presented in Fig. 2. First, we investigate the situation AS-IS. This means we need to understand how the current version of the language can be used to deal with ISSRM. We have reported this analysis in [Matulevičius et al., 2008b] and present major findings in Table 1 and section 3.3.

Next step is to deal with the observed limitations of the Secure Tropos support for ISSRM and to define the situation TO-BE. This means we need to understand how it is possible to improve the language that it would sufficiently

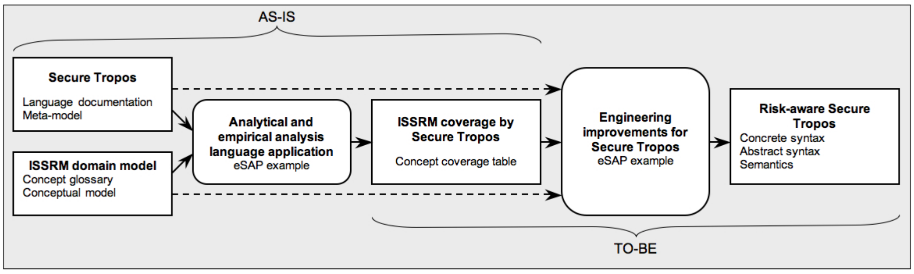


Figure 2: Research Method Followed

support the ISSRM concepts and the risk management process. In Section 4, we introduce extensions to Secure Tropos.

The study of Secure Tropos is done at the different levels of the modelling language. We analyse abstract syntax of the language by extending the current meta- model of Secure Tropos with concepts to address ISSRM. Based on this, we extend concrete syntax, thus, allowing modeller to represent asset, risk and risk treatment models. Finally, at the semantic level, we align Secure Tropos constructs with the ISSRM concepts, thus providing the semantical and methodological hints of language application.

3.3 Aligning Secure Tropos with ISSRM

In previous work [Matulevičius et al., 2008b], we have analysed how Secure Tropos can help to solve ISSRM problems at the early stages of IS development. First, we have surveyed the existing Secure Tropos literature [Mouratidis and Giorgini, 2007a], [Mouratidis et al., 2002b], [Mouratidis et al., 2006], [Mouratidis et al., 2005], [Mouratidis et al., 2003a] in order to understand its major principles and concepts. Next, we have applied Secure Tropos in the running example – eSAP [Mouratidis, 2004]. This application was strongly followed by the process guidance and the concepts suggested by the ISSRM domain model [Mayer et al., 2007], [Matulevičius et al., 2008a] [Matulevičius et al., 2008b], [Mayer, 2009]. We have resulted in the semantic alignment between ISSRM and Secure Tropos as illustrated in Table 1. This table shows how (and if) Secure Tropos can be aligned with the principles of ISSRM. The first two columns list the concepts of the ISSRM domain model, the third column provides synonyms of the ISSRM concepts found in the Secure Tropos literature [Mouratidis and Giorgini, 2007a], [Mouratidis et al., 2002b], [Mouratidis et al., 2006], [Mouratidis et al., 2005], [Mouratidis et al., 2003a]. The fourth column lists the Secure Tropos constructs used to address the ISSRM concepts.

Our alignment of Secure Tropos constructs with the concepts of the ISSRM

The ISSRM model		Secure Tropos	
		Synonyms found in literature*	Language constructs
Asset-related concepts	Asset	—	Actor, Goal, Softgoal, Plan, Resource
	Business asset	—	
	IS asset	—	
	Security criteria	Security feature, Protection property	Security constraint, Softgoal
Risk-related concepts	Risk	—	—
	Impact	—	Contribution between the threat and softgoal
	Event	—	Threat
	Threat	—	Goal, Plan
	Vulnerability	—	Belief **
	Threat agent	Attacker	Actor
	Attack method	—	Plan, Relationship attacks
Risk treatment-related concepts	Risk treatment	—	—
	Security requirement	Secure goal, secure plan, secure resource, protection objective, security requirements	Actor, Goal, Softgoal, Plan, Resource Security constraint
	Control	—	New model which implements security requirements

Table 1: Alignment ISSRM and Secure Tropos

domain model has shown several limitations of Secure Tropos, to investigate security risk management at the early stages (requirements) of the IS development. At the same time, it suggests a number of possible improvements for Secure Tropos, in the context of security risk management. We have noticed that Secure Tropos could be improved with additional constructs to better cover the concepts of ISSRM. Table 1 indicates that several concepts such as *risk* and *risk treatment* are not in the Secure Tropos approach. Thus, one needs either to define graphical constructs to address these concepts, or to provide methodological guidelines how these concepts might be addressed in the model.

Our analysis showed that Secure Tropos has to provide guidelines as to when and how to use each constructs, in order to avoid misinterpretations of the ISSRM concepts. For example, as shown in Table 1, the *plan* construct can be used to model *business assets*, *IS assets*, *threats* and *security requirements*. One possible solution in this situation might be introduction of labels in front of the construct label (e.g., [BS] – *business assets*, [IS] – *IS assets*, [Th] – *threat*, and [SR] – *security requirements*). Another solution is to design a discriminating concrete syntax, which would allow to separate these concerns. Finally, decomposition of the model into separate diagrams, where separate concerns (business assets, IS assets, attack scenario and security requirements) would be modelled, should be considered. The latter two aspects we develop in Section 4.

Finally, the semantics of individual modelling constructs should be adapted so that they adequately represent ISSRM concepts. For example, the *belief* construct only partially covers *vulnerability*. A possible improvement, on the one hand, is to suggest the modelling construct which would adequately support modelling of system vulnerabilities. On the other hand, recently in [Elahi and Yu, 2007], Elahi and Yu have introduced *vulnerable points*. We will investigate the latter option in Section 4.

4 A Risk-aware Secure Tropos

The purpose of this section is to develop syntactic, semantic and methodological extensions to Secure Tropos, that would support modelling security risks and their countermeasures. First, we suggest extensions to the concrete syntax and show how they are addressed in the abstract syntax. Next, we define methodological guidelines to use Secure Tropos for security risk management.

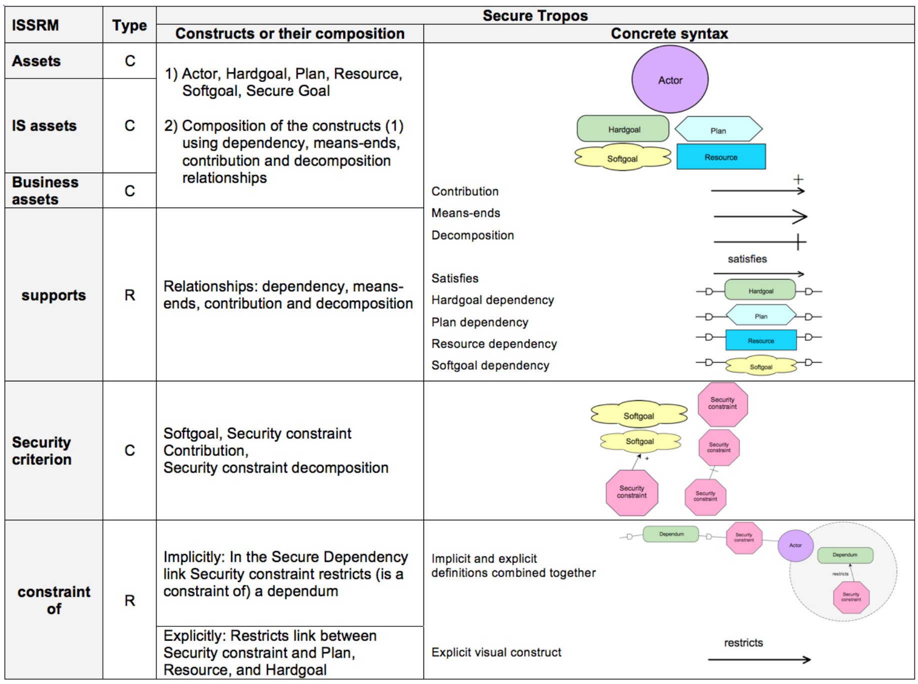
4.1 Concrete Syntax

In Section 3 (Table 1), we have separated concrete syntax of Secure Tropos according to three construct categories: asset-related concepts (Fig. 3), risk-related concepts (Fig. 4 and 5), and risk treatment-related concepts (Fig. 6). In addition to the ISSRM constructs aligned in Table 1, here in Fig. 3, 4, 5 and 6, we consider how ISSRM relationships (e.g., *supports*, *constraint of*, *exploits*, *targets*, *mitigates*, and others) can be expressed with Secure Tropos. We also make a link between Secure Tropos concrete and abstract syntax, which is considered in Section 4.2.

4.1.1 Asset-related concepts

The ISSRM *assets* (see Fig. 3) are modelled using *actor*, *hardgoal*, *plan*, *resource*, *softgoal* constructs and their compositions constructed using *dependency*, *means-*

ends, *contribution*, and *decomposition* relations. Moreover, ISSRM *supports* relationship, between IS and business assets, is expressed using the different Secure Tropos relationships. The ISSRM *security criterion* is represented through *softgoal* and/or *security constraint*. *Softgoal* represents generally high-level *security criteria* and *security constraint* their refinement. Note that in Secure Tropos, one *security constraint* can be decomposed to others, thus, forming a *security constraint* hierarchy. The ISSRM relationship *constraint of* is addressed both implicitly and explicitly in Secure Tropos. Firstly, in the Secure Tropos *actor model*, we can observe an implicit restriction of the dependum (*hardgoal*, *task* or *resource*) in the *dependency* relationship. This means that *security constraint* is imposed to the depender or/and dependee *actor*. Secondly, in the Secure Tropos SEGM, the ISSRM *constraint of* relationship is presented explicitly by the *restricts* relationship. It shows the actual goal, plan or resource restricted by the *security constraint*.



ISSRM	Type	Secure Tropos: Constructs or their composition	Secure Tropos: Concrete syntax
Assets	C	1) Actor, Hardgoal, Plan, Resource, Softgoal, Secure Goal 2) Composition of the constructs (1) using dependency, means-ends, contribution and decomposition relationships	
IS assets	C		
Business assets	C		Contribution Means-ends Decomposition
supports	R	Relationships: dependency, means-ends, contribution and decomposition	Satisfies Hardgoal dependency Plan dependency Resource dependency Softgoal dependency
Security criterion	C	Softgoal, Security constraint Contribution, Security constraint decomposition	
constraint of	R	Implicitly: In the Secure Dependency link Security constraint restricts (is a constraint of) a dependum	Implicit and explicit definitions combined together
		Explicitly: Restricts link between Security constraint and Plan, Resource, and Hardgoal	Explicit visual construct

Figure 3: Asset-related Concepts (C – concept, R – relationships)

4.1.2 Risk-related concepts

As presented in Table 1, standard Secure Tropos constructs can be used to model risk-related concerns. However, there exists a high degree of misinterpreting the presented information. Thus, we recommend to differentiate concrete syntax of these Secure Tropos concepts. Liu Et. Al [Liu et al., 2003], use black shadows to represent malicious language constructs. Elsewhere, Sindre and Opdahl [Sindre and Opdahl, 2005], model malicious information using contrasting construct colours (e.g., white *vs.* black). For Secure Tropos, we suggest to use more solid (darker) colours applied for the construct background (see Fig. 4). We represent *threat agent* as an *actor*, *attack method* as a *plan*, *threat* as a *hardgoal* and/or *plan*. As proposed in Section 3.3, *vulnerability point* is introduced to represent a vulnerability. This extension coming from Elahi and Yu [Elahi and Yu, 2007] is more aligned with *vulnerability* of ISSRM than the existing *Belief*. Secure Tropos *attacks* relationship represents the *targets* relationship of ISSRM. In order to be compliant with ISSRM, we also introduce the *exploits* relationship, which defines a link between a *plan* (ISSRM *threat*) and an *asset* with a *vulnerability point*.

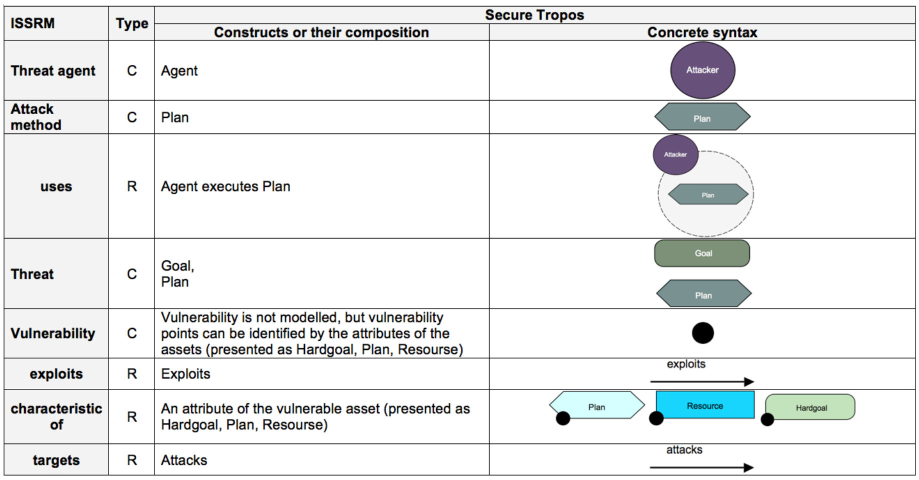
After defining how we can represent *threat agent*, *attack method*, and *vulnerability*, we can combine these concepts to represent the *event* of the risk (see Fig. 5). To generalise this representation, one can use the Secure Tropos *threat* constructs. The former representation of the risk *event* is used in the *security attack scenario*, in order to represent details of the event. The latter representation is used in the *security reference diagram*, to identify risks to assets. Here, a *risk* is understood as the combination of the risk *event* (represented as the Secure Tropos *threat*) and *impact* (represented using the *impacts* relationship).

4.1.3 Risk treatment-related concepts

For the necessity of differentiating ISSRM concepts, we also need to update the visual syntax of risk treatment-related concepts. Constructs, like *actor*, *hardgoal*, *plan*, *softgoal*, and *security constraint* (and/or their combinations), which represent *security requirements* and/or *controls*, need to carry a dotted background pattern (see Fig. 6). Security requirement mitigates the identified risk. To represent this, we introduce *mitigates* relationship, defining a link between constructs representing the ISSRM *security requirement* concept and the *threat* (as the ISSRM *event* of the risk).

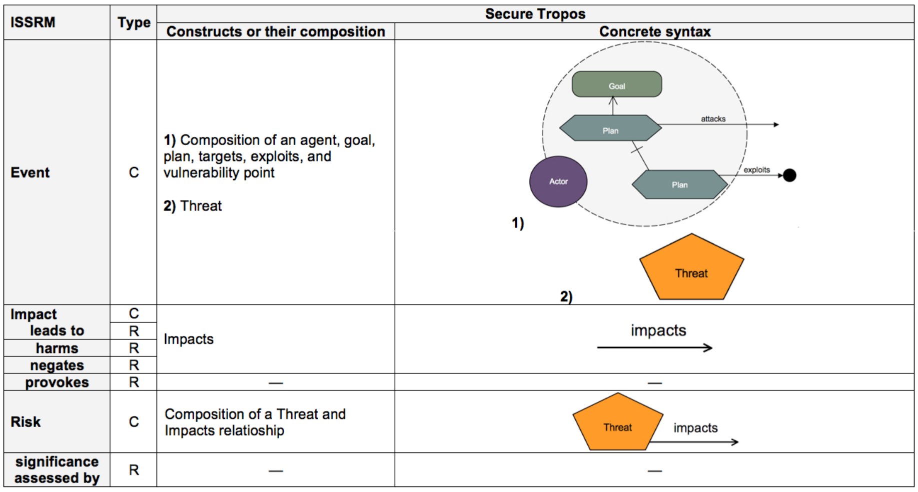
4.2 Abstract Syntax

In Section 3, we have not presented abstract syntax of Secure Tropos due to the need of the simple introduction of the language itself. However, to illustrate how



ISSRM	Type	Secure Tropos: Constructs or their composition	Secure Tropos: Concrete syntax
Threat agent	C	Agent	Attacker
Attack method	C	Plan	Plan
uses	R	Agent executes Plan	Attacker, Plan
Threat	C	Goal, Plan	Goal, Plan
Vulnerability	C	Vulnerability is not modelled, but vulnerability points can be identified by the attributes of the assets (presented as Hardgoal, Plan, Resourse)	
exploits	R	Exploits	exploits
characteristic of	R	An attribute of the vulnerable asset (presented as Hardgoal, Plan, Resourse)	Plan, Resource, Hardgoal
targets	R	Attacks	attacks

Figure 4: Risk-related Concepts - I (C – concept, R – relationships)



ISSRM	Type	Secure Tropos: Constructs or their composition	Secure Tropos: Concrete syntax
Event	C	**1)** Composition of an agent, goal, plan, targets, exploits, and vulnerability point **2)** Threat	1) Goal, Plan, attacks, Actor, Plan, exploits 2) Threat
Impact	C	Impacts	impacts
leads to	R		
harms	R		
negates	R		
provokes	R	—	—
Risk	C	Composition of a Threat and Impacts relatioship	Threat, impacts
significance assessed by	R	—	—

Figure 5: Risk-related Concepts - II (C – concept, R – relationships)

the proposed syntactic Secure Tropos extensions are used, we need to present abstract syntax elements and the rules how they can be combined together. The abstract syntax of Secure Tropos consists of two meta-models: Security Enhanced Actor Model(SEAM) and Security Enhanced Goal Model (SEGM). Due to the need of reducing presentation complexity, in addition to these two meta-models, we will discuss abstract relationships of the *security constraint* and *attack scenario diagram* separately.

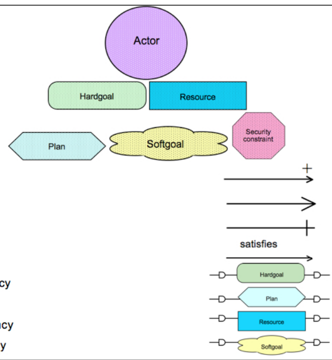
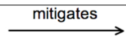
ISSRM	Type	Secure Tropos		
		Constructs or their composition	Concrete syntax	Reference to the meta-model
Risk treatment	C	—	—	
intention to treat	R	—	—	
Security requirement	C	1) Actor, goal, plan, resource, softgoal, security constraint 2) Components constructed when combining constructs (1) using dependency, means-ends, contribution and decomposition relationships	 Contribution Means-ends Decomposition Satisfies Hardgoal dependency Plan dependency Resource dependency Softgoal dependency	
Controls	C			
refines	R	—	—	
mitigates	R	Mitigates		
implements	R	Implicitely in the process of modelling		

Figure 6: Risk Treatment-related Concepts (C – concept, R – relationships)

4.2.1 Security Enhanced Actor Model

Fig. 7 presents the SEAM abstract syntax. The major element is an Actor who might be a *depender* or *dependee* in a Dependency relationship [Bresciani et al., 2004] [Yu, 1997]. A Security Constraint is imposed to an Actor, that represents a restriction on the Hardgoal(s), Plan(s) and/or Resource(s) on an Actor related to security issues [Mouratidis and Giorgini, 2007a]. A Security Constraint enhances the language by defining the notion of Secure Dependency.

A Secure Dependency introduces one or more Security Constraint(s) that must be fulfilled for the dependency to be valid [Mouratidis and Giorgini, 2007a]. We distinguish among three types of secure dependencies: *dependee secure dependency*, *depender secure dependency*, and *double secure dependency*. Different Secure Dependency types are defined using Depender and Dependee attributes of Security Constraint. *Dependee secure dependency* is represented when Dependee_SC is defined (Security_Constraint.Dependee equals one), *depender secure dependency* is received when Depender_SC is defined (Security_Constraint.Dependee equals one), and we get *double secure dependency* when both Depender_SC and Dependee_SC are defined.

4.2.2 Security Enhanced Goal Model

Fig. 8 presents the SEGM abstract syntax. Again the major element of this meta-model is an Actor who *executes* Plans, *uses* Resources, and *has* Goals. Plans

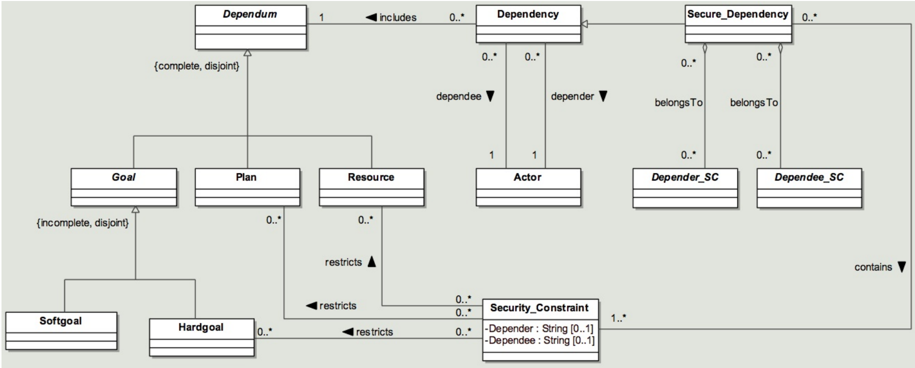


Figure 7: SEAM Abstract Syntax

can be and/or decomposed to other Plans, Resources, or Hardgoals. Hardgoals (and *Secure goals*) are achieved through and/or Means-ends relationship by satisfying other Hardgoals, executing Plans or making Resources available. In order to *satisfice* Softgoals, a sufficient degree contribution should be defined with other Softgoals, Security constraints, Plans, Resources or Hardgoals.

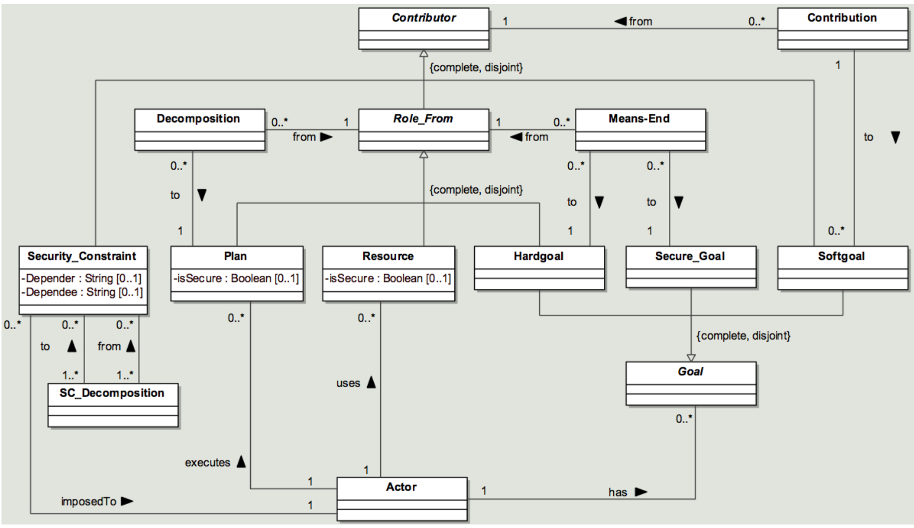


Figure 8: SEGM Abstract Syntax

As already illustrated in the SEAM meta-model, Security constraint is *imposed* to Actor. It Restricts (see Fig. 9) execution of Plans, availability of Resources and achievement of Hardgoals held by this Actor. One analyses Security Constraints using a number of modelling techniques, such as *security constraint decomposition*; *security constraint delegation* and *security constraint assignment* [Mouratidis, 2004]. A *secure goal* represents the strategic interest of an Actor with respect to security. *Secure goals* are mainly introduced to contribute to the satisfaction of Security Constraints by defining Satisfies relationship (see Fig. 9). A *secure plan* is defined as a Plan (by managing isSecure attribute) that represents a particular way for satisfying a *secure goal.* On the other hand, a *secure resource* is defined as an entity that is security critical for the system under development.

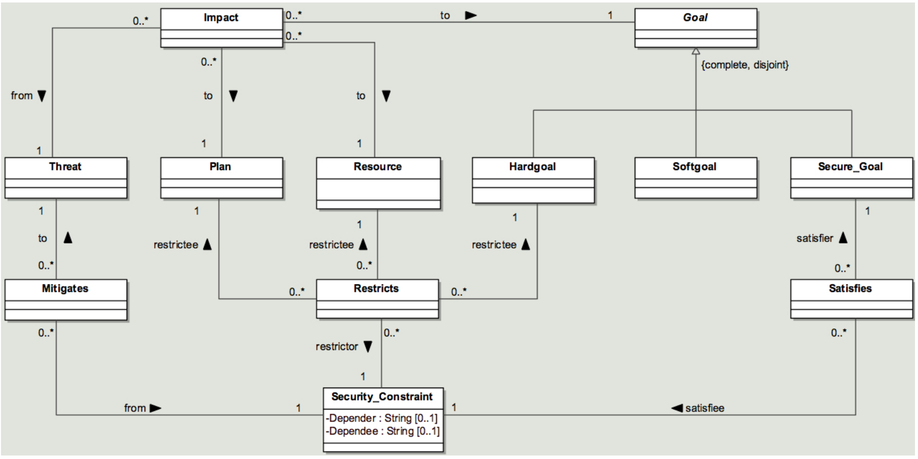


Figure 9: Abstract Syntax of Security Constraint and Threat

A proper definition of how Security constraint is satisfied, is needed to illustrate how it can mitigate a Threat. Threats are mitigated to lower the Impact to Plans, Resources and Hardgoals.

As discussed above, Security Constraint is one of the major elements which defines security concerns in the model; thus, it requires a special attention (see Fig. 9). Security Constraint has a number of relationships with other constraints of the language. Security Constraint can Restrict Plan, Resource and Hardgoal. The visual representation for Restrict is used in the SEGM model, however its implicit meaning is contained already in the SEAM model because Security Constraint places restrictions on the Secure Dependency fulfillment.

4.2.3 Security Attack Scenarios

Fig. 10 presents the abstract syntax of Secure Tropos used when defining the *security attack scenarios*. Here we have to note that, in *security attack scenarios*, two conceptually different sets of constructs are used: asset- and risk-related constructs to address the corresponding ISSRM concepts. Thus, they both obey the same syntax rules (presented in Fig. 7 and 8) when combined within this conceptual boundary. The difficulty arise when one wants to show relationship between them both.

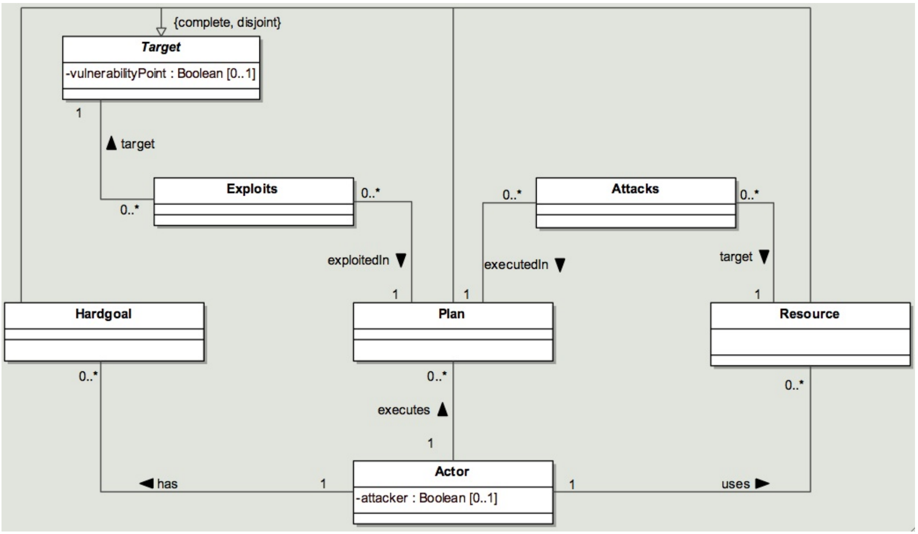


Figure 10: Abstract Syntax of Security Attack Scenario

We need to distinguish system actors (*assets*) from malicious actors (*attakers*). First, we introduce an attribute attacker to the class Actor, as shown in Fig. 10. Next, we define an integrity constraint, saying that Actor **A** who executes a Plan *exploiting/attacking* other elements in the diagram, and Actor B who holds *exploited/attacked* elements, are different. Finally, for actor **A**, we set an attribute attacker *true*, meaning a malicious actor (graphical representation is provided in Fig. 4). Actor **B**'s attribute is set as false, meaning that this actor represents attacked assets (graphical representation in Fig. 3).

Two relationships are defined between elements held by these two actors. A Plan executed by an *attacker* Exploits a target (Hardgoal, Resource, or Plan). Exploits relationship points to the vulnerability point (see attribute vulnerabilityPoint) of the target. The Attacks relationship shows a link between a Plan executed by a *malicious actor* and the Resource used by an attacked actor.

In the next subsection, we will provide methodological guidelines for the Risk-aware Secure Tropos application. We will use the eSAP example, already introduced in Section 3.

4.3 Application of Risk-aware Secure Tropos

The objective of this section is to demonstrate how concrete and abstract syntax extensions are used in an example. Here, we will use the eSAP example [Mouratidis, 2004] again and incrementally provide guidelines for modelling with Risk-aware Secure Tropos.

Language application includes three major stages. The first stage covers the two first steps of the ISSRM process, presented in Section 2: *context and asset identification* and *determination of security objectives*. The second stage comprises *risk analysis and assessment*. Finally, the third stage corresponds to *security requirements definition*, coming from *risk treatment* decisions, and leading to new controls.

4.3.1 Stage 1. Asset identification and determination of security objectives

At this stage, concrete syntax of Secure Tropos does not differ from the standard one presented in [Mouratidis et al., 2002b], [Mouratidis et al., 2002a], [Mouratidis et al., 2003b], [Mouratidis and Giorgini, 2004], [Mouratidis et al., 2006] and used in Section 3. However, as we discussed in Section 3, here we need to make a separation between two ISSRM concepts, namely *business assets* and *IS assets*. We do this separation by constructing two diagrams: one presenting business assets (Fig. 11), another introducing IS assets (Fig. 12). In the first diagram shown in Fig. 11, there is no information about how the eSAP system supports different processes or information (i.e. how it supports the business assets). Here, we represent only goals (e.g., Care information collected), plans (e.g., Collect info about treatment) and resources (e.g., Patient personal information) related to business artefacts and activities.

Following the steps of the risk management process, we need to define security objectives. In Secure Tropos, it is possible to identify general security objectives using softgoals (e.g., Privacy in Fig. 11) and then to refine them using security criteria expressed with security constraints (e.g., Keep system data privacy and Share info only if consent obtained). This strategy is a 'top-down' security objectives identification. However, in Secure Tropos, after defining actor model, it is more natural to define implicit security objectives as the secure dependencies. Then, identified security constraints (e.g., Share info only if consent obtained) are examined with respect to security objectives of higher level (e.g., Keep system data privacy and Privacy) for the system. This strategy we name as 'bottom-up'.

Then, the IS assets are represented in a diagram, shown in Fig. 12. Here, the main objective is to discover what *plans* have to be performed, *resources* should be available, and *goals* need to be fulfilled, in order to *support business assets*. Satisfying security constraints is performed through *secure goals*, which are considered as IS assets. In Fig. 12, Share info only if consent obtained is satisfied if the goal Consent has been obtained is fulfilled. The plan Perform authorisation checks satisfies the security goal System privacy ensured, thus, contributing to the support of business assets through guaranteeing security constraints. In Fig. 12, *business assets* and *IS assets* are mixed, because we need to represent how IS assets *support* business assets.

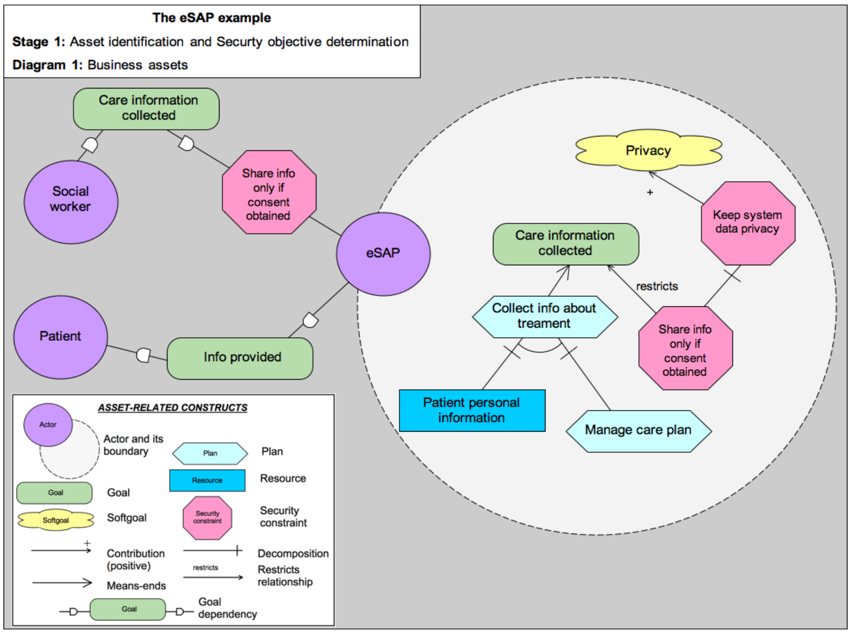


Figure 11: Modelling of eSAP Business Assets

4.3.2 Stage 2. Risk analysis and assessment

At the second stage, we introduce possible risks to the eSAP system. We start by determining the security events. Fig. 13 focuses on a possible *event* of the risk to which eSAP could be exposed, called Authentication attack. It describes a situation where a *threat agent* passes himself off as a trusted actor in order to fake identity and to damage the Patient personal information in the Information database. The Authentication attack impacts Privacy. The traceability between Privacy and Patient personal information shows the *harm* at the business level (Fig. 11). However, in this situation Privacy can be interpreted twofold. Firstly, it can represent an *asset*, which is important to an organisation. Then, the impacts link represents a *harm* the risk makes. Secondly, Privacy could be considered as a

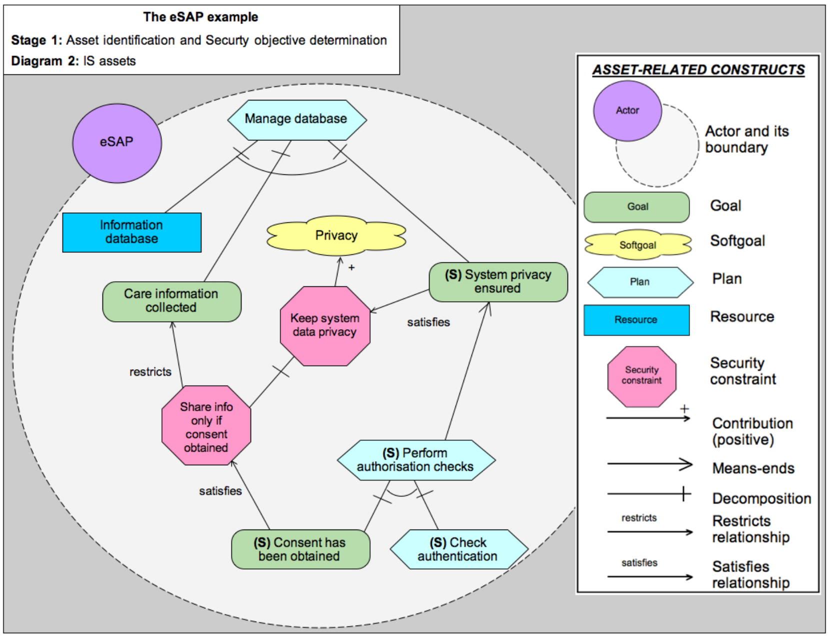


Figure 12: Modelling of eSAP IS Assets

emphsecurity criterion, which needs to be respected. In this case, *impact* defines *negation* of the security criterion.

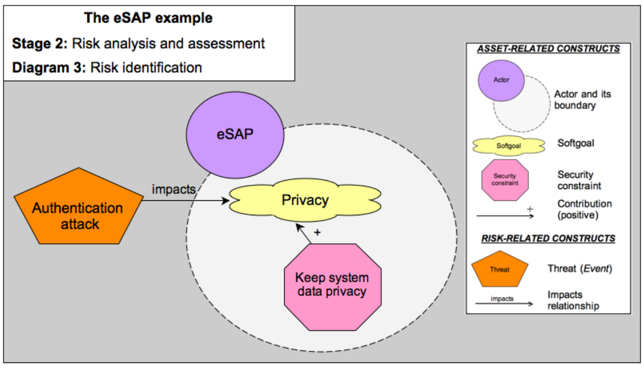


Figure 13: Identification of an Authentication Attack

After identifying the possible risk, we need to refine it in terms of *threat*, *vulnerability*, *threat agent* and *attack method*. This is done in the security attack scenario in Fig. 14. Here, an Attacker has a *threat* (Collect info about breaking the system) to an IS asset Information database, which supports business asset Patient personal information. Attacker *attacks* Information database through exploiting the vulnerability identified in Perform authorisation check. Thus, the *exploits* link shows a relationship between an *attack method* (Check eSAP access repeatedly) and a *vulnerable IS asset* (Perform authorisation checks).

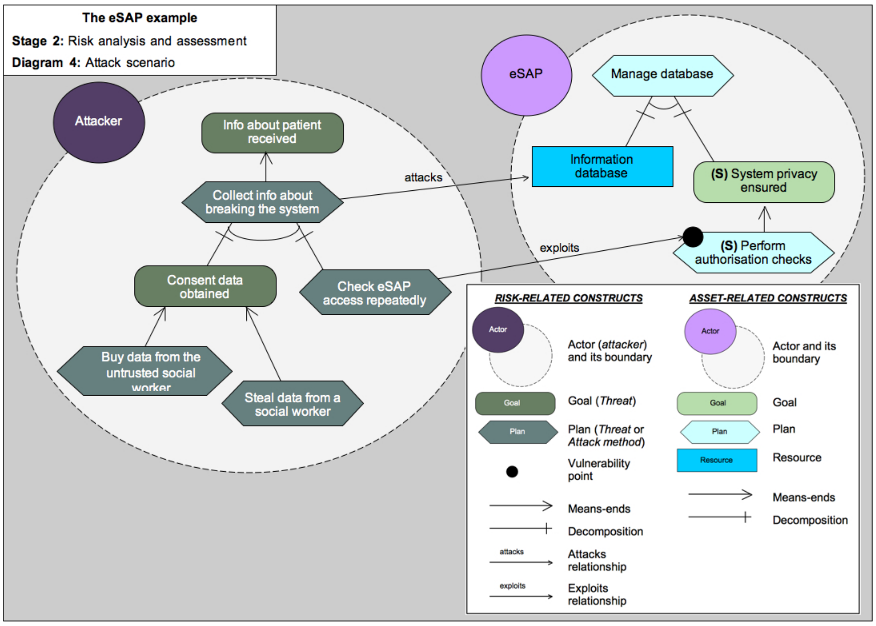


Figure 14: Potential Attack Scenario

4.3.3 Stage 3. Security requirements definition

In order to mitigate the identified risk about an Authentication attack, in our example, we have chosen a *risk reduction* decision. This means we have to design goals and plans that mitigate the risk. In this example, we substitute plans Perform authorisation check and Check authentication (see Fig. 12) with plan Perform cryptographic procedures decomposed in Encrypt data and Decrypt data (see Fig. 15). Note that new plans have a dotted background pattern, thus, identifying that they represent security requirements in this diagram. In this situation, the Keep system data privacy also becomes a security requirement mitigating the risk.

As discussed in Section 2, ISSRM process is iterative. After definition of security requirements, one needs to test the system again against new possible risk

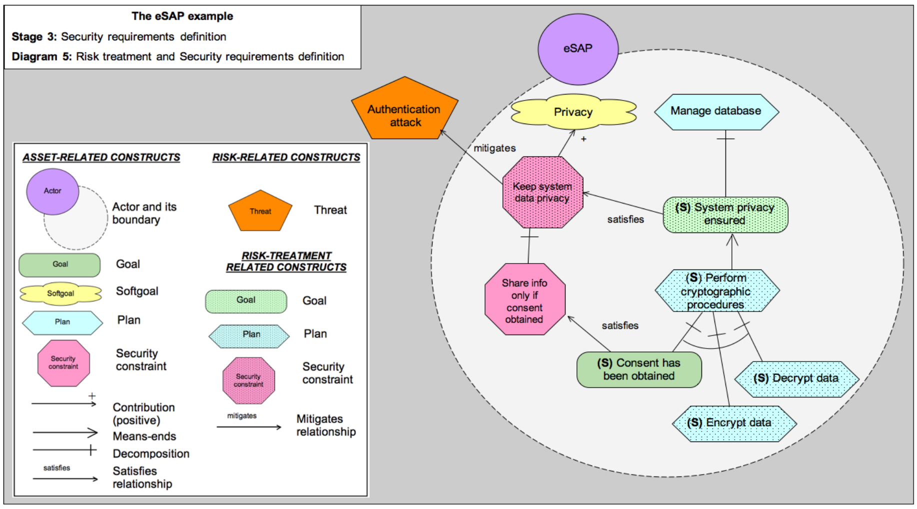


Figure 15: Risk Treatment and Security Requirements Definition

events. For example, modeller can now identify Cryptographic attack. This means that the modeller will need to analyse new vulnerabilities and define new countermeasure. The first iteration activity is to assume new security requirements become controls, and are, therefore, part of IS. This means that plans Perform cryptographic procedures, Encrypt data, and Decrypt data should become IS assets, so removing their pattern in the diagram. A risk analysis and assessment can be performed again.

4.4 Theoretical evaluation

We will evaluate our proposal according to the principle of semiotic clarity proposed by Opdahl and Henderson-Sellers and Moody [Opdahl and Henderson-Sellers, 2005], [Moody, 2009]. According to this principle, there should be a one-to-one correspondence between a visual language construct and its referent concept. Otherwise we need to speak about language *redundancy*, *overload*, *incompleteness* (*deficit*), and *under-definition* (*excess*) problems.

4.4.1 Redundancy

means that two language constructs have the same or overlapping semantics. Redundancy problems with respect to ISSRM were identified in Secure Tropos. Firstly, in Risk-aware Secure Tropos, we have decreased redundancy level by introducing different visual constructs to model asset-, risk-, and risk treatment-related concepts. Secondly, it might seen that, within the conceptual groups, there is still a high degree of redundancy. For example, an ISSRM *asset* can be expressed using almost all concepts of Risk-aware Secure Tropos (e.g., Actor, Hardgoal, Softgoal, Plan, Resource). However, we do not see it as a limitation, but rather the opposite. When following the ISSRM *asset* definition, we need to have means to express information (by Resource), process (by Plan) and different organisational objectives (by different types of a Goal). Similar needs can be observed within other two conceptual groups.

An ISSRM *security criterion* can be represented either by Softgoal or by Security constraint. This correspondence is not used for the same modelling purpose. We represent abstract *security criterion* (e.g., confidentiality, integrity, and availability) using *Softgoals* and more concrete *security criterion* using *Security constraints*.

As mentioned previously, the concept of *event* is represented by Threat in the *security reference diagram* and by a set of constructs (e.g., goals, plans, actors, etc.) in the *security attack scenario*. Hopefully, this separation of concepts to different levels of abstraction gives better model analysis possibilities, and facilitates the user to catch the information provided in the diagrams [Moody, 2002]. However, this needs to be validated in empirical settings.

4.4.2 Overload

exists if the same language construct has several meanings. In our proposal, there is a link impacts, which is used to represent *impacts negates* and *impact harms* concepts of the ISSRM domain model. We allow this overload, first, because it keeps the language relatively simple, without too many modelling constructs. Second, the semantical difference is captured in the label of the impacted construct (Goal, Plan, Resource or Softgoal), as we have discussed in Section 4.3.

4.4.3 Incompleteness

(or deficit) appears when a language does not convey information on a certain phenomenon. With respect to the incompleteness, first, we need to discuss concepts, which, although present in the ISSRM domain model, are skipped in the Risk-aware Secure Tropos. These are *Risk treatment* (and relationships *decision to treat* and *leads to*), relationships *provokes* and *refines*.

We do not define visual construct for *Risk treatment* (also relationships *decision to treat* and *leads to*), because this concept does not present any modification done to the modelled IS. This concept stands as a rationale and indicates modeller's mental decision. Nevertheless, it needs to be recorded in the system specification, additionally to the created IS model, using other means.

In Risk-aware Secure Tropos, we do not define the single concept of *Risk*. We represent it as combination of a Threat and Impacts relationship. This means that the ISSRM relationship *significance assessed by* is not explicitly represented by a link. However, we can implicitly identify this relationship by analysing links between *security criteria* (expressed using Softgoals or Security constraints) and the concerned *risk* (expressed by the Threat and Impacts).

Due to the overlapping semantics of the Impacts relationship, we can only implicitly define *provokes* relationship. This is done through multiple use of the impacts link. However, language does not allow modelling which impact has provoked which impact. This information needs to be captured using other means.

Some concepts addressed in Risk-aware Secure Tropos are considered differently than how they are defined in the ISSRM domain model. For example, the ISSRM *threat* consists of a *threat agent* and an *attack method*. Following principles of Tropos, we define that attack agent (Actor) holds *threat* and *attack method* (expressed using Hardgoals and Plans).

Further, the ISSRM *event* consists of *threat* and *vulnerability*. In case of Risk-aware Secure Tropos, we define *event* either as a Threat or as a combination of an Actor, Goal, Plan, Vulnerability point, Targets and Exploits. In this situation, we are not able to identify the precise vulnerability *per se* (only the point where it exists). This means that exact vulnerability needs to be specified using other means.

4.4.4 Under-definition

(or excess) arises when a language construct has no semantics. In our proposal, we do not observe any under-definition problem.

5 Related Work

At the various IS development stages, security can be addressed using various models; for example, *goal models* created with *i** [Yu, 1997], Tropos [Bresciani et al., 2004] or KAOS [van Lamsweerde, 2001]; UML class diagrams [Object Management Group (OMG), 2004]; BPMN [OMG, 2008], and so on. But these languages were not designed with security in mind and, thus, their support for it is weak. There are also *security modelling languages* specifically dedicated for analysis and modelling of IS security concerns. For example, Abuse frames [Lin et al., 2004] suggest means to consider security during early RE. UMLsec [Jurjens, 2002] and SecureUML [Lodderstedt et al., 2002] are used to address security during system design. Goal modelling languages have also been adapted to security. Secure *i** [Elahi and Yu, 2007] addresses security trade-offs. KAOS extension to security [van Lamsweerde, 2004] adds anti-goal models designed to elicit attackers rationales. Tropos has been extended to the Secure Tropos [Mouratidis and Giorgini, 2007a] methodology considered in this paper.

Abuse cases [McDermott and Fox, 1999], Misuse cases [Sindre and Opdahl, 2005] and Mal-activity [Sindre, 2007] diagrams address security concerns through negative scenarios or processes executed by the attacker. Relevant to risk, Mellado et al. [D. Mellado, 2007] and [D. Mellado, 2010] have presented work related to security requirements approaches based on risk. the Tropos Goal-Risk (GR) framework is another Tropos extension that considers the concept of 'risk' [Asnar and Giorgini, 2006]. Its objective is to assess the risk of uncertain events over organisation strategies and to evaluate the effectiveness of treatments [Asnar et al., 2008]. Regarding our scope, it is necessary to note that the range of risks supported by Tropos GR framework is not focussed on IS security. It is open to risk in general, taking place in different domains at the level of an organisation, like risk in project management or financial risk. Finally, in [Gandhi and Lee, 2007], a model is proposed to explain the relationships between security requirements and risk components, for certification and accreditation purpose. It is used for identifying the risk components, and map them to concepts in domain-specific taxonomies (e.g., of threats, assets, vulnerabilities, countermeasures) defined within the approach. This model is based on the Common Criteria model [Common Criteria, 2006], that is considered in our ISSRM domain model too.

In most cases, the above languages have not been specifically designed with security aspects in mind. Such aspects have been incrementally introduced and have enriched existing languages, because of the growing importance of security. As a consequence, such languages have progressively included security concepts, without a real systematic language design approach. Moreover, no perfect match with respect to ISSRM is provided by any existing modelling language. Although some languages include some risk concepts, their approaches are not complete regarding ISSRM. The languages also lack guidelines on how they can fulfil the needs of different stakeholders; i.e., representing and unifying individual viewpoints and concerns related to IS security and security risk management.

6 Conclusions

In this paper, we have analysed how Secure Tropos can be used to manage security risks at the early stages of IS development. First, we have identified language limitations with respect to the ISSRM domain model. Next, we have extended both language syntax and semantics, in order to respect the guidelines of ISSRM. Our work has resulted in a Risk-aware Secure Tropos. In addition to the language itself, we have defined methodological guidelines for the language application.

Our proposal has few limitations with respect to Secure Tropos, from which it was derived. In this work, we have stressed that our purpose is to develop a security risk management approach specifically used during the early stages of IS development. This means that we do not consider Secure Tropos extensions to security, which are defined at the late stages of system development. For example, we do not take into account actor *capability* analysis [Mouratidis et al., 2004], [Mouratidis and Giorgini, 2007b], or how Secure Tropos models can be used in the system design stages [Mouratidis et al., 2006]. We understand that these extensions are important for the later modelling stages, however, with respect to Risk-aware Secure Tropos, they require additional investigation.

Although we have applied our proposal to the running eSAP example, we acknowledge that more practice-oriented case study is necessary. As the future work, we plan to experiment the language in a case study to validate its usefulness and effectiveness. Application of the Risk-aware Secure Tropos would be easier if it was supported by a software tool. Currently, we are working in the area of the meta-case tool development [Englebert and Heymans, 2007]. We hope that a meta-case tool would allow us to engineer case tools from the modelling language meta-model. The meta-model of the Risk-aware Secure Tropos will be used as the input to generate a prototype tool supporting our proposal, and to test it in the experimental environment.

References

[Alberts and Dorofee, 2001] Alberts, C. J. and Dorofee, A. J. (2001). *OCTAVE Method Implementation Guide Version 2.0.* Carnegie Mellon University - Software Engineering Institute.

[Asnar and Giorgini, 2006] Asnar, Y. and Giorgini, P. (2006). Modelling Risk and Identifying Cuntermeasure in Organizations. In *Proceedings of the 1st Interational Workshop on Critical Information Intrastructures Security*, pages 55–66. Springer-Verlag Berlin Heidelberg.

[Asnar et al., 2008] Asnar, Y., Moretti, R., Sebastianis, M., and Zannone, N. (2008). Risk as Dependability Metrics for the Evaluation of Business Solutions: A Model-driven Approach. In *ARES*, pages 1240–1247.

[AS/NZS 4360, 2004] AS/NZS 4360 (2004). *Risk Management.* SAI Global.

[Bresciani et al., 2004] Bresciani, P., Giorgini, P., Giunchiglia, F., Mylopoulos, J., and Perini, A. (2004). TROPOS: an Agent-oriented Software Development Methodology. *Journal of Autonomous Agents and Multi-Agent Systems*, 8:203–236.

[Common Criteria, 2006] Common Criteria (2006). *Common Criteria for Information Technology Security Evaluation version 3.1.*

[D. Mellado, 2007] D. Mellado, E. Fernndez-Medina, M. (2007). A Common Criteria Based Security Requirements Engineering Process for the Development of Secure Information Systems. *Computer Standards and Interfaces*, 29:244–253.

[D. Mellado, 2010] D. Mellado, E. Fernndez-Medina, M. (2010). Security Requirements Engineering Framework for Software Product Lines. *Information and Software Technology*, 52:1094–1117.

[Elahi and Yu, 2007] Elahi, G. and Yu, E. (2007). A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs. In Parent, C., Schewe, K.-D., Storey, V. C., and Thalheim, B., editors, *Proceedings of the 26th International Conference on Conceptual Modelling (ER 2007)*, volume 4801, pages 87–101. Springer-Verlag Berlin Heidelberg.

[Englebert and Heymans, 2007] Englebert, V. and Heymans, P. (2007). Towards More Extensible MetaCASE Tools. In Krogstie, J., Opdahl, A. L., and Sindre, G., editors, *Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE 2007)*, pages 454–468.

[Firesmith, 2007] Firesmith, D. (2007). Engineering Safety and Security Related Requirements for Software Intensive Systems. In *Companion to the Proceedings of the 29th International Conference on Software Engineering (ICSE COMPANION '07)*, page 169. IEEE Computer Society.

[Gandhi and Lee, 2007] Gandhi, R. A. and Lee, S.-W. (2007). Discovering and Understanding Multi-dimensional Correlations among Certification Requirements with application to Risk Assessment. *Requirements Engineering, IEEE International Conference on*, 0:231–240.

[Haley et al., 2008] Haley, C., Laney, R., Moffett, J., and Nuseibeh, B. (2008). Security Requirements Engineering: A Framework for Representation and Analysis. *IEEE Transactions on Software Engineering*, 34(1):133–153.

[Insight Consulting, 2003] Insight Consulting (2003). *CRAMM (CCTA Risk Analysis and Management Method) User Guide version 5.0.* SIEMENS.

[ISO/IEC 13335-1, 2004] ISO/IEC 13335-1 (2004). *Information Technology – Security Techniques – Management of Information and Communications Technology Security – Part 1: Concepts and Models for Information and Communications Technology Security Management.* International Organisation for Standardisation.

[ISO/IEC 27000, 2009] ISO/IEC 27000 (2009). *Overview and vocabulary, International Organisation for Standardisation.* International Organisation for Standardisation.

[ISO/IEC 27001, 2005] ISO/IEC 27001 (2005). *Information Technology–Security Techniques–Information Security Management Systems–Requirements, International Organisation for Standardisation.*

[ISO/IEC Guide 73, 2002] ISO/IEC Guide 73 (2002). *Risk management – Vocabulary – Guidelines for Use in Standards.* International Organisation for Standardisation.

[Jurjens, 2002] Jurjens, J. (2002). UMLsec: Extending UML for Secure Systems Development. In *Proceedings of the 5th International Conference on the Unified Modelling Language (UML'02)*, pages 412–425.

[Lin et al., 2004] Lin, L., Nuseibeh, B., Ince, D., and Jackson, M. (2004). Using Abuse Frames to Bound the Scope of Security Problems. In *Proceedings of the 12th IEEE international Conference on Requirements Engineering (RE'04)*, pages 354–355. IEEE Computer Society.

[Liu et al., 2003] Liu, L., Yu, E., and Mylopoulos, J. (2003). Security and Privacy Requirements Analysis within a Social Setting. In *Proceedings of the 11th IEEE International Requirements Engineering Conference (RE'03)*, page 151. IEEE Computer Society.

[Lodderstedt et al., 2002] Lodderstedt, T., Basin, D. A., and Doser, J. (2002). SecureUML: A UML-based Modeling Language for Model-driven Security. In *Proceedings of the 5th International Conference on the Unified Modelling Language (UML'02)*, pages 426–441. Springer-Verlag.

[Matulevičius et al., 2008a] Matulevičius, R., Mayer, N., and Heymans, P. (2008a). Alignment of Misuse Cases with Security Risk Management. In *Proceedings of the ARES 2008 Symposium on Requirements Engineering for Information Security (SREIS 2008)*, pages 1397–1404. IEEE Computer Society.

[Matulevičius et al., 2008b] Matulevičius, R., Mayer, N., Mouratidis, H., Dubois, E., Heymans, P., and Genon, N. (2008b). Adapting Secure Tropos for Security Risk Management during Early Phases of the Information Systems Development. In *Proceedings of the 20th International Conference on Advanced Information System Engineering (CAiSE 2008).* Springer-Verlag Berlin Heidelberg.

[Mayer, 2009] Mayer, N. (2009). *Model-Based Management of Information System Security Risk.* PhD thesis, University of Namur, Namur, Belgium.

[Mayer et al., 2007] Mayer, N., Heymans, P., and Matulevičius, R. (2007). Design of a Modelling Language for Information System Security Risk Management. In *Proceedings of the 1st International Conference on Research Challenges in Information Science (RCIS 2007)*, pages 121–131.

[McDermott and Fox, 1999] McDermott, J. and Fox, C. (1999). Using Abuse Case Models for Security Requirements Analysis. In *Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC'99)*, page 55.

[Moody, 2002] Moody, D. L. (2002). Complexity Effects on End User Understanding of Data Models: an Experimental Comparison of Large Data Model Represetation Mehods. In *Proceedings of the 10th European Conferece on Information Systems (ECIS'2002)*.

[Moody, 2009] Moody, D. L. (2009). The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering*, 35(6):756–777.

[Mouratidis, 2004] Mouratidis, H. (2004). *A Security Oriented Approach in the Development of Multiagent Systems: Applied to the Management of the Health and Social Care Needs of Older People In England*. PhD thesis, Department of Computer Science, University of Sheffield, UK.

[Mouratidis and Giorgini, 2004] Mouratidis, H. and Giorgini, P. (2004). Enhancing Secure TROPOS to Effectively Deal with Security Requirements in the Development of Multiagent Systems. In *Proceedings of the 1st International Workshop on Safety and Security in Multiagent Systems (AAMAS 2004)*.

[Mouratidis and Giorgini, 2007a] Mouratidis, H. and Giorgini, P. (2007a). Secure Tropos: A Security-oriented Extension of the Tropos Methodology. *International Journal of Software Engineering and Knowledge Engineering (IJSEKE)*, 17(2):285–309.

[Mouratidis and Giorgini, 2007b] Mouratidis, H. and Giorgini, P. (2007b). Security Attack Testing (SAT) – Testing the Security of Information Systems at Design Time. *Information Systems*, 32(8):1166–1183.

[Mouratidis et al., 2002a] Mouratidis, H., Giorgini, P., Gordon, M., and Philp, I. (2002a). A Natural Extension of Tropos Methodology for Modelling Security. In *Proceedings of the Agent Oriented Methodologies Workshop (OOPSLA 2002)*.

[Mouratidis et al., 2002b] Mouratidis, H., Giorgini, P., and Manson, G. (2002b). Using Tropos Methodology to Model an Integrated Health Assessment System. In *Proceedings of the Fourth International Bi-Conference on Agent-oriented Information Systems (AOIS'02)*.

[Mouratidis et al., 2003a] Mouratidis, H., Giorgini, P., and Manson, G. (2003a). Integrating Security and Systems Engineering: Towards the Modelling of Secure Information Systems. In *Proceedings of the 15th Conference On Advanced Information Systems Engineering (CAiSE'03)*, pages 63–78. Springer-Verlag.

[Mouratidis et al., 2004] Mouratidis, H., Giorgini, P., and Manson, G. A. (2004). Using Security Attacks Scenarios to Analyse Security during Information Systems Design. In *Proceedings of the 6th International Conference on Enterprise Information Systems 2004 (ICEIS'04)*.

[Mouratidis et al., 2005] Mouratidis, H., Giorgini, P., and Manson, G. A. (2005). When Security Meets Software Engineering: a Case of Modelling Secure Information Systems. *Information Systems*, 30(8):609–629.

[Mouratidis et al., 2006] Mouratidis, H., Jurjens, J., and Fox, J. (2006). Towards a Comprehensive Framework for Secure Systems Development. In Dubois, E. and Pohl, K., editors, *Proceedings of the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06)*, pages 48–62. Springer-Verlag.

[Mouratidis et al., 2003b] Mouratidis, H., Philp, I., and Manson, G. (2003b). A Novel Agent-Based System to Support the Single Assessment Process of Older People. *Journal of Health Informatics*, 9(3):149–162.

[Object Management Group (OMG), 2004] Object Management Group (OMG) (2004). Unified Modeling Language: Superstructure, version 2.0.

[OMG, 2008] OMG (2008). Business Process Modeling Notation, v1.1. OMG Available Specification.

[Opdahl and Henderson-Sellers, 2005] Opdahl, A. L. and Henderson-Sellers, B. (2005). A Unified Modelling Language without Referential Redundancy. *Data and Knowledge Engineering (DKE). Special Issue on Quality in Conceptual Modelling*, (277-300).

[Sindre, 2007] Sindre, G. (2007). Mal-activity Diagrams for Capturing Attacks on Business Processes. In *Proceedings of the Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2007)*, pages 355–366. Springer-Verlag Berlin Heidelberg.

[Sindre and Opdahl, 2005] Sindre, G. and Opdahl, A. L. (2005). Eliciting Security Requirements with Misuse Cases. *Requirements Engineering Journal*, 10(1):34–44.

[van Lamsweerde, 2001] van Lamsweerde, A. (2001). Goal-Oriented Requirements Engineering: A Guided Tour. In *Proceedings of the 5th IEEE International Conference on Requirements Engineering (RE'01)*, page 249, Washington, DC, USA. IEEE Computer Society.

[van Lamsweerde, 2004] van Lamsweerde, A. (2004). Elaborating Security Requirements by Construction of Intentional Anti-models. In *Proceedings of the 26th International Conference on Software Engineering (ICSE'04)*, pages 148–157. IEEE Computer Society.

[Vraalsen et al., 2007] Vraalsen, F., Mahler, T., Lund, M. S., Hogganvik, I., den Braber, F., and Stølen, K. (2007). Assessing enterprise risk level: The CORAS approach. In Khadraoui, D. and Herrmann, F., editors, *Advances in Enterprise Information Technology Security*, pages 311–333. Idea group.

[Yu, 1997] Yu, E. (1997). Towards Modeling and Reasoning Support for Early-phase Requirements Engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE'97)*, page 226. IEEE Computer Society.