Designing secure software systems

Combining goal-oriented modeling and risk management

Author:

Nikolaos Argyropoulos

MSc Business Informatics

Student number: 3954420

n.argyropoulos@students.uu.nl

Supervisors:

Dr. Fabiano Dalpiaz

Dr. Marco Spruit

October, 2014

Utrecht University

Faculty of Science

Department of Information and Computing Sciences

Princetonplein 5, De Uithof,

3584 CC Utrecht

Universiteit Utrecht

Abstract

Software systems are broadly used to support the provision of e-services and the facilitation of business processes. Sensitive information is exchanged within such systems between human actors and software agents. As a consequence, their design should encompass security aspects in addition to functional ones, in order to provide an environment in which the users can achieve their goals while keeping their information secure. By reviewing the literature of the areas of security requirements engineering and risk management and surveying practitioners of the field, we identified the need for a structured approach that leads to *security by design*, taking into account the system's extended socio-technical environment and managing risk, from the early stages of the development lifecycle. In this work we develop a structured method to integrate elements of risk management in the security requirements engineering process. By combining method fragments from established methods in the field of security requirements and risk management we construct a new method that uses the results of the risk evaluation and prioritization as input for the identification of user's security needs, creating a complete socio-technical model of the system to-be. We apply this method in practice via a retrospective case study, in order to evaluate its completeness and performance. This application of our method in practice shows promising results, as the method is able to accurately model the studied system and uncover a number of previously unidentified security requirements.

Keywords: *Security requirements engineering, risk management, socio-technical systems, information security, risk analysis, threat prioritization*.

Acknowledgements

I would like to thank my supervisors, Fabiano and Marco, for their support and guidance throughout this thesis project. I would also like to thank Michiel for the time he dedicated and the help he provided for the implementation of the case study. The contribution of the all the survey responders and the stakeholders of the studied system should also be acknowledged as it was instrumental for the successful completion of this work. Finally, I would like to thank my family and friends for their continuous and unconditional support.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Problem statement

Nowadays, information systems (IS) are essential for the provision of e-services and the facilitation of business processes (O'Brien, 2002). Actors interacting with such systems have important assets at stake (e.g., private information) so it is of utmost importance that the security of these assets is guaranteed (Youseef &Liu, 2012). The software that makes such information systems functional, often lacks the appropriate security features (Verdon & McGraw, 2004), thus exposing the systems to a number of threats with costly implications for individual participants and organizations (Whitman, 2003). Therefore it becomes apparent that risks should be assessed and security features should be carefully selected during the design of software systems in a structured and systematic manner.

The elicitation and analysis of security requirements is an essential part in the requirements engineering process for the design of secure software systems. Security requirements engineering (Haley, Laney, Moffett & Nuseibeh, 2008), promotes the adoption of a systematic process for identifying, analyzing, and specifying the security requirements for a system to-be. This contrasts with ad-hoc approaches where security mechanisms are chosen on the basis of personal experience or latest technological trends, and without considering the reason why a given security mechanism is adequate. It also encourages the timely consideration of security during the early phases of software development. By considering security in the early development stages, rather than implementing security measures as an afterthought on an already designed system, can lead to more robust system designs that will not require costly readjustments during their lifecycle (Mouratidis, 2011).

Another aspect to account for, during system design, is the interplay between the multiple actors and software systems, which exist in the context where the to-be secure system will be deployed. This social aspect of the analysis can help identifying threats that could not be otherwise predicted, especially when designing the system considering only its technical aspect. Such systems with complex interaction between autonomous participants and software applications are known as socio-technical systems (STS) (Paja, Dalpiaz & Giorgini, 2013a). As the complexity of such systems grows with the addition of more actors and software agents, ensuring their security becomes a complex task which cannot be handled by traditional security requirements engineering methods. An answer to this challenge could be the use of goal-oriented requirement engineering methods (Van Lamsweerde, 2001), (Eric, Giorgini & Maiden, 2011) as suggested by literature.

A number of methods has been developed which use goal models to represent and analyze the objectives of participating actors, taking into account both the technical and the organizational aspects of the system. With the assistance of support tools developed for such methods, the modelling and analysis of STSs should lead to a more efficient and complete identification of their security requirements. Unfortunately, these methods do not provide explicit primitives to represent how security requirements are originated from the risks that threaten the stakeholders' assets.

It is important to systematically identify and analyze those threats in order to better understand the underlying risks involved for the actors and the assets of the system. Various risk management methods have been developed to "*identify, control, and minimize the impact of uncertain events*" (Peltier, 2005). However these approaches to risk management require, at least, a structural description of the to-be system in order to identify possible targets for an attack (Fabian, Gurses, Heisel, Santen & Schmidt, 2010). A high level overview of the system's design, which is an important prerequisite for risk analysis (Verdon & McGraw, 2004), can be provided once the requirements dictating the basic system's functionalities have been identified during the early stages of the development lifecycle. It is, therefore, important for effective risk analysis, to be considered not as an isolated process, but as an integral part of an organization's system development lifecycle (Peltier, 2005).

The problem that this thesis addresses is how to facilitate this integration between risk analysis - the first step in the risk management process - and goal-oriented approaches for security requirements engineering. We expect that this integration will enable (i) identifying threats in the very early stages of system design, and (ii) justifying why security requirements are posed on the system to-be.

1.2 Research questions

This thesis aims to study security requirements engineering and risk management in software systems, explore the gap between the state-of-the-art and the state-of-the-practice in this field and attempt to bridge it with the creation of a new approach. This means that we will focus our attention in two separate fields of study (security requirements engineering and risk management), each with its unique characteristics and research communities, and attempt to bring them together. In order to accomplish those research goals, research questions must be defined which will be decomposed further to sub-questions. An overview of the research questions of this thesis is provided in Table 1.1.

RQ1	*What does the literature suggest for identifying risks and security requirements in software systems?*
SQ 1.1	Which are the state-of-the-art methods for security requirement engineering?
SQ 1.2	Which are the state-of-the-art methods for risk analysis in software development?
SQ 1.3	What shared insights and empirical evidence can be found in the literature of those fields?
RQ2	*How are risks assessed and security requirements identified and specified in practice?*
SQ 2.1	When, how and by whom is security and risk elaborated during software development in practice?
SQ 2.2	Which factors influence the selection of a method for security requirements engineering and risk management, to be used in practice?
SQ 2.3	What is the gap between the state-of-the-art and current practices used for identifying risk and specifying security requirements?
RQ3	*How can we devise an effective method that integrates risk management in the security requirements engineering process?*
SQ 3.1	How can the findings of RQ1 and RQ2 contribute to the development of the new method?
SQ 3.2	Is this method effective and applicable enough to assist the security requirements definition and risk analysis process in real life settings? What are the limitations?

Table 1.1: The research questions of this thesis

RQ1 will provide insights on the scientific literature involved in the field of security requirements engineering and risk analysis of software systems. Various established methodologies will be identified and analyzed in order to extract relevant concepts and techniques, while also best practiced described by the same literature will be collected.

In RQ2 the state-of-the-practice will be explored through a survey which will be used to gather expert opinions by researchers and practitioners of the field of software development. Software security and risk will be the main theme of the survey which will provide insights about who is involved, when does it take place and what factors influence the decision making during the software development lifecycle. The conclusions drawn by these findings will be compared with the previous literature findings to identify what gaps exist between the two sides.

Finally, for RQ3 a new method will be assembled which will integrate risk analysis elements in an existing security requirements engineering approach. For the final research sub-question (SQ 3.2) a case study will be designed which will involve the application of the constructed method, in the real life settings of a software system. The participants of this case study will provide validation of the completeness and effectiveness of the method while also some criteria will be identified in order to assess the contributions and limitations of the method in a practical environment.

1.3 Research approach

In order to tackle the research questions formulated in the previous section, a hybrid approach will be adopted. A high level overview of that approach is presented in the Figure 1.1 and described in the rest of this section. Further elaboration of the method followed for each step of our research will be provided in a separate section at the beginning of each chapter (cf. Sections 2.1, 3.1, 4.1 and 5.1).

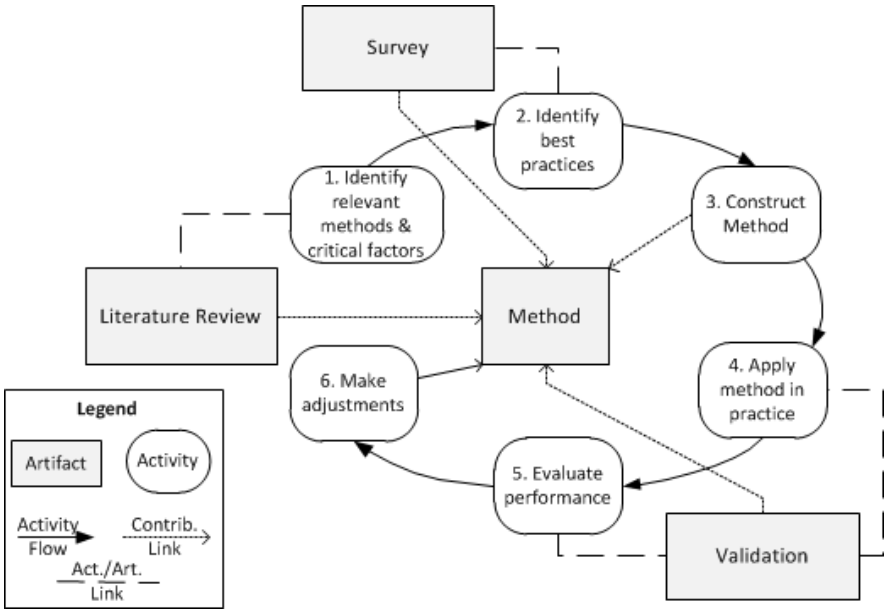


Figure 1.1: Research approach model

Initially the state-of-the-art in the fields of security requirements engineering and risk analysis methods for software systems, needs to be identified. Through an exploratory literature review a number of established methods in both fields will be identified. Literature reviews on security requirements engineering (e.g., Mellado et al., 2010) and risk analysis methods (e.g., Vorster & Labuschagne, 2005) provide a starting point from which a set of methods can be selected for further analysis. In addition to that, shared insights and best practices regarding security definition and risk analysis presented in the selected literature will be collected. Later during our research, by comparing those findings with the state of practice, misalignments and gaps between theory and practice will be identified.

In order to explore how security and risk are elaborated during software development in practical settings, a survey will be distributed among researchers and practitioners of the field. The survey will aim to identify the level of abstraction at which professionals consider security when developing software systems, the roles

involved, the methods selected and the factors that guide the decisions made during the process. It will also include questions regarding risk assessment and asset identification. The structure of the survey will allow participants to both indicate what happens in practice and also evaluate certain aspects of the process. This will allow us to get an overview of the practices of the software industry and the consensus among professionals working in this field. The results of the survey will then be compared with the previous literature findings. This way the differences and common points between what is described in theory and what happens in practice, can be identified and studied.

During the next part of the study, method fragments and techniques identified in the state of the art will be assembled to a comprehensive method that will incorporate risk analysis elements in the security requirement elaboration process. The methods assembly process will follow the framework established by Brinkkemper et al. (1999) for the construction of situational methods from a collection of relevant method fragments. The best practices and insights gained from the survey will help refine the method so it can be adjusted according to the needs of practitioners.

Once a coherent and satisfactory method is created it should be validated through a retrospective case study in a real life software system. Case studies are widely used for qualitative research in the field of IS as they allow researchers to closely examine how their innovation interacts with organizations in practical contexts (Darke, Shanks, & Broadbent, 1998). For this case study, the application of our method will include the modeling of the system and the identification of the security requirements resulting from its analysis. During the process an iterative feedback loop will ensure that modifications and adjustments are made to the method in order for it to optimally reflect the experiences observed in practice. Since the studied system will be already designed and operational, the requirements produced by the application of our method will be compared with the already implemented functionalities of the system in order to assess our method's added value. Finally a selection of both quantitative (e.g., questionnaires) or qualitative (e.g., interviews) methods can be used to capture the opinion of the involved stakeholders and their experiences regarding the added value and limitations of the developed method in the construction of secure software.

In Figure 1.2 a process-deliverable diagram (PDD) is introduced to summarize the sequence of activities and deliverables of our research attempt. A PDD is a meta-modeling technique which integrates two diagrams, a meta-process model of the activities on the left side and a meta-deliverable model of the deliverables on the right. The rules and notation used for the construction of PDDs, as introduced by van de Weerd and Brinkkemper (2008), dictate that the concepts on the right side of the diagram are connected with the activities at the left side, from which they

derive. Process-Deliverable diagrams can provide a comprehensible illustration of structured methods and will be used throughout this research for this purpose.

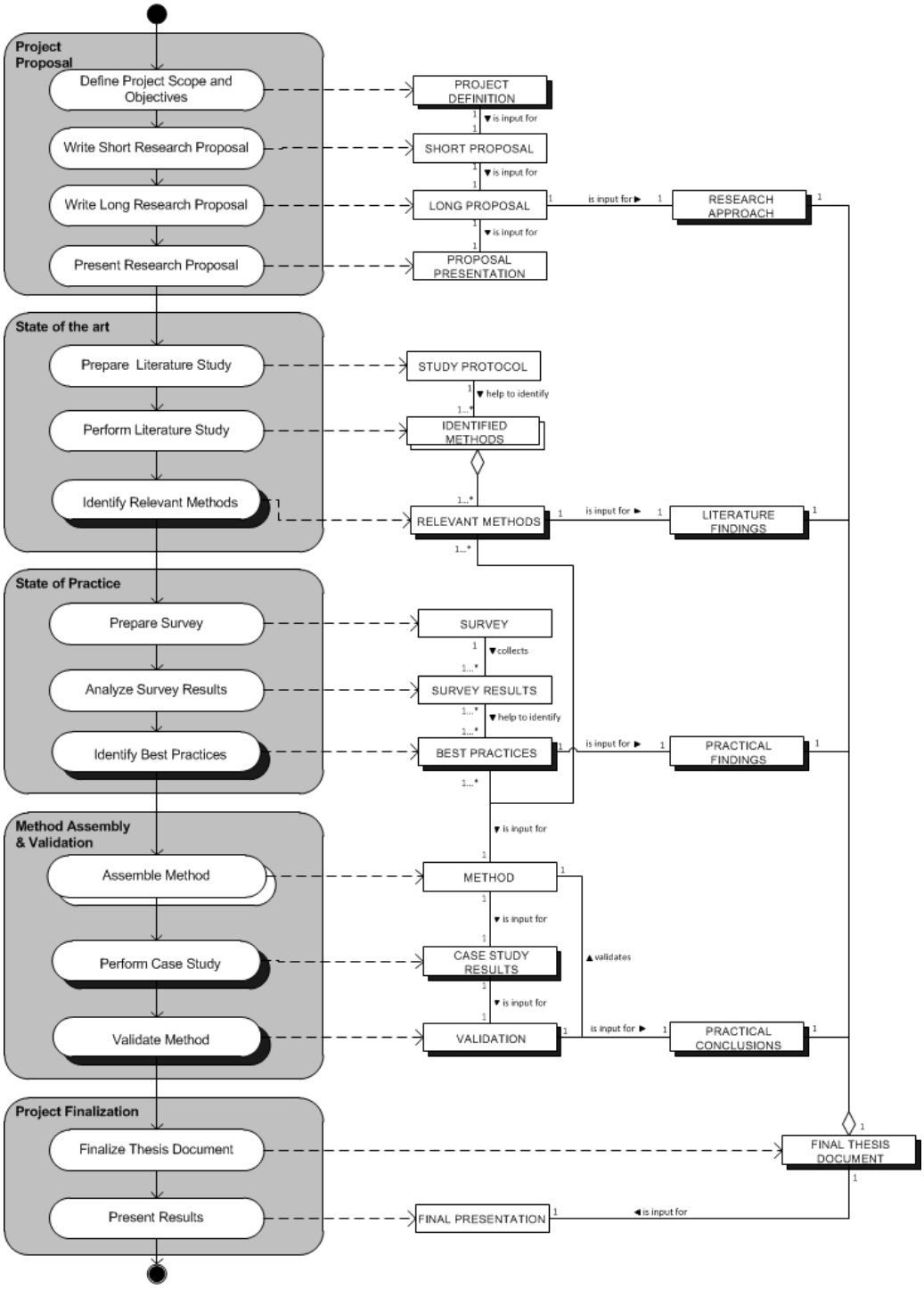


Figure 1.2: PDD describing the steps and deliverables of this research

1.4 Relevance

From a societal point of view, the implications of secure software systems are multifaceted. Today's software systems handle sensitive information (e.g., healthcare, financial data) and perform critical operations (e.g., banking transactions, public services) with costly implications when they fail to perform as designed. Due to the plethora of applications of software systems in everyday life, the need for secure interactions with them is as prominent as ever. In this context, user security can include a number of criteria that should be met by the software system. Literature sources indicate that confidentiality, authentication, integrity, access control, availability and non-repudiation are the main factors that define security (Mouratidis & Giorgini, 2007). All those different aspects will be taken into account when designing our method in order to ensure the security of the users and the integrity of the business processes supported by such systems.

Modern organizations heavily depend on software systems in order to conduct their main business processes. The optimization and stability of those processes can be a crucial factor for organizations seeking a competitive advantage in their field. It is therefore of great importance that unauthorized or malicious interactions that can disrupt those critical processes are prevented by security measures designed into the systems that support them. It is well reported in scientific literature that when security is implemented as an afterthought in such systems, it often leads to inconsistencies and conflicts that create vulnerabilities (Mouratidis, Giorgini & Manson, 2003). As a result, the organizations developing and maintaining those systems have to spend additional time and money, redesigning them (Kim, Kim & Park, 2005). In our work the designed method will support security elaboration during the early stages of the development lifecycle in order to prevent the costly implications of phenomena like the previously discussed.

Finally, from an academic standpoint, this thesis will contribute to the state of the art of security requirements engineering methods. By designing and testing such a method useful evidence and best practices will surface which can contribute to the literature of the field security in multi agent software systems. In addition to that, the responses gathered from the survey of practitioners will also result in quantitative data on the state of practice which can provide valuable insights for future works on the field.

1.5 Outline

The rest of this work is structured as follows; first, in *Chapter 2*, we present the literature findings and introduce established methods in order to provide an overview of the state of the art in the fields of security requirements engineering and risk management. Next in *Chapter 3*, the structure of the survey and the results gathered from it will be discussed and analyzed in order to identify the state of practice in the field of software security. Combining the findings of these two sections, in *Chapter 4* we will present a new approach that combines elements of security requirements engineering and risk management. Next, the newly introduced approach will be tested in practical settings through a case study. The design of the case study and its results will be presented in *Chapter 5*. The last sections (*Chapters 6*) will include our conclusions regarding the research questions of this research and discussion of the limitations, as well as future directions of this work. An overview of the structure of the remainder of this work is provided in Figure 1.3.

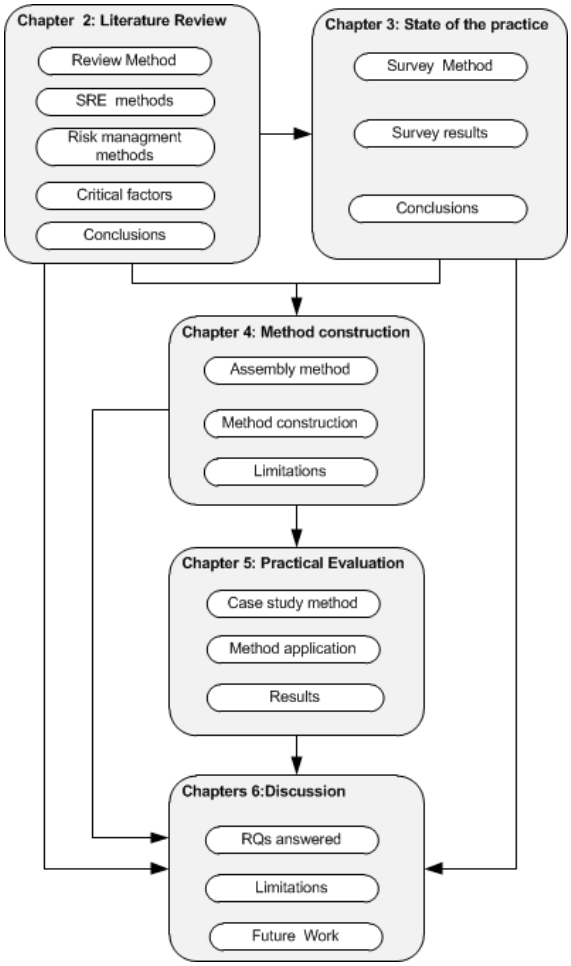


Figure 1.3: Outline of the research project

Chapter 2. Literature Review

In this section we analyze the state of the art of the literature areas of security requirements engineering and risk management. We begin with a brief overview of the method followed for this literature review; focusing on literature sources and filtering criteria applied. Next we will present and discuss the literature findings, focusing on the identified state-of-the-art methods, for each of which a small overview will be provided. In the final part of the section a discussion will take place, overviewing conclusions drawn from the literature of this area. Critical factors, techniques and methods which can further assist our research will be identified and our selection criteria will be elaborated.

2.1 Method

This exploratory literature study will follow the *snowball method* (Streeton, Cooke & Campbell, 2004) which allows the researchers to identify relevant literature for a field of study by gathering and filtering the works referenced by recent important literature on the research topic. This particular method, also known as *reference tracking*, was selected for this literature study due to its documented capability to efficiently identify literature sources that may be missed by often time-consuming predefined protocol-driven studies (Greenhalgh & Peacock, 2005). Initially the topic and research objectives should be defined. In our case the two topics are *"security requirements engineering"* and *"risk analysis"* methods and our research objective is to identify systematic reviews on those topics which will provide an inventory of methods to be studied. The initial selection of literature will be screened by title, abstract and year of publication so only recent and relevant studies are selected. Eligible data sources for this selection are journal and conference papers published within the last twenty years, available in full text by the electronic library subscription of Utrecht University searched via Google Scholar.

2.2 Security requirements engineering (SRE)

The area of interest for this research is software Requirements Engineering (RE) and especially the sub-area of Security Requirements Engineering (SRE). The field of requirements engineering is described by literature as "the branch of software

engineering concerned with the real-world goals for, functions of and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families" (Zave, 1997). Security has been considered as a non-functional requirement in traditional requirements engineering (Chung & Nixon, 1995). Non-functional requirements (NFRs) are often challenging to identify due to their qualitative nature, and thus difficult to accurately measure the degree of their satisfaction by a proposed system design. Therefore, by expressing security exclusively through the abstract construct of NFRs can lead to suboptimal results.

The consensus amongst researchers - e.g., (Baskerville, 1992), (Liu, Yu & Mylopoulos, 2003), (Youseef &Liu, 2012) - is that in order to effectively built secure software, it is of major importance to incorporate security in the early stages of the development process in a structured and iterative manner. The sub-discipline of Security Requirements Engineering provides such techniques, methods and standards for coping with security requirements, therefore tackling the issue of security throughout the IS development cycle (Mellado, Fernandez-Medina & Piattini, 2007).

Model-based security requirements engineering is a prominent approach to develop security critical software (Best, Jurjens & Nuseibeh, 2007). The construction of models during the early stages of the development lifecycle (e.g., requirements and design phases) contributes towards the creation of high quality system designs (Basin, Doser & Lodderstedt, 2006). Models allow for conflicts and design flaws to be identified and handled early in the development process thus providing formal specifications for the later development phases of the system. An advantage of using model-based approaches for the elicitation and elaboration of security requirements is the automated model checking capabilities of CASE tools. Such automated support tools can analyze a created model and check its consistency and completeness, with speed and precision that cannot be achieved by manual analyses of the models by security experts. Models can also "*be used to abstract away irrelevant details, rigorously specify the interplay between security and functional requirements, and provide a basis for analysis and transformation*" making them the "cornerstone" of software and system development (Basin, Clavel & Egea, 2011).

2.2.1 UML-based methods

As previously discussed, traditional modeling approaches to security requirements engineering include a number of established methods often supported by automated analysis tools that guide the process throughout the software development lifecycle. A number of such methods have originated from the Unified Modeling Language (UML) (Rumbaugh, Jacobson & Booch, 2004) either by using UML techniques in the

context of security elicitation (e.g., use cases) or by expanding UML concepts to cover security related elements of the designed system.

A positive aspect of UML-based approaches for security is the fact that they are based on proven object-oriented modeling techniques, making them easier to understand by the project developers compared to complex mathematical security models. Another advantage is the intuitive nature of the created models which makes them comprehensible by non-technical personnel such as management or end-users of the system, therefore enabling their participation in the process of threat identification and security elaboration. While such methods present certain advantages for modeling simple user-system interaction they become harder to use when the size and complexity of the designed systems grow (Liu, Yu & Mylopoulos, 2003). Another general limitation of use-case based techniques, as observed by Mouratidis (2011), is the lack of support for security analysis in the social level, since their focus is on the system level. Therefore a broader system perspective must be included as part of security requirements engineering methods which should be able to support social analysis and modeling (van Lamsweerde, 2004).

Abuse case modeling (McDermott & Fox, 1999)

Abuse case modeling was introduced by McDermott and Fox as an adaptation of use cases to capture and analyze security requirements. The definition which the authors provide is the following:

"We define an abuse case as a specification of a type of complete interaction between a system and one or more actors, where the results of the interaction are harmful to the system, one of the actors, or one of the stakeholders in the system." (McDermott & Fox, 1999)

Figure 2.1: Process diagram of abuse case modeling

The goal of an abuse case is to describe the abuse of privilege used to complete an activity (or group of activities) harmful to the system. Use case diagrams and use case descriptions are borrowed from use case modelling to also describe abuse cases. No additional notation or extra symbols need to be introduced. The concepts of actors, objectives and scenarios are used to model an abuse cases and provide their description.

Based on the proven object-oriented modeling technique of use cases, abuse case modelling is easier to understand by the project developers compared to complex mathematical security models. Additionally stakeholders such as end users and customers can also participate in the modelling process since the models can be intuitively understood with limited experience in UML or case modeling. All the above contribute to the reputation of abuse case modelling as a useful complimentary tool to support different phases of the development process.

A limitation of abuse case modelling, as mentioned earlier, is that it is not a substitute for any part of the security engineering process but rather a tool which complements each of its steps. Another general limitation of use-case based techniques, as observed by Mouratidis (2011), is the lack of support for security analysis in the social level, since their focus is on the system level.

Misuse Cases (Sindre &Opdahl, 2005)

Similar to abuse case modelling, misuse cases were introduced as an extension of use case modelling in order to specify unwanted behavior for the developed system in order to elicit security requirements. The creators of this method, Sindre and Opdahl define misuse cases as follows:

"A sequence of actions, including variants that a system or other entity can perform, interacting with misusers of the entity and causing harm to some stakeholder if the sequence is allowed to complete." (Sindre & Opdahl, 2005)

A misuser, who represents the actor that initiates a misuse case, can be associated with other misusers and their misuse cases using relationships borrowed from use case modelling (e.g., include, extend and generalize). Use-case diagrams and lightweight or extended textual descriptions are used to capture a misuse case. A five step process is proposed by the creators of the method in order to elicit security requirements. The process begins with the identification of the critical assets that need to be protected. Next the security goals to be achieved are defined, followed by the identification of potential threats. The identification and analysis of risks follows and the method is concluded by defining the security requirements for the system to-be.

One of the positive contributions of misuse cases is that they enable the organization, prioritization and traceability of security requirements, while they can also be reused at different implementations or projects. Another advantage of the method is the informal nature of its models which promote the participation of stakeholders without extended technical knowledge at the process of threat identification.

Limitations of misuse cases include the lack of a precise set of guidelines for their definition which renders them unsuitable for certain kinds of threats especially when a large number of critical assets are involved. The method also fails to provide guidance on when and how identified security issues can be tackled and how the produced security requirements can be linked to the rest of the development process.

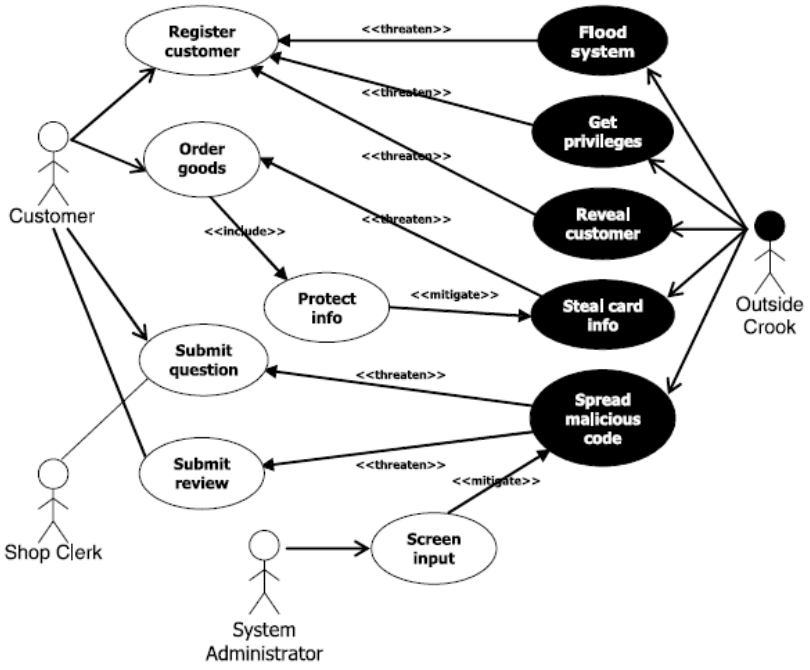


Figure 2.2: Example of use and misuse cases (Source: Sindre & Opdahl, 2005)

Finally, when comparing misuse and abuse cases, Sindre and Opdahl (2005) find abuse cases as a complementary approach to misuse cases especially since both methods focus on different phases of the development lifecycle. Misuse cases are more effective during the elicitation process whereas abuse case modelling is more focused on the design and testing phases of the development. They also notice that contrary to abuse case modelling, misuse cases are modeled in the same diagram as the use cases, which allows potential relationships between the two to be easier to identify.

Anti-Models (van Lamsweerde, 2004)

Anti-Models were introduced by van Lamsweerde (2004) as a method for elaborating security requirements. The purpose of the method, as explained by its creator, is the following:

"Our approach is intended to provide constructive guidance in early elaboration of security concerns; it supports incremental reasoning on partial models and formal derivation when higher assurance is needed; alternative threats and countermeasures may be modelled explicitly." (van Lamsweerde, 2004)

The method is based on the idea of modelling the two different aspects of the system under development in an iterative and concurrent process. One model is concerned with the interrelations between goals, agents, objects, assumptions and requirements of the system-to-be. The second is an anti-model of the same system which is concerned with threats arising from certain elements of the proposed design, as well as whom and why may use them for malicious purposes. An iterative sequence of activities aiming at producing security requirements is proposed by the creator of Anti-models. This approach is described by the following steps:

a) Instantiate specification patterns associated with property classes such as confidentiality, privacy, integrity, availability, authentication or non-repudiation.

b) Derive anti-model specifications threatening such specifications.

c) Derive alternative countermeasures to such threats and define new requirements by selection of alternatives that best meet other quality requirements from the model.

UMLsec (Jurjens, 2002)

UMLsec is an extension of the Unified Modelling Language (UML) introduced by Jujens (2002), which allows developers to embed security related information and conduct security analysis for the system to be (Best, Jurjens & Nuseibeh, 2007) .

UMLsec uses already established UML diagrams to model different security aspect of the system under development, such as integrity or confidentiality. An example of this property of UMLsec is given by Melado et al. where it is noted that "*state chart diagrams model the dynamic behavior of objects, and sequence diagrams are used to*

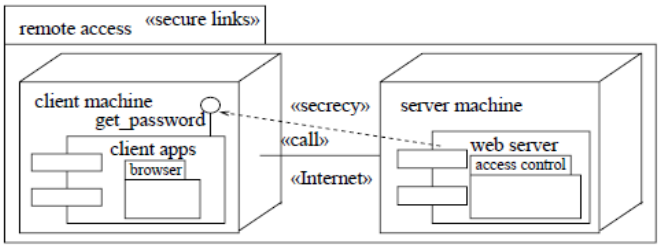


Figure 2.3: Example of "secure links" stereotype of UMLsec (Source: Jurjens, 2002)

model protocols […] deployment diagrams are also used to model links between components across servers." (Melado et al., 2010). Security requirements are formulated using UML mechanisms such as stereotypes and tags. Constraints give criteria that determine whether the requirements are met by the system design, by referring to a precise semantics of the used fragment of UML (Best et al., 2007).

Since using UML techniques and diagrams is a common practice among developers, it is easier for them to incorporate UMLsec in the development process even when they lack expert knowledge on the security domain. Another valuable aspect of the UMLsec approach is its ability to contribute in security specification from the early steps all the way through the development cycle (Best et al., 2007).

2.2.2 Goal-oriented methods

Goal-oriented approaches to security requirement engineering seem to be better equipped to deal with complex systems that support a combination of software and social actors. As suggested by literature "g*oal-oriented approaches to security requirements engineering seem to be appropriate for designing secure STSs, since they build upon the concepts of intentional and social actors, who have objectives to achieve and interact with others to achieve them.* "(Paja, Dalpiaz, Poggianella, Roberti & Giorgini, 2012). Goals are able to cover both functional (e.g., services to be provided) and nonfunctional concerns (e.g., safety, security, performance) of the design and can be used to model objectives that the system under design should achieve (Van Lamsweerde, 2001). Those objectives, expressed through goals and their decomposition to subgoals, often reflect high level business goals for the system's stakeholders. This way the system requirements can be influenced by the business context while also non-technical stakeholders can be part of their identification process, therefore enabling the alignment between business and IT for the organization (Yu & Mylopoulos, 1998).

Secure i* (Elahi & Yu, 2007)

Secure i* was proposed by Elahi and Yu (2007) as an extension to the i* framework introduced by Liu, Yu and Mylopoulos (2003). Secure i* makes use of the concepts and notation introduced by i* but also adds the concepts necessary for security trade-off analysis. According to the authors, *"the proposed modeling notation is accompanied with a qualitative trade-off analysis procedure based on goal model evaluation methods, which provide the designers with assessment of security mechanisms' impact on actors' goals and threats*." (Elahi & Yu, 2007).

The proposed method uses the already establish notation of the i* framework to model actors, goals, resources and dependencies between them. It allows the definition of actors within a system for whom security and privacy are goals to be achieved. Those high-level goals can later be decomposed into more specific sub-goals that can be achieved by certain implementable mechanisms (e.g., encryption, firewalls) in the system-to-be. Adding to that notation, extra concepts are introduced to model threats and vulnerabilities. For example the vulnerability point modeling element was added to i*, accompanied with a graphical notation to connect a vulnerability point to the corresponding attacks, and to attach it to a resource.

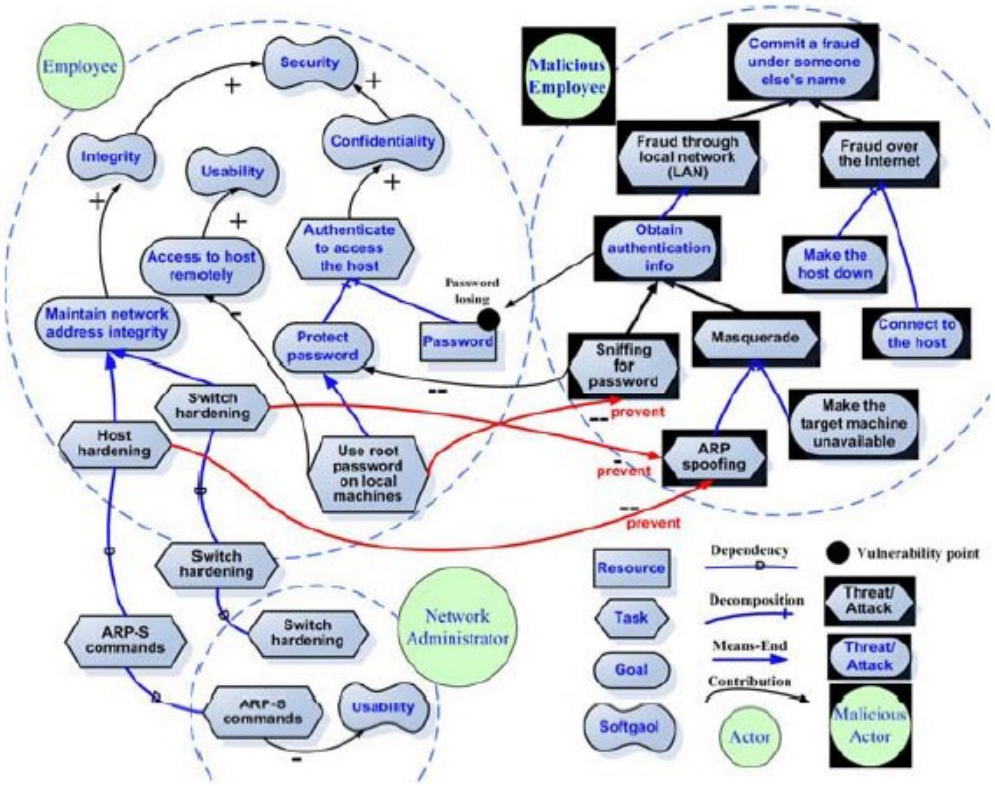


Figure 2.4: Example of Secure i* notation (Source: Elahi & Yu, 2007)

It has been reported that a possible limitation of Secure i* is the scalability of the goal models it creates (Elahi & Yu, 2007). As more goals are added to the model the complexity of the trade-off analysis grows and the model may become too complex and inefficient to use. Another limitation inherent to this approach is the selection of the security mechanisms to be implemented, which is left at discretion of the stakeholders. Since system developers may not always be aware of every security mechanism and its contributions, some risks for the system-to-be may arise.

Secure Tropos (Mouratidis, Giorgini & Manson, 2003)

The Secure Tropos method was introduced by Mouratidis, Giorgini and Manson as an extension of the Tropos methodology (Giunchiglia, Mylopoulos, & Perini, 2003) in order to focus on the elicitation of security requirements throughout the development stages. Using the rules and notation established by Tropos as a basis, the creators of Secure Tropos introduced new concepts focusing on security. As a result Secure Tropos makes use of the Security Diagram which "*represents the connection between security features, threats, protection objectives, and security mechanisms that help towards the satisfaction of the objectives*" (Mouratidis, Giorgini & Manson, 2003). This type of diagram makes use of other novel concepts such as Security Constraints, Secure Entities, Dependencies and Capabilities in order to illustrate goals, tasks and resources of the system, their interdependencies, constraints imposed on them and their capabilities to contribute towards security.

The modelling process begins with the creation of the Security Diagram where the security needs of the system to-be are displayed along with possible security problems and their solutions. Next the Security Constraint modelling takes place, where constraints imposed on the system are related to the environment in which the system will function. These constraints may arise from the stakeholders of the system or can be identified through the Security Diagram and can be countered by assigning Secure Capabilities to the Entities of the system (e.g., actors, goals) during the Secure Entities and Secure Capabilities modelling. All the different modelling stages are spread throughout the various stages of the development lifecycle so the whole process, from the early requirements to the design stage, can be supported by the Secure Tropos method. A CASE tool, secTro tool (Pavlidis & Islam, 2011), has been developed to support the modelling process, thus aiding the adoption and usage of the Secure Tropos method in practical settings.

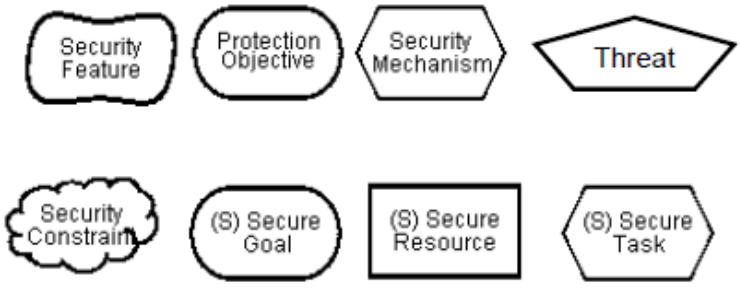


Figure 2.5: Example of Secure Tropos notation (Source: Mouratidis, Giorgini & Manson, 2003)

According to the creators of the method, its main advantage is "*the iterative nature of the methodology, [which] allows the re-definition of security requirements in*

different levels therefore providing a better integration with system functionality" (Mouratidis, Giorgini & Manson, 2003). Secure Tropos also offers developers the capability to reason about security issues from both a technical and social point of view, in different stages of the development process, while also providing validation that the proposed design satisfies the initial requirements (Mouratidis, 2011).

STS-ml (Dalpiaz, Paja & Giorgini, 2011)

The Socio-technical Systems modelling language (STS-ml) is an actor- and goal-oriented security requirements engineering framework (Paja, Dalpiaz & Giorgini, 2012) which makes use of similar concepts with the Tropos and Secure Tropos methods to model actors, goals, security needs, delegations etc. In STS-ml the system to-be is modelled through three different perspectives which create a representation of the system that includes a social, an information and an authorization view. Those three complementary views of the system make possible for the system designer to analyze the various interactions between actors and information from different perspectives and according to the security needs unique to each actor or type of interaction.

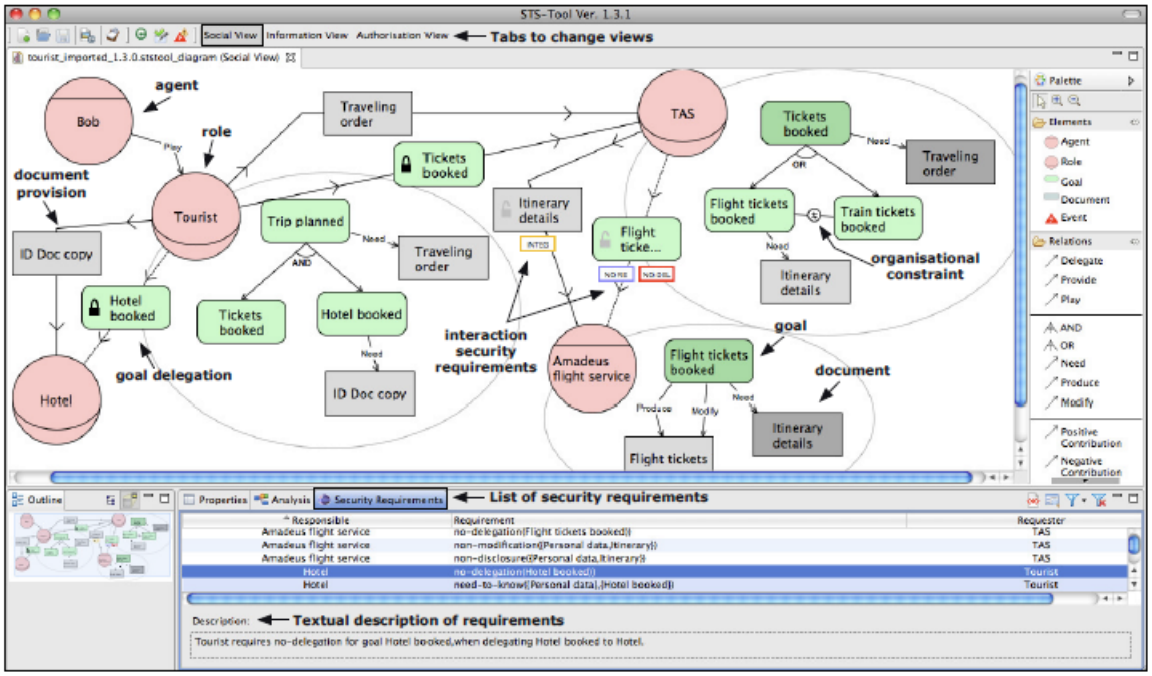


Figure 2.6: Example of social view model created by the STS-tool (Source: Paja, Dalpiaz & Giorgini, 2013b)

The first step in the modelling process is the creation of the social view of the system. In this view the main actors and their goals within the system are identified. Each actor has a personal set of goals which can be decomposed to a sub-goal tree, part of which can be delegated to other actors within the system in order

to be accomplished. Delegated goals along with documents containing informational assets are subject to security constraints which are included in this view of the model. In the information view the documents exchanged between actors are decomposed to pieces of information and ownership is assigned. Finally the authorization view represents the authorization that the actors provide over the information they own within the system. The STS-tool (Paja, Dalpiaz, Poggianella, Roberti & Giorgini, 2012) was developed to support this iterative modelling process while also providing automated security requirements derivation and consistency checking to validate the created models.

STS-ml creates flexible system designs where each actor can independently define his own security needs for his interactions with the rest of the system. It also able to handle complex system designs, when used in practical settings, where it can identify security conflicts which may would be hard to identify manually or by traditional approaches (Paja, Dalpiaz & Giorgini, 2012). The automated support provided by the STS-tool is another strong point of the method.

2.3 Risk management

Another point of interest in our research is how to integrate risk management elements to the security elaboration process. Risk management encompasses three processes: risk assessment, risk mitigation, and evaluation and assessment (Stoneburner, Goguen & Feringa, 2002) as it aims to identify risks, represented by vulnerabilities to an organization's assets and to reduce them to an acceptable level. A number of risk analysis methods for software systems can be identified by literature, amongst the most popular are CORAS (Stolen et al., 2002), CRAMM (Yazar, 2002), OCTAVE (Alberts, Dorofee, Stevens & Woody, 2003), IS Risk Analysis based on a Business Model (Suh & Han, 2003) and ISRAM (Karabacak & Sogukpinar, 2005). Some of those methods produce qualitative results which usually represent the level of risk in a "Low-High" scale, while other produce quantitative outcomes in the form of metrics, such as Annual Loss Expectancy (ALE). While quantitative outcomes are usually preferred, as they are considered more solid, they often require a number of mathematical calculations to be performed by analysts dedicated to the process. Qualitative methods may be easier to implement in terms of time and resources, nevertheless they produce more subjective results. Next a number of state-of-the-art risk management methods will be overviewed accompanied by Process Deliverable Diagrams (PDDs) which will outline the main activities and concepts of each identified method.

CORAS

The CORAS framework (Stolen et al., 2002) was created in order to provide risk assessment for security critical systems. CORAS combines elements of different risk management methods, modelling languages and tools to provide model-based risk assessment (Stolen et al., 2002). The Unified Modelling Language (UML) is an important tool used in CORAS as the main means of modelling the system and elaborating on security issues. The CORAS framework, according to its creators, is "*founded on four pillars*", namely: a risk documentation framework, a risk management process, an integrated risk management and development process and a platform of tool inclusion.

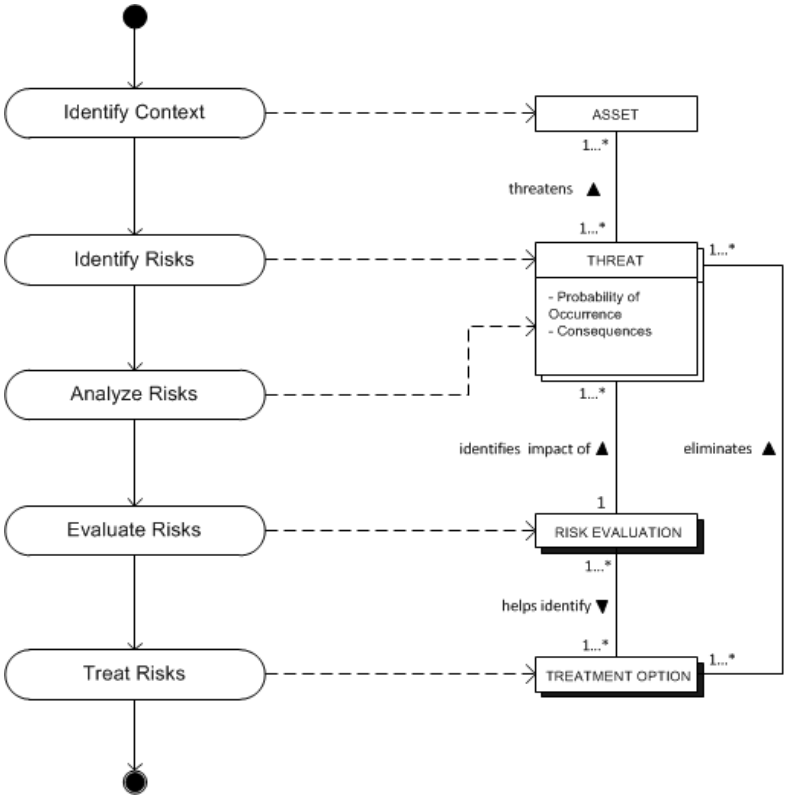


Figure 2.7: CORAS Framework PDD

The risk management process, which is the subject of our interest in this framework, requires the participation of stakeholders and experts in brainstorming sessions in order to, initially, identify the context of the system's extended environment, and potential risks. Next the identified risk factors are analyzed in order to determine their likelihood and their impact in the system. A risk evaluation follows, where identified risks are grouped and prioritized, in order for the experts to provide treat advice as the final step of the process.

CRAMM

CRAMM (CCTA Risk Analysis and Management Method) is a qualitative risk analysis and management tool developed by UK government's Central Computer and Telecommunications Agency in 1985 (Yazar, 2002). The first step in the CRAMM risk management process ("Initiation") is concerned with data collection about the system through questionnaires, interviews and meetings with its stakeholders and users. Next important informational assets (data, software, physical assets) are identified and evaluated by the users of the system, "values are

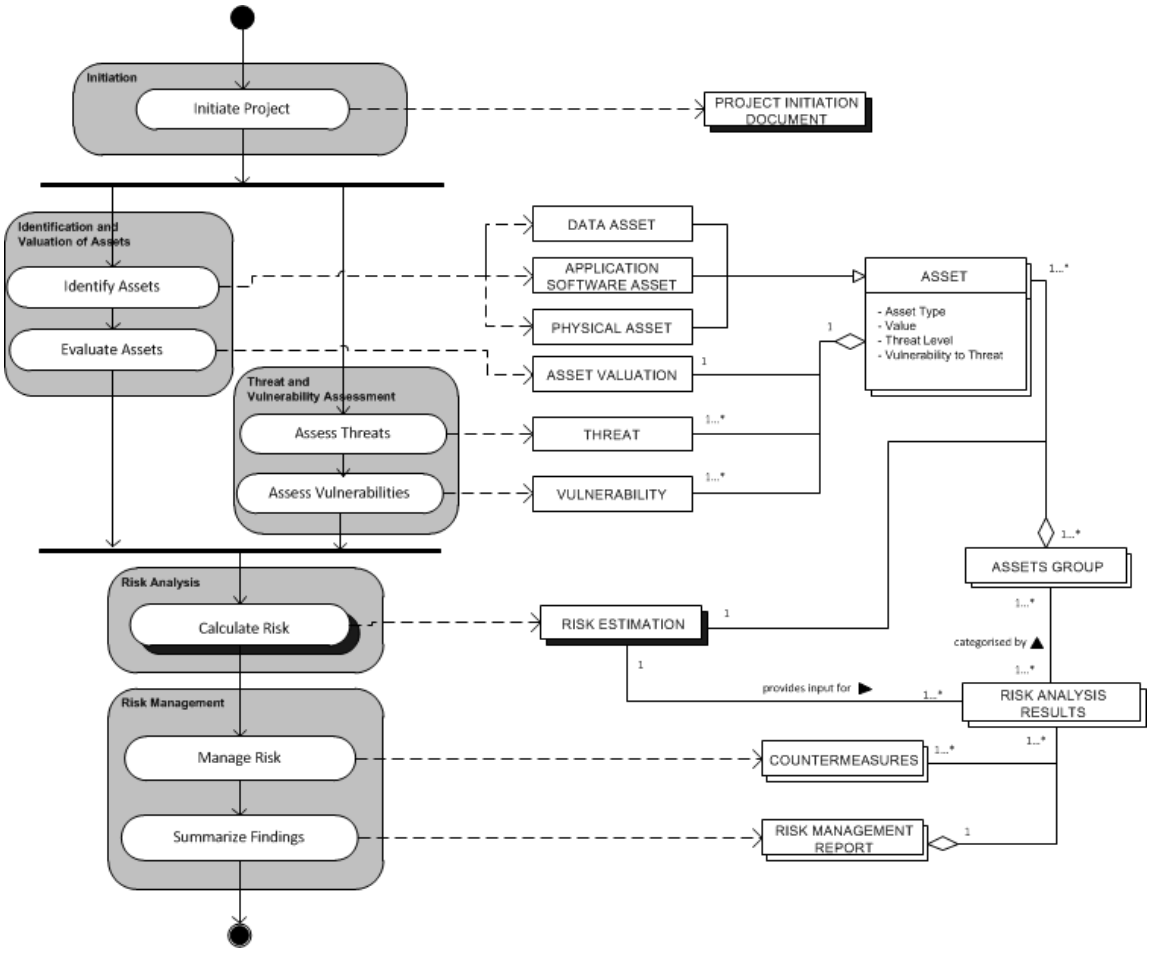


Figure 2.8: CRAMM PDD

derived from the impacts of breaches of confidentiality, integrity, availability and non-repudiation" (Yazar, 2002).

Threats and vulnerabilities assessment follows, where security analysts with the input of the users assess the level of threat to assets and the level of vulnerability to threats in a "Low – High" scale. The next step is the Risk Calculation, where each group of assets receives a risk value for each threat it is potentially vulnerable to. Similarly to the previous step a seven point scale scoring system is used for the

calculation of each assets group risk value. Finally the Risk Management phase produces a risk management report which, based on the risk analysis findings, describes what countermeasures are required to protect the system against the identified threats. Overall, CRAMM is a structured approach which promotes user involvement in the risk management process and provides quantifiable results, comprehensible to non-technical users. On the other hand a number of experienced security experts are necessary for the successful completion of the CRAMM process, which is highly documentative and requires a large timeframe.

OCTAVE

OCTAVE (Alberts, Dorofee, Stevens & Woody, 2003) is a qualitative risk evaluation method aiming to assist organizations in making strategic security decisions. The process is asset-driven, iterative and can be self-directed by already existing

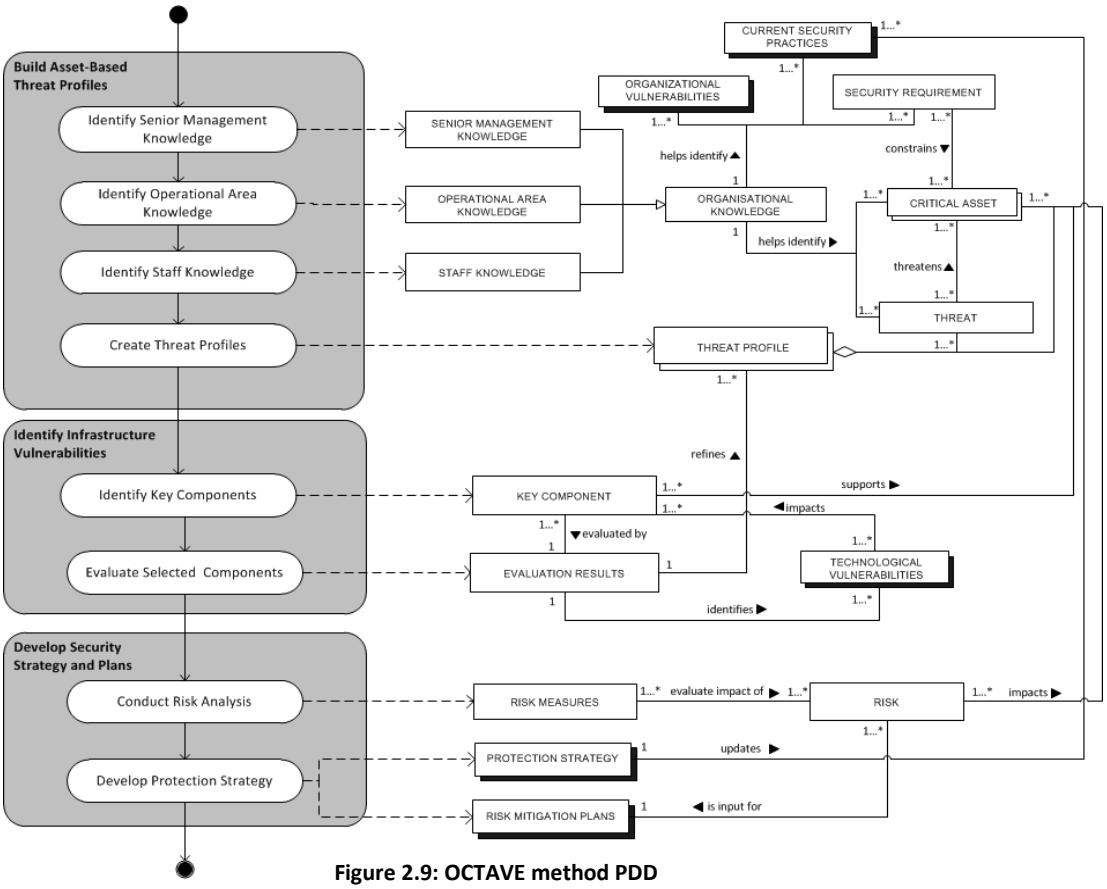


Figure 2.9: OCTAVE method PDD

technical and managerial personnel within the organization. The first main phase deals with the creation of "Asset-based Threat profiles" where the most important informational assets of the organization are identified and potential threats for each of them are listed. In the second phase the physical and technical infrastructure of

the organization is evaluated for vulnerabilities that can threaten its critical assets. In the final phase the impact value for the identified threats is given a qualitative value from a "Low-High" scale, according to different evaluation criteria. As a result the security strategy and risk mitigation plans of the organization are formulated based on the input gathered from the previous analysis. According to the creators of OCTAVE, it does not require external assistance for its completion except from the documentation and form templates provided by the method. This makes the process flexible and easier to implement while also raising the security awareness of the participating organization members. On the other hand the qualitative nature of the method's output coupled with the subjective evaluations of the project's team used as input, may impact the overall quality of the result.

ISRAM

ISRAM (Karabacak & Sogukpinar, 2005) is a quantitative risk analysis method focused on assessing the security risk of an organization through paper-based surveys. The process of ISRAM is focused on creating surveys with weighted questions that when completed, can give a clear overview of the as-is situation of the organization. Initially, potential security problems are identified by the security analysts in charge of the ISRAM process. Those security issues are the focus of the survey questions which attempt to identify the probability and the consequences of their occurrence. When the survey results are collected risk tables are used to calculate a risk value for each threat factor, based on the survey answers provided by the users. Finally all the individual risk values are inputted in the risk calculation formula used by ISRAM which calculates a single qualitative risk value in a scale of 1 to 25, which is the main outcome of the process. Overall, ISRAM provides qualitative results sourcing from user input which promotes involvement and security awareness. Nevertheless it only partially covers the risk management process as it is mainly concerned with identifying potential risks but not elaborating on their management with countermeasures. It also requires experienced security analysts and a wide timeframe to create, distribute and analyze paper-based surveys.

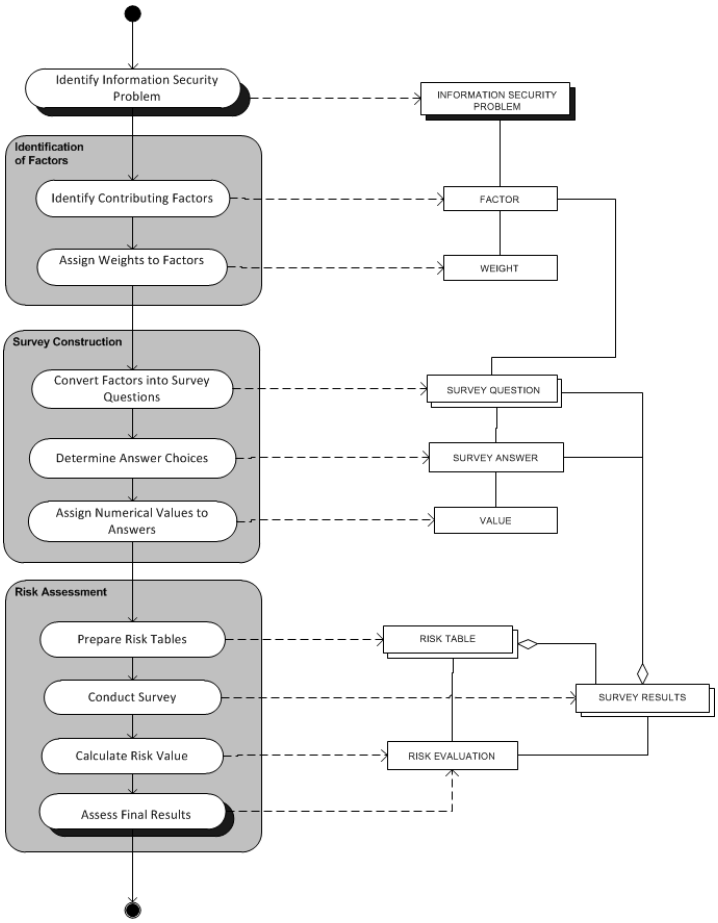


Figure 2.10: ISRAM PDD

ISRA-BM

The IS Risk Analysis based on a Business Model (ISRA-BM, Suh & Han, 2003) is a quantitative risk management method where "the importance level of various business functions of the business model and the necessity level of various IS assets are determined" (Suh & Han, 2003). The process is initiated in on a high level where the mission and the objectives of the organization are identified and associated with business functions which are then decomposed further to sub-functions. In the lowest level of this functional decomposition lay assets (physical, informational etc.) which are vulnerable to threats. Before the risk analysis begins the business functions and assets within scope are evaluated according to their importance towards achieving the organizational goals by a variety of stakeholders, in a hierarchical prioritization process. Next relationships between assets and business functions are mapped on an asset dependency diagram and a cross-table is created in order to calculate the relative importance of each asset. The probability of various threat occurrences is identified by security experts using available techniques (e.g., Delphi) and plugged in to the formulas available by the method to calculate the annual loss expectancy (ALE) for the organization.

The advantage of ISRA-BM is that it provides a qualitative result by using a concrete underlining mathematical framework. Additionally, since it focuses on linking high level strategic goals to the risk management process, it requires the active involvement of both experts and management in the process, thus promoting security awareness in the highest organizational level. Some limitations of this approach, as noted by its creators, lay in the level of detail of the asset dependency diagram and in the precision of the formulas calculating monetary loss as there is a plethora of contributing factors.

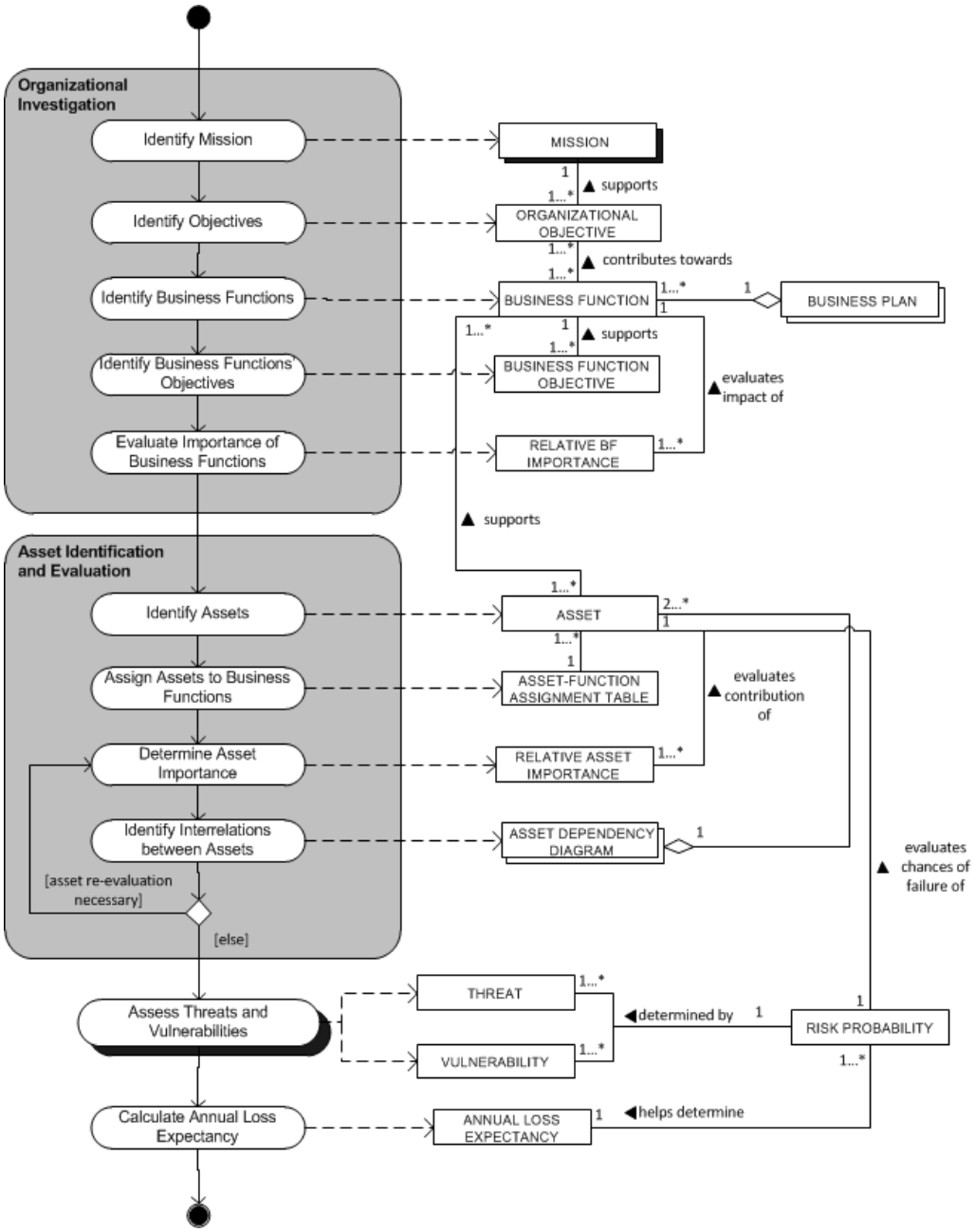


Figure 2.11: ISRA-BM method PDD

2.4 Analysis of results and baseline selection

The literature findings of this section brought forward some critical factors that determine the success of a security requirements engineering process, thus highly affecting the quality of the designed system. The most important factor is the elaboration of security from the *early* steps and *throughout the software development lifecycle*, which was a point highlighted by the majority of the literature of the area. By introducing security at the early requirements phase and elaborating on it through each phase of the development process, the resulting system will incorporate security features by design. This way, system developers can avoid costly redesigns due to security issues that were never discovered or discovered too late in the development of the software system. Another important factor is the context on which security is elaborating during the development process. It is often noted in software projects that security is considered a technical issue usually tackled at a low-level from a technical scope. Nevertheless, the consideration of the *social aspect of security* is emphasized in literature, especially when dealing with modern systems with numerous and complex interactions between human actors and software agents (Paja et al., 2012). A *high-level social overview* of the interactions between the participants of the system should provide useful input for the security elaboration of the system. The creation of such an overview can also facilitate the discussion of security at an *organizational level*, linked to high-level strategic goals which the designed system aims to accomplish. The participation of a multitude of system stakeholders, with varying backgrounds (e.g., upper management, technical personnel, etc.), in the process also contributes in that direction.

A number of security requirements engineering methods were also overviewed. As already discussed, methods based on UML and use cases, while intuitive and user-friendly, do not seem to be able to handle larger scale systems with more complex interactions between humans, organizations and software systems. They can provide convenient tools to describe some basic interactions between specific users and software agents but they are not adequate for the purposes of modern software system modelling. Goal-oriented methods, on the other hand, provide more fitting approaches as they are able to create comprehensive system models, capable of illustrating complex relationships while also providing automated support and validation throughout the process. Additionally, goals are a concept which is easier to associate with high level organizational needs and strategic decisions, thus allowing the involvement a wider spectrum of stakeholders in the process, raising security awareness throughout the organizational ranks.

	Abuse cases	Misuse cases	Anti-Models	UMLsec	Secure i*	Secure Tropos	STS-ml
Support from early requirements phase	-	-	-	+	+	+	+
Support of social analysis from an organizational point of view	-	-	-	-	+	+	+
Integration of high level strategy and organizational needs	-	-	-	-	±	±	+
Able to model complex systems	-	-	-	-	±	+	+
User-friendly, intuitive	+	+	+	+	±	±	+
Automated support tool availability	±	±	-	±	+	+	+

"+": total compliance, "±": partial compliance, "-": no compliance

Table 2.1: A comparison of SRE methods

Table 2.1 provides a quick overview of how the identified factors correspond to each of the previously discussed methods for security requirements engineering (cf. Section 2.2). As already mentioned, a common shortcoming of UML-based methods, in our case abuse, misuse case, anti-models and UMLsec, is their inability to support social aspects of analysis and integrate high level organizational goals. The nature of their analysis (mostly variants of UML use cases) leads to limitations on the complexity of the interactions it can support, thus making them unfit for modeling large systems. With the exception of UMLsec, the rest of the presented UML-based methods are used as a complimentary tool to evaluate the security of an already designed system, therefore they cannot support the development process during its initial phases. Finally, since these methods are UML-based they can be partially supported by a number of already existing CASE tools used for UML projects.

The three goal-oriented methods presented, namely Secure i*, Secure Tropos and STS-ml, were developed in order to support the early requirements phase of the software development lifecycle, as it is critical for security to be taken into account early during the system's development process. Another common attribute of these methods is their ability to capture the social aspects necessary for the development of modern complex systems. Through the usage of goal-oriented analysis, high-level organizational strategy can be reflected in the design choices made by the system's designers. Goal-models offer the ability to decompose a high level goal to a series of smaller, low-level activities thus making such methods able to handle complexity and scalability while creating more or less clear and comprehensible models. Their support from automated tools for the construction and validation of the models is another positive point, since the analysis process can be more precise and brief than manual attempts.

With all the above mentioned factors taken into account, STS-ml stands out as the most fitting method for the context of our research. The rationale behind this decision is based on the ability of the method to handle large scale systems, focusing on the security of delegations of goals and informational assets between different actors. The tool support available was also an important parameter since automated reasoning support and model validation are critical the development of such systems. Lastly, a number of published case studies and the personal experience of the authors with STS-ml testify to its performance in practical settings, making it a fitting choice for further research and possible expansions with new features.

Risk management was another topic of interest in this study, as it represents an important part of security in software systems. A number of methods for risk management were overviewed, both qualitative and quantitative in nature, most of which required stakeholder participation and expert support during the process. Techniques for risk assessment included in such methods can be a useful addition to the SRE process and a prime candidate for expanding the STS-ml method. More specifically, techniques like hierarchical prioritization of assets, risk prioritization and asset-dependency diagrams, included in ISRA-BM method, can be adapted to fit STS-ml process and add a new perspective of risk management in the process. The assembly of such a new method, based on the findings of this section, will be elaborated in Chapter 4 of this work.

Chapter 3. State of the Practice

This section focuses on the evaluation of the state of practice in the field of security requirements engineering and risk management. The evaluation was performed using the results of a survey created for practitioners and researchers of the field, distributed and completed online. In the following sub-section the method followed for the creation, distribution and analysis of the survey will be elaborated. Next, the results for each of the survey's questions will be presented, using relevant statistical analysis. Finally, our findings will be discussed and conclusions will be drawn concerning the state of the practice.

3.1 Method

The questionnaire included in the survey was created by the authors of this work. The questions included revolved around the demographics of the subjects, their practical experience with security and risk management during software development projects and their opinion on several critical factors and best practices, identified by our literature study (cf. Chapter 2). After a number of iterations a final set of questions was decided and a test run of the survey was performed. During the test run, five volunteers with a background in the fields of our study, completed the survey and provided feedback about its quality. Final adjustments were made, according to the feedback received before the final version of the survey (cf. Appendix A.1) was solidified. An extensive overview of the questions included in the survey will be provided in the following section (cf. Section 3.2).

The survey form was uploaded in an online cloud service (Google Drive) and a link for its completion was distributed to online communities. Eligible subjects for our survey included researchers, practitioners and students involved in software development projects, with no limitation on location, age, or experience. The platform hosting the survey ensured the anonymity and confidentiality of participants. However, the subjects could select to provide an e-mail address if they were interested in receiving the results of the survey. In order to attract a diverse group of participants, the survey was linked in a variety of information security and software security groups in social media (LinkedIn, Facebook) and a relevant mailing list (AISworld listserv). Participants from the personal and academic network of the authors were also recruited via direct e-mail communication. The survey became available online from late July up to the end of September of 2014.

The responses of the participants were gathered by the online hosting platform and exported for further statistical analysis. The analysis was performed using IBM's SPSS software and MS Excel for the creation of relevant graphs and charts. Descriptive statistics (e.g., median, mean, standard deviation) are calculated for each of the questions. These statistical findings will allow us to get an overview of what practitioners consider as best practices and will be a valuable input for the creation of our new method.

3.2 Results

The initial questions of the survey aimed to measure the demographics and some background information of the responders. The first question asked participants to select their age group for a number of available choices (18-24, 25-34, 35-54, 55-64 and over 65). Out of the twenty four (24) total responses collected, thirteen (13) responders were in the age group of 35-54, eight (8) were in the group of 25-34 and one at each of the rest age groups, as illustrated in Figure 3.1. In the second question the geographic location of the responder was enquired, and a list of all the continents was provided as the response. As seen in Figure 3.2, the majority of our responders (17 out of 24) were located in Europe, while five (5) were from North America and two (2) from Asia.

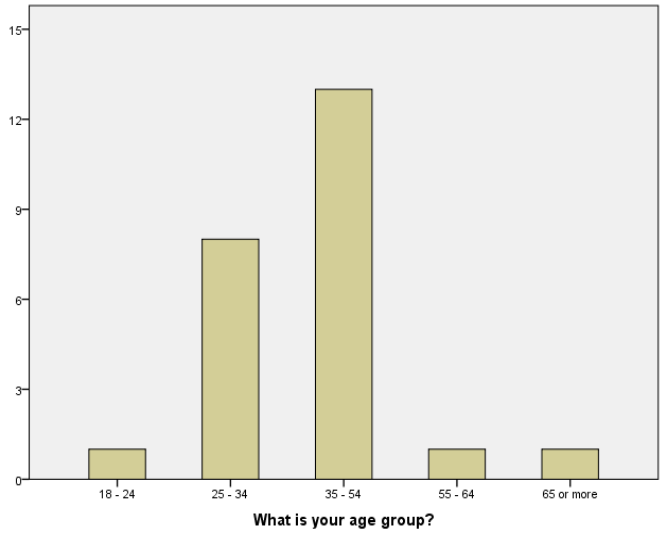


Figure 3.1: Age of survey responders

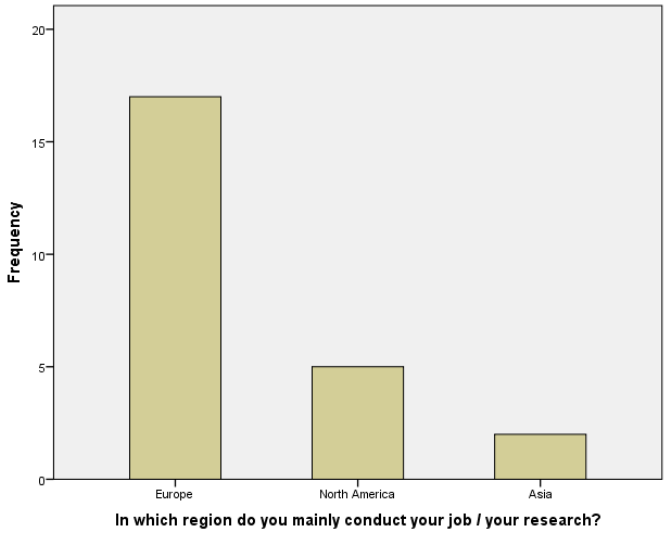


Figure 3.2: Location of survey responders

Next, participants had to indicate their background in the field of software / information systems engineering. The available responses included "Researcher",

which was selected by seventeen (17), "Practitioner" which was selected by four (4) and "Student" selected by three (3) responders (Figure 3.3). The years of experience in the field of software engineering and information security of each participant were measured by the next two questions. In both questions participants could select an available response ranging from "No experience" to "10 years of more". As illustrated in Figure 3.4, twelve (12) responders indicated experience of "10 years or more" in the field of software engineering, five (5) indicated "more than 5 but less than 10" years of experience, six (6) indicated experience of "more than 1 but less than 5" years, while one (1) indicated no experience at all in the field.

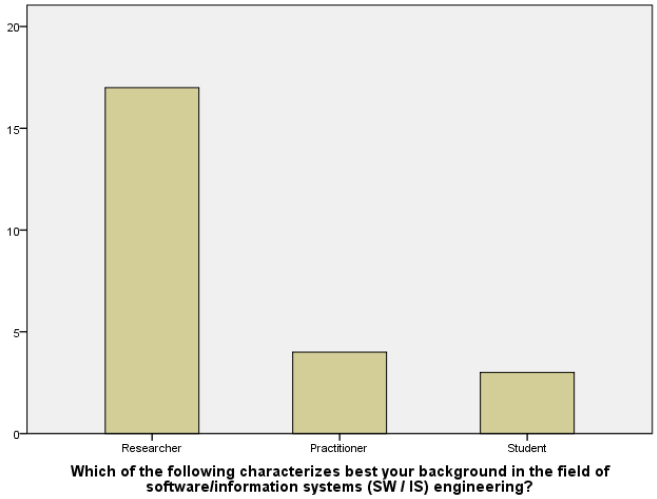


Figure 3.3: Background of responders

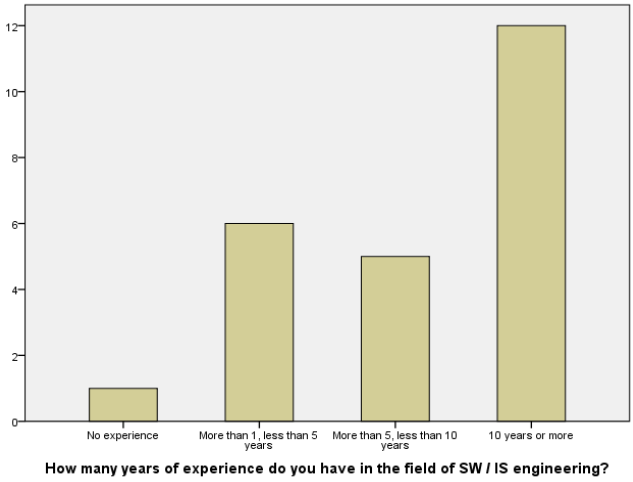


Figure 3.4: Experience in SW/IS engineering

Regarding the responder's experience in the field of information security, five (5) participants had no previous experience, one (1) had less than a year, nine (9) indicated "more than 1 but less but 5" years, four (4) selected "more than 5 but less than 10" years and five (5) had "10 or more" years of experience (Figure 3.5). Finally, the last question for the demographics section aimed at evaluating how recently the responders were involved in a software development project. From a total of twenty four (24) responders, twelve (12) indicated that they are currently involved in a project, one was involved within the last 2 years, eight (8) were involved more than two years ago while three (3) have never been involved in a software development project (Figure 3.6).

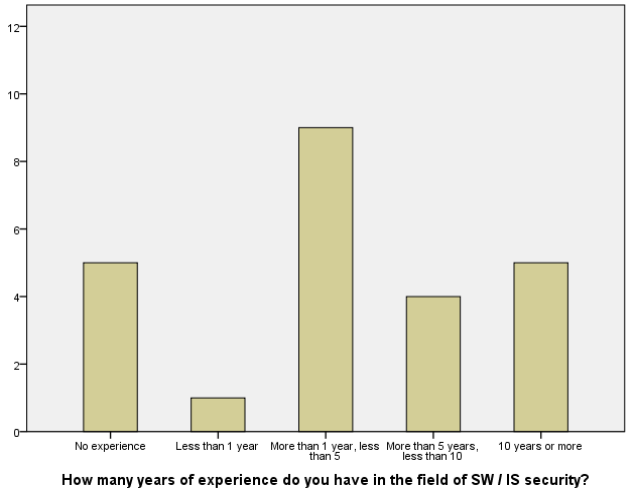


Figure 3.5: Experience in SW/IS security

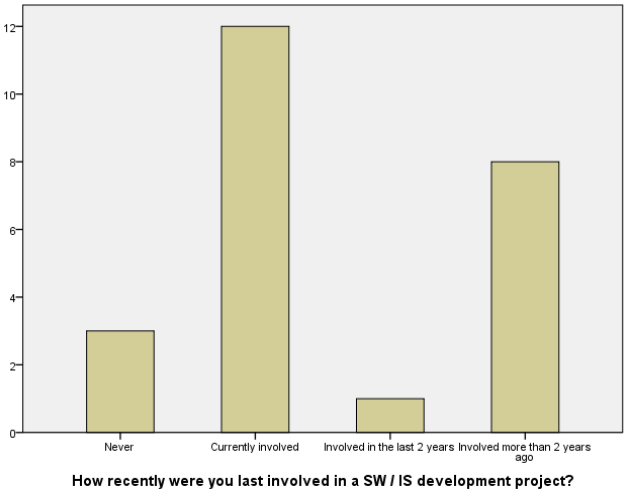


Figure 3.6: Involvement in SW/IS development projects

The next part of the survey was concerned with the security elaboration process during the development lifecycle. The first question here, asked participants how often they considered the social aspects of the system, when discussing and analyzing its security. In this question the responders could select a value from a 5-point scale ranging from "Never" to "Always". Descriptive statistics reveal that the median response is "Often", which was the 4th point in the scale (mean= 3.61, sd= 1.53), with over 60% of responses being "Often" or "Always" (Figure 3.7). In a similar tone, the next question inquired how often did other technical systems, already in place, got considered during security elaboration. Once again the responses ranged in a 5-point scale from "Never" to "Always". The median response to this question, according to the statistical analysis, is "Sometimes", which is the mid-point of the scale (mean= 3.52, sd= 1.08) selected by 37.5% of the responders (Figure 3.8).

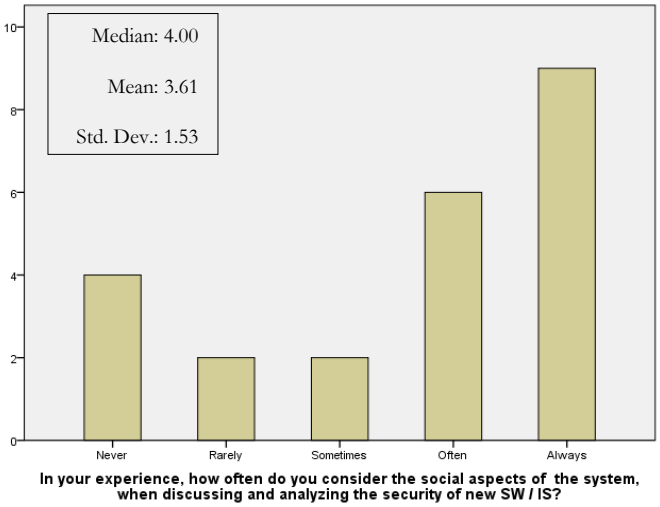


Figure 3.7: Consideration of social aspects of the system

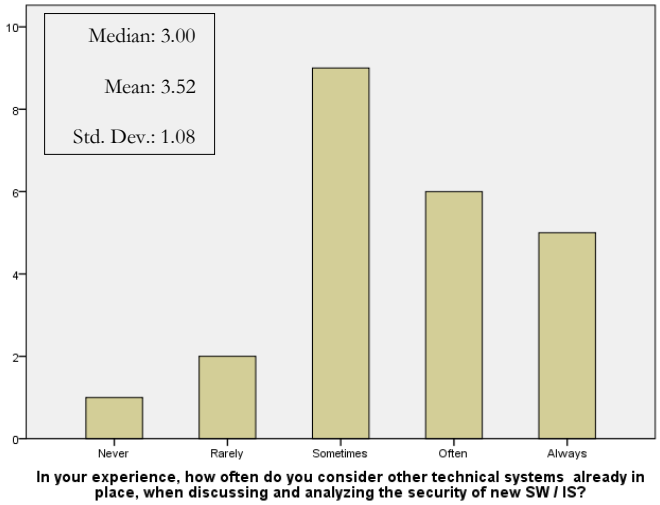


Figure 3.8: Consideration of technical environment of the system

The next question asked the participants of the survey how beneficial they considered analyzing system security from an organizational, instead of a purely technical, perspective. The available answers were on a 5-point scale ranging from "Very harmful" to "Very beneficial". The median response was "Very beneficial" which is the highest value of the scale (mean= 4.33, sd= 0.816), selected by 54.2% of responders. Similarly, the subsequent question dealt with how beneficial participants considered the adoption of security from the early stages (requirements phase) of the software development lifecycle. The same 5-point scale, used in the previous question, was available here and the statistical analysis revealed the median response as "Very beneficial" (mean= 4.71, sd= 0.624), selected by nearly 80% of responders.

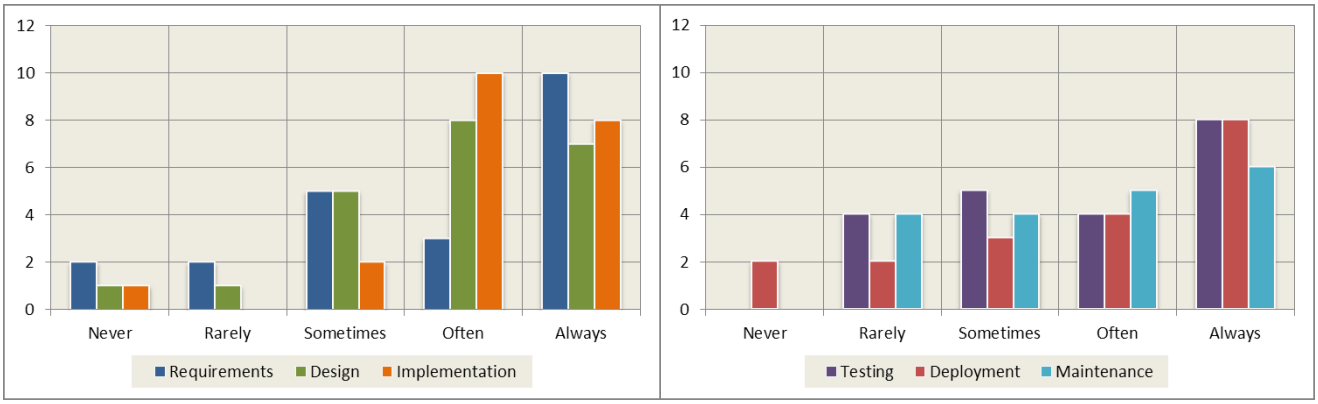
The last question of the first part of the survey aimed to determine how often is security discussed and analyzed during each phase of the software development lifecycle. The phases of the development lifecycle were listed from the earliest to the latest and a 5-point scale ranging from "Never" to "Always" was provided. The option "Don't Know" was also available for participants with no experience at certain phases of the lifecycle. The descriptive statistics were analyzed for each of the development lifecycle phases and are presented at Table 3.1.

How often do you discuss and analyze security at each phase of the development life cycle?

	Early requirements Phase	Design Phase	Implementation Phase	Testing Phase	Deployment Phase	Maintenance Phase
Mean	3,46	3,54	3,63	3,29	2,96	2,92
Median	4,00	4,00	4,00	3,50	3,50	3,00
Std. Deviation	1,693	1,503	1,663	1,681	1,989	1,840
Variance	2,868	2,259	2,766	2,824	3,955	3,384

Table 3.1: Survey results, frequency of security elaboration for each phase of the development lifecycle

The descriptive statistics analysis shows that the median frequency value for the first three phases (Early requirements, Design and Implementation phase) is "Often" (4th point on the scale) while it drops towards "Sometimes" (mid-point of the scale) for the remaining three phases of the lifecycle (Testing, Deployment and Maintenance phase). A graphical overview of the distribution of the responses is provided in Figures 3.9 and 3.10.



Figures 3.9, 3.10: Frequency of security elaboration during each phase of the software development lifecycle

The second part of the survey explores the perceived importance of different roles and factors contributing to the security elaboration of a software system. Initially the responders have to evaluate the importance of a number of roles participating in the development process. The scale used here is a 5-point scale with values ranging from "Not important at all" to "Essential", along with the additional option of "Don't know". The descriptive statistics for each of the roles is presented at Table 3.2 while a graphical representation of the responses is illustrated in Figure 3.11. A quick overview of the results reveals that the role of the security expert scores the highest in perceived importance with a median value of "Essential", while the rest of the roles are all rated with a median value of "Very important".

How important do you consider each of the following roles regarding their contribution to security elaboration?

	Software Architect	Requirement Analyst	Security Expert	Software Developer	Management / Stakeholders	End User
Mean	4,08	3,88	4,25	3,96	3,58	3,50
Median	4,00	4,00	5,00	4,00	4,00	4,00
Std. Deviation	1,283	1,227	1,422	1,122	1,472	1,414
Variance	1,645	1,505	2,022	1,259	2,167	2,000

Table 3.2: Survey results, perceived importance of roles during security elaboration

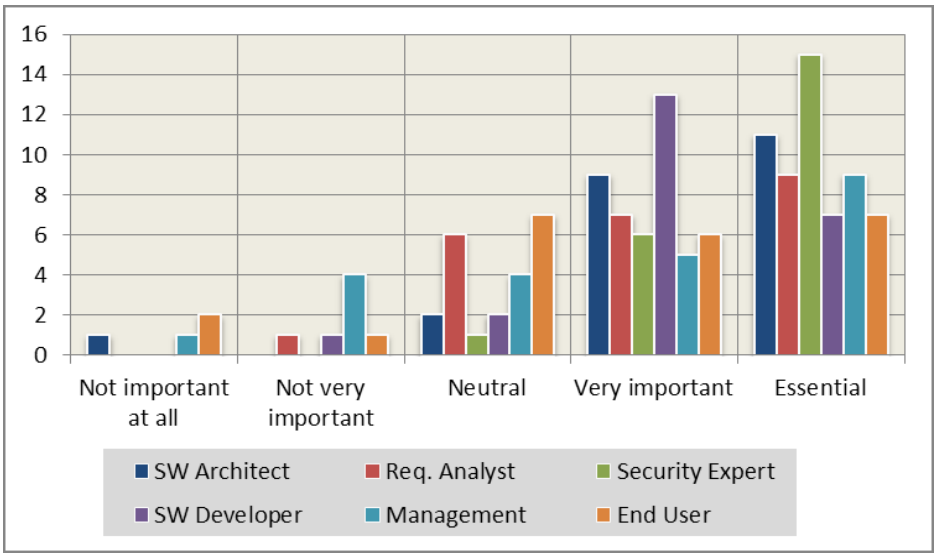


Figure 3.11: Perceived importance of roles during SW development

The importance of various factors influencing the approach followed to elaborate on security during the development lifecycle is evaluated by the responders. A 5-point scale is also used here, similar to the previous question, to represent how important each of the listed factors is perceived. Table 3.3 provides a quick overview of the descriptive statistics for each factor, where we can conclude that all factors were considered as "Very important" (median= 4) except the "reputation of the approach" which was evaluated as "Neutral" (median= 3). A visual breakdown of the evaluation results is provided in Figure 3.12.

How important do you consider each of the following factors when deciding on how to approach security for the system to-be?

	Similar past experiences	Popularity / Reputation of approach	Standardisability	Situational project needs	Stakeholder suggestions	Scientific / Academic relevance of approach	Support tool availability
Mean	3,83	3,08	3,50	3,54	3,33	3,83	3,79
Median	4,00	3,00	4,00	4,00	4,00	4,00	4,00
Std. Deviation	1,204	1,139	1,414	1,382	1,404	1,373	1,141
Variance	1,449	1,297	2,000	1,911	1,971	1,884	1,303

Table 3.3: Survey results, perceived importance of factors influencing the security elaboration process

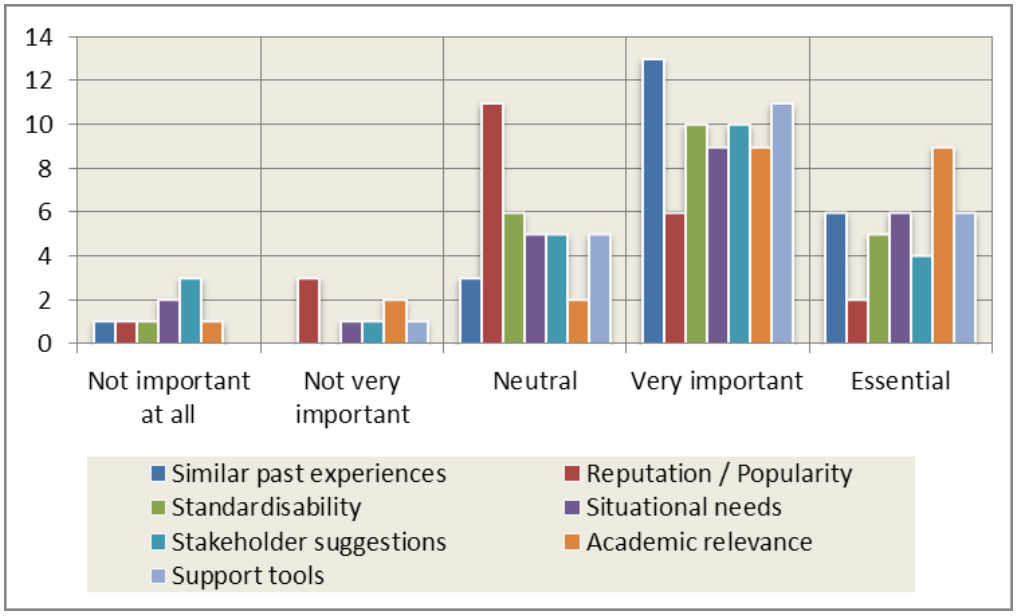


Figure 3.12: Perceived importance of influencing factors

The third part of the survey contained questions regarding the adoption of systematic approaches and automated tools during the security elaboration process. In more detail, the responders had to answer how often they use systematic approaches and automated support tools when elaborating on a software system's security, and how beneficial they consider these practices. Once again, all answers were provided in a 5-point scale ranging from "Never" to "Always" when measuring frequency and from "Very harmful" to "Very beneficial" when measuring usefulness. The results of the statistical analysis on the gathered data indicated that the median response for the frequency of usage of systematic approaches was "Sometimes" (mean= 3.27, sd= 1.386) which corresponds to the middle of the 5-point scale (Figure 3.13). Similarly, the frequency of usage of automated support tools had a median response of "Rarely" (mean= 2.41, sd= 1.5) (Figure 3.14). Regarding the perceived benefit of systematic approaches for security elaboration, the median response was found to be "Very beneficial" (mean= 4.36, sd= 0.848). In a similar question about the perceived benefit of automated support tool usage, the median response was "Beneficial" (mean= 3.80, sd= 1.105). This indicates that the responders had a very positive opinion about both factors despite their low frequency of use.

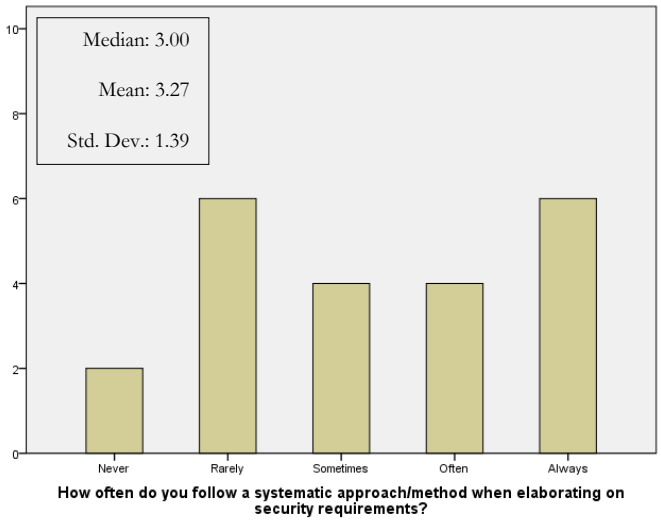


Figure 3.13: Frequency of use of systematic approaches

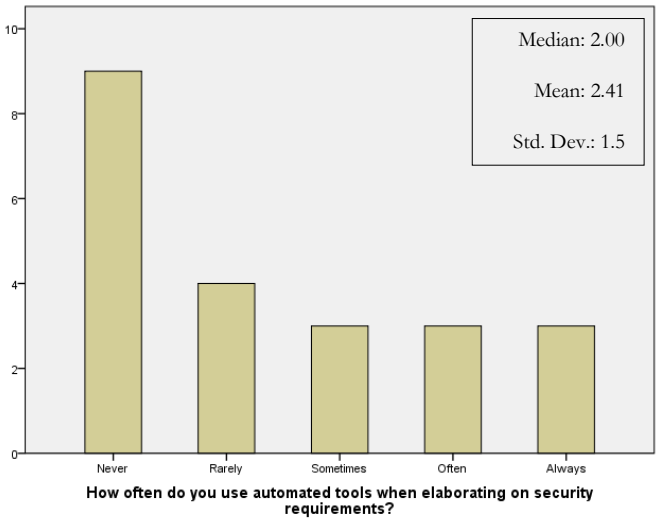


Figure 3.14: Frequency of use of automated tools

The final part of the survey was centered on risk management during the development lifecycle. The first question in this part, aimed to measure how often is risk discussed and analyzed during each phase of the software development lifecycle. The phases of the development lifecycle were listed from the earliest to the latest and the answers were given in a 5-point scale ranging from "Never" to "Always". The option "Don't Know" was also available for participants with no experience at certain phases of the lifecycle. The descriptive statistics were analyzed for each of the development lifecycle phases and are presented at Table 3.4.

How often do you analyze risk at each phase of the development lifecycle?

	Early requirements phase	Design Phase	Impementation phase	Testing phase	Deployment phase	Maintenance phase
Mean	3,25	3,21	2,83	2,50	2,54	2,46
Median	3,00	4,00	3,00	2,50	2,50	2,00
Std. Deviation	1,675	1,532	1,551	1,588	1,744	1,769
Variance	2,804	2,346	2,406	2,522	3,042	3,129

Table 3.4: Survey results, frequency of risk analysis for each phase of the development lifecycle

From the statistical analysis results above it can be concluded that risk management takes place more often during the design phase of the development lifecycle, with a median value of "Often". During the early requirements and implementation phase risk management is "Sometimes" performed, according to the median value given by responders. In the last three phase of the lifecycle (testing, deployment and maintenance) risk management is a less common occurrence as indicated by the lower median values selected by the survey responders. A better

illustration of the responses given for the occurrence of risk management for each phase of the development lifecycle is provided in Figure 3.15.

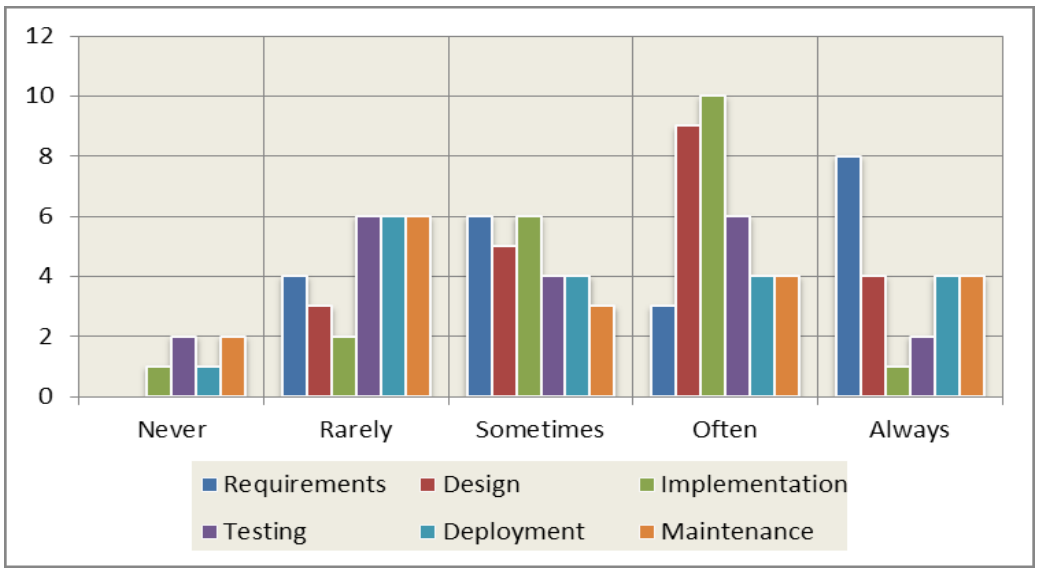


Figure 3.15: Risk management during the development lifecycle

The next question regarded the importance of different types of assets during the risk management process. Choosing from a 5-point scale ranging from "Not important at all" to "Essential" the participants had to evaluate four different asset groups (Information, Applications, Infrastructure and Organizational assets) according to their importance for the overall system's security. A "Don't know" option was also available. The descriptive statistics for the responses of this question are provided in Table 3.5.

How important for the overall security of the system do you consider each of the following asset groups?

	Information / Data	Applications / Software	Infrastructure / Physical assets	Personnel / Organizational assets
Mean	4,29	4,08	4,08	4,04
Median	5,00	4,00	4,00	4,00
Std. Deviation	1,160	1,100	1,176	1,197
Variance	1,346	1,210	1,384	1,433

Table 3.5: Survey results, perceived importance of different asset types during risk management

The statistical analysis reveals that all asset groups are considered at least "Very important" with Information assets achieving a median value of "Essential". A graphical illustration of the asset groups' evaluation is provided by Figure 3.16.

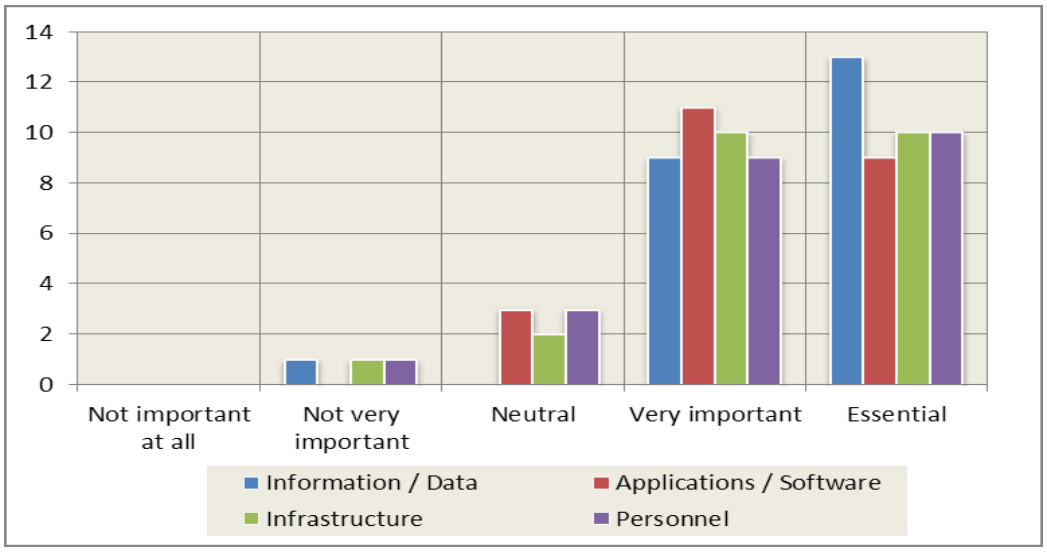


Figure 3.16: Perceived importance of asset groups

Next, the frequency of integration of risk management with the security elaboration process is inquired. A 5-point scale ranging from "Never" to "Always" was provided to the participants. The statistical analysis of the responders' input revealed a median response of "Often" (mean= 3.18, sd= 1.435), meaning that elements of risk analysis are often integrated in the security requirements elaboration process in practice (Figure 3.17). Similarly, the next question aimed to identify how often systematic approaches are followed during risk analysis, using the same scale of answers as the previous question. Here the median response, according to the statistical analysis performed, is "Sometimes" (mean= 2.86, sd= 1.356) which represents the mid-point of the 5-point scale used to measure frequency (Figure 3.18). Finally, the last question of the survey inquired about the usage of qualitative and quantitative methods to perform risk analysis. In order to identify whether qualitative or quantitative methods are preferred in practical settings a 5-point scale was used, with values ranging from "Only qualitative" to "Only quantitative". The analysis of the results revealed a "Neutral" median response (mean= 2.60, sd= 0.995), representing the mid-value of the scale, implying no significant preference for either type of risk analysis method.

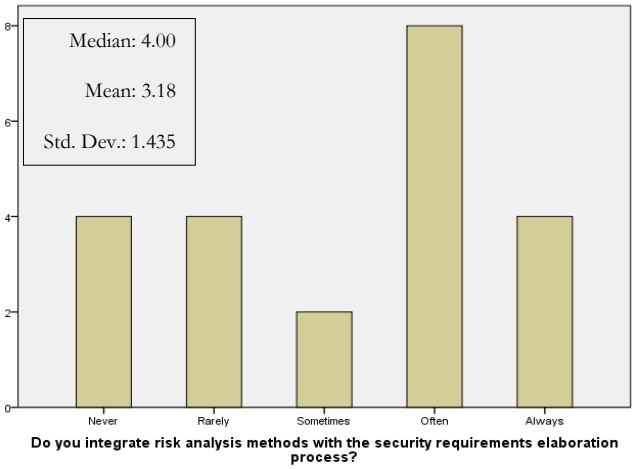


Figure 3.17: Risk analysis integration with security elaboration

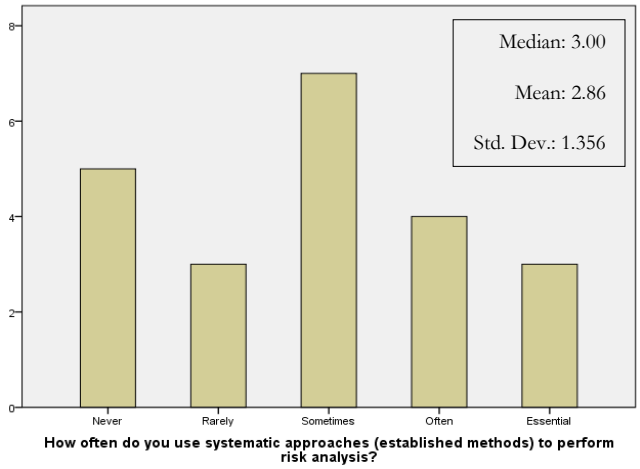


Figure 3.18: Frequency of systematic approach usage during risk analysis

3.3 Findings

From the results of the statistical analysis of the survey responses, conclusions can be drawn regarding the state of practice. The majority of responders indicated that, according to their previous experience with such projects, the social aspects of the system are usually considered during security elaboration (see Figure 3.7). Similarly, the technical infrastructure already in place was also a factor to consider, albeit less frequently than the social aspects (see Figure 3.8). It would be interesting, for future research, to investigate which type of projects take such factors into account and what is the rationale behind that decision.

Responders also indicated a very positive opinion about the benefits of including an organizational view during the security consideration, instead of approaching it from solely a technical point of view. These findings agree with the literature suggestions about the need of considering *social aspects of the system* under design, from a *high-level organizational point of view*, when dealing with security. Social high-level modeling will therefore be an important part of the analysis provided by our new method.

Regarding the place of security elaboration in the software development lifecycle, a very positive opinion was expressed regarding the benefits of the adoption of security in the early stages of the lifecycle, during the requirements phase. The actual adoption of security, according to the responders' past experiences, usually takes place during the initial three phases of the development lifecycle, namely

requirements, design and implementation phase(see Figures 3.9 and 3.10). These results indicate that the benefits of *early security elaboration* are known but, in practice, security may be dealt with at a later stage of the development, after the requirements phase. Our method will thus aim to encourage security elaboration from the initial phase of the development lifecycle.

Next different factors and roles contributing to the security elaboration process were evaluated by the responders. According to the results, the input of all different roles (analysts, architects, developers, end users, etc.) is considered important, with the role of the security expert standing out as essential for the security elaboration process (see Figure 3.11). Factors that influence the selection of the method used for security elaboration vary from situational project needs and past experiences of the analysts to the standardisability and tool support of the method (see Figure 3.12). Similarly, systematic approaches and automated analysis tool support were perceived as very beneficial for the security elaboration process, but their use in practical settings was indicated as limited.

The above results suggest that the *input of a multitude of roles* and stakeholders is an important part of the process, as also suggested by the scientific literature of the area. Additionally while *systematic approaches* and *support tools* are recognized as beneficial, their adoption in practice has yet to become widespread. All the above factors will be taken into account during the development of a new systematic method, involving a variety of stakeholders and experts to the security elaboration process.

Finally, regarding the risk management process, the responders indicated that, according to their experience, it more often takes place during the design phase of the development lifecycle (see Figure 3.15). It was also reported that risk analysis is often integrated with the security elaboration process. Systematic approaches for risk management were only occasionally used, while no preference was reported towards quantitative or qualitative methods. Informational assets were regarded as the most important asset group during the risk management process, with the rest of the asset groups (software, physical and organizational assets) still having a significant value (see Figure 3.16). This *integration of risk management elements in the security elaboration process* will play a key role in our newly developed method, which will focus on the *informational assets* of the system and provide risk evaluation and prioritization from the early stages of the development.

Our approach to identify the state of practice presented some limitations which should be addressed. The sample size of our survey included twenty-four responders, which may hinder the generalizability of the survey's findings. This can be attributed to a multitude of factors, the most important of which was the limited timeframe available for the distribution and analysis of this survey. A greater number of responders would also be possible if the survey had been communicated

through other mediums, since in our case it personal invitations via e-mail that attracted the majority of our responders. Another point worth elaborating is the composition of the responder's group. As previously presented (cf. Section 3.2), most of the responders had a research background while practitioners were limited to a small percentage of the overall subjects (4 out of 24). This fact may have affected our findings, skewing them towards opinions more popular in the field of research than in practice. The same factor also made comparative between-groups analysis impossible, as the resulting groups would be highly unequal in size.

Nevertheless, as indicated by their demographics, most of the responders had multiple years of experience in the field of software engineering, with involvement in software development projects, regardless of their background as a practitioner or researcher. Therefore, their level of expertise and their past experiences make their contributions to this survey valuable.

Chapter 4. Integrating elements of risk management in SRE

In this section we will introduce a new method that integrates elements of risk management in the security requirements engineering process. First the method assembly process will be presented where an overview of the building blocks of the method will be provided and the rationale behind their selection will be elaborated. Next the newly created approach will be presented by focusing on the activities, deliverables and stakeholders required, followed by a short discussion on its limitations.

4.1 Method components and integration

As already discussed in the Literature Review section of this work, STS-ml will be the main building block upon which the new method will be developed. STS-ml uses goal models to map the system and the interrelations between its actors. Actors in a socio-technical system can be humans, organizations and software agents, all of which with specific goals to accomplish. In order to do so, delegation of goals and documents takes place and different security needs are mapped onto these exchanges. Each stakeholder can have different security needs which are taken into account, checked for consistency and finally translated into security requirements by automated tools supporting the process.

The only elements of risk management present in this method are "events" which represent threats towards documents, goals and actors of the system. These events are mapped onto the model according to the analyst's discretion but no further elaboration on their impact or likelihood to occur takes place throughout the process. Our review of risk management methods (cf. Section 2.3) revealed some useful methods, fragments of which can be integrated in the STS-ml process in order to lead to a more mature handling of risk elements. More specifically the prioritization process followed by the IS Risk Analysis based on a Business Model (ISRA-BM) can be easily adapted to fit the STS-ml process. During this process stakeholders prioritize processes and assets which are then interrelated and ranked according to their relevant importance to the system. Threats on these assets are then identified by the analysts and the overall risk they impose to the system is calculated by using mathematical formulas.

An elaboration of the steps of each of these two methods will follow accompanied by process deliverable diagrams (PDDs) of their main activities and deliverables. Their integration into a new approach will be the main focus of the remainder of this section.

4.1.1 STS method overview

The process that the system designer has to follow in order to perform system modeling and analysis using the STS method revolves around the creation of three complementary system views. The iterative process followed for the construction of the model can be supported by the STS-Tool. The first main phase is the creation of the social view of the model. The initial step here is to *identify the stakeholders* of the system which are modelled as actors who can either be agents, when referring to "concrete participants", or roles, when referring to abstract actors.

The next step requires the *identification of the assets and interactions* of the stakeholders included in the model. The assets of an actor consist of the goal he needs to achieve as part of the system and the documents necessary to do so. A goal can be refined by "AND" and "OR" decompositions to a goal-tree which contains sub-goals that divide the main goal into smaller, independent tasks. The interactions between actors illustrate the delegation of such tasks (sub-goals) to other actors of the system when they cannot be accomplished by the actor alone. The interactions mapped on the model also include the exchange of documents necessary for the completion of goals which need to be provided by other actors. The next step focuses on the modeling of the *security needs* of the identified stakeholders. In this step stakeholders express their expectations regarding the security of the interactions in which they participate. The security needs supported by the STS method are: non-repudiation, redundancy, non-delegation, trustworthiness, availability, integrity and confidentiality. The final step of the first phase is the *threat modeling*. Here a security analyst identifies events that can either threaten the completion of a (sub-) goal or affect the availability of a document so it can lead to a disruption to the system's process.

During the second phase of the STS method the information view of the system's model is created. In this phase ownership is assigned at pieces of information which are then matched to the documents containing them. The first step is concerned with *modeling the ownership of information* which assigns pieces of information to the stakeholders that own them. In the next step the *information structure* is illustrated, where information is mapped to one or more documents which include it. Through this mapping the interconnection between documents can be visualized

while also the informational content of each document is exhaustively listed. The next phase is concerned with the creation of the *authorization view* of the model. The stakeholders, during this step authorize the usage, modification, production and distribution of their information from other actors of the system. All the interactions involving information exchanges via documents are reviewed and authorized by the information owner with the assistance of the security analyst.

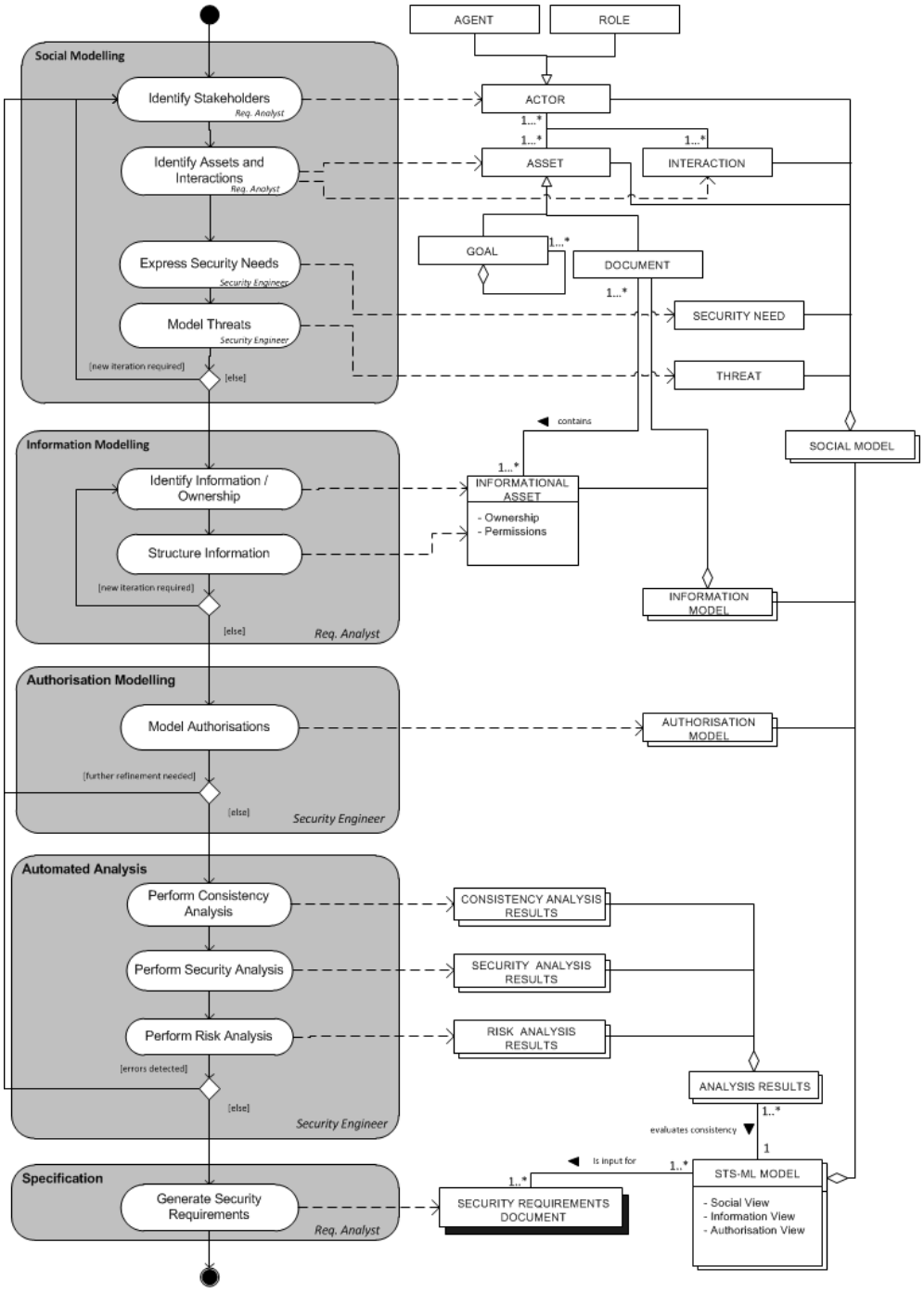


Figure 4.1: STS method PDD

The *automated analysis* of the created model is performed during the last phase of the method. With the support of the STS-tool three different analyses are performed and feedback is provided when errors or inconsistencies are found. The first type of analysis is the *consistency analysis* which verifies the validity of the created model in terms of completeness. Next the *security analysis* takes place where the model is checked for conflicting security requirements and security needs not covered by the current version of the model. The last analysis performed is the *risk analysis* where the threat posed by certain events is propagated through the model. Using the feedback of all three analyses the designer of the system should make the appropriate adjustments to the design and run the automated analysis process again, with the improved version of the model as input. When the final version of the design is solidified the security requirements can be automatically derived and listed by the STS-tool. Finally a report can be created for the stakeholders that will be used further in the development process of the system.

4.1.2 ISRA-BM method overview

The IS Risk Analysis based on a Business Model (ISRA-BM) method aims to quantitatively calculate the risk imposed on an organization in terms of annual monetary loss expected (ALE). In order to so, a sequence of actions must take place with an analyst overviewing the whole process and various stakeholders providing their input. The first stage of the process focuses on the *organizational investigation* in order to understand and map the organizational environment that will be later analyzed. The *mission* of the organization is first identified followed by the *organizational objectives* derived by it. Then the *business model* of the organization is created where the different business functions necessary for the achievement of the organizational objectives are represented. By applying a functional decomposition the business functions are broken down to sub-functions, which can then be further decomposed to sub-sub-functions until the lowest level of activity is represented and linked to the IS assets supporting it. Next the stakeholders in charge of each business function *identify its objectives* and takes part in *determining its relative importance*. The relative importance is an arithmetic value between 0 and 1 and it represents the degree of contribution of each business function towards the organizational objectives. Due to the functional decomposition applied to each business function a hierarchal tree model of (sub-) functions has been created. By applying a "successive pairwise comparison" technique all the sub-functions of a business function are compared in pairs, in order to assess their effect on the objectives of the business function they are part of. This technique, part of on the Analytical Hierarchy Process (AHP), when applied for all functional decompositions

present in the model, results in a relative importance value (FI_i)for each (sub-) function which represents its degree of contribution towards the objectives of the function it supports. Different techniques can be used for the assessment of the relative importance values from stakeholders. The analyst may choose to average the individual values given by each stakeholder for each sub-function, let all involved stakeholders decide together on a value, or combine both approaches.

The second stage of the process is concerned with the *identification and evaluation of the organizational assets*. In the ISRA-BM approach the assets can be categorized in the following seven groups: hardware, software, data/database, personnel, documentation, and various facilities. Once all assets are identified and categorized the analyst assigns them to the business function which they support, with the assistance of stakeholders if necessary. The next step is the *evaluation of the relevant necessity of each asset* (N_{ij}), which is a numerical value between 0 and 1 that represent the degree of the asset's contribution towards the goals of the business function it supports. The asset evaluation process is similar to the process described earlier for the assessment of the relative importance of business (sub-) functions. Once the relative necessity of every asset is calculated a new calculation takes place to identify its *relative importance* towards the achievement of the overall organizational goals (AI_i). To calculate the relative importance, the necessity value of the asset (N_{ij}) is multiplied by the relative importance of each function it supports (FI_i). The resulting value (AI_i) represents the degree of the overall contribution of the asset towards the organizational goals, but it can be influenced by interdependencies between the asset and the rest of the organizational assets. In order to identify such dependencies between assets the analyst creates an *asset dependency diagram* which can be consulted when the relative importance of each asset is identified. According to the rules of ISRA-BM if an asset is dependent on other assets, the relative importance of the asset is equal to the highest relative importance of the assets it is depending on.

The *assessment of the threats and vulnerabilities* of the organization is dealt with in the next stage of the process. Threats and vulnerabilities that pose risks to the assets of the system are identified by the analyst. The probability of threat occurrence, which is integral for the calculation of the overall risk later on, is identified by the use of external resources and techniques, chosen by the analyst according to his experience and the project's requirements. During the final stage of the process *the annual loss expectancy* (ALE) is calculated by using mathematical formulas provided by the ISRA-BM method. Initially the income loss for each asset is calculated by taking into account different factors such as the assets relative importance (AI_i) and the estimated recovery time. The ALE calculation takes into account the income loss, the replacement costs and the probability of the threat occurrence of each asset and provides a monetary value which represents the loss that the organization will have to endure.

4.1.3 Method assembly process

The framework of Brinkkemper (1996), illustrated in Figure 4.2, will be followed as a guide for the assembly process. According to this framework method fragments should be selected from the method base, in order to be assembled into a new method. In our case, the method base consists of the methods already identified and overviewed during the literature study. The main methods from which fragments are selected have been analyzed in depth in the previous section, so all the prerequisites for the assembly phase are covered.

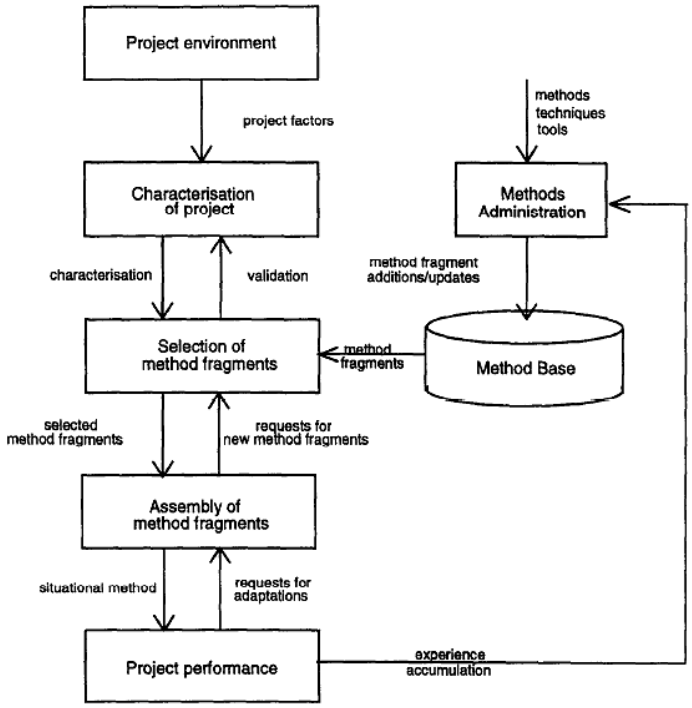


Figure 4.2: The method assembly process (Source: Brinkkemper, 1996)

For the first phase of the new method the activity of evaluating the importance of the systems goals, present in the initial phase of the ISRA-BM method, is added to the information view modeling phase of the STS method. In more detail, after the identification of stakeholders, goals and assets, the evaluation of all the system's goals takes place from the involved stakeholders. This way each goal in the social view of the model also has a relative importance value. Additionally, the identification of threats and security needs, present during the initial phase of the STS method, will be performed later on in our process. More specifically, the identification of threats will be grouped with other activities related to risk management. The output of the risk management process can then be taken into consideration and guide the users when expressing their security needs concerning the delegations and information exchanges taking place in the system.

The next phase is concerned with the creation of the information model of the system. Here the process followed by the STS method is enriched with a new activity, inherited by the ISRA-BM method, during which the importance of the informational assets is determined by the system's stakeholders (e.g., using a scale from "*Not important*" to "*Essential*"). As a result the identified information, besides being structured and interrelated with other information and documents, also has a necessity value, which will assist in assessing potential risks during the next phase of the process.

During the threat assessment phase we introduce the activity of threat identification, which was implied in the "Model threats" activity of the STS method. This explicit division of the two activities is due to the fact that in our approach the identification of threats is an important activity which can be performed following a number of external methods and techniques by a number of security specialists. The threat modeling activity that follows, inherited by the STS method, is now repurposed to simply modeling the identified threats as events in the social view of our model. After that, borrowing the activity from ISRA-BM, the probability of a threat occurrence is determined by the security specialists, once again using external resources. At the same time, the threat's impact value is calculated by a newly introduced activity. During the threat impact assessment the assets and goals affected by the threat are traced through the system's social model and their importance values are multiplied to calculate the threat's impact value. Next a risk value is calculated for each threat and all the identified threats are ranked according to their value.

The user requirements modeling phase follows, where, as in the STS method, the authorizations given by each system actor are modelled in the authorisational view of the system's model. Parallel to the authorization modelling another activity takes place during which the stakeholders express their security needs which will pose restrains in the delegations modelled in the social view of the system. This activity is also adopted by the STS method where it takes place during the social modeling phase. In our method it is moved in the later stages of the process, after the risk management has been completed and the different threats to the system have been prioritized. This allows the analysts and the system users to have a move clear overview of potential security issues when expressing their security needs. This way the selection of constrains imposed on the system by each user, via his security needs, is based on an informed decision and can be rationalized and traced back to specific threats. Finally the last two phases, namely automated analysis and specification, are borrowed from the STS method without any major modification. The automated risk analysis activity, included in the STS method is omitted here, as the threat impact tracing it performed is already covered during the threat impact assessment activity, earlier in our method.

In order to provide a quick overview of the newly created method, a process deliverable diagram (PDD) has been created (Figure 4.3) accompanied by the activities and concepts tables (Tables 4.1 and 4.2) which provide descriptions of its building blocks. An overview of the origin and modifications of each of the methods fragments is provided in Appendix A.2.

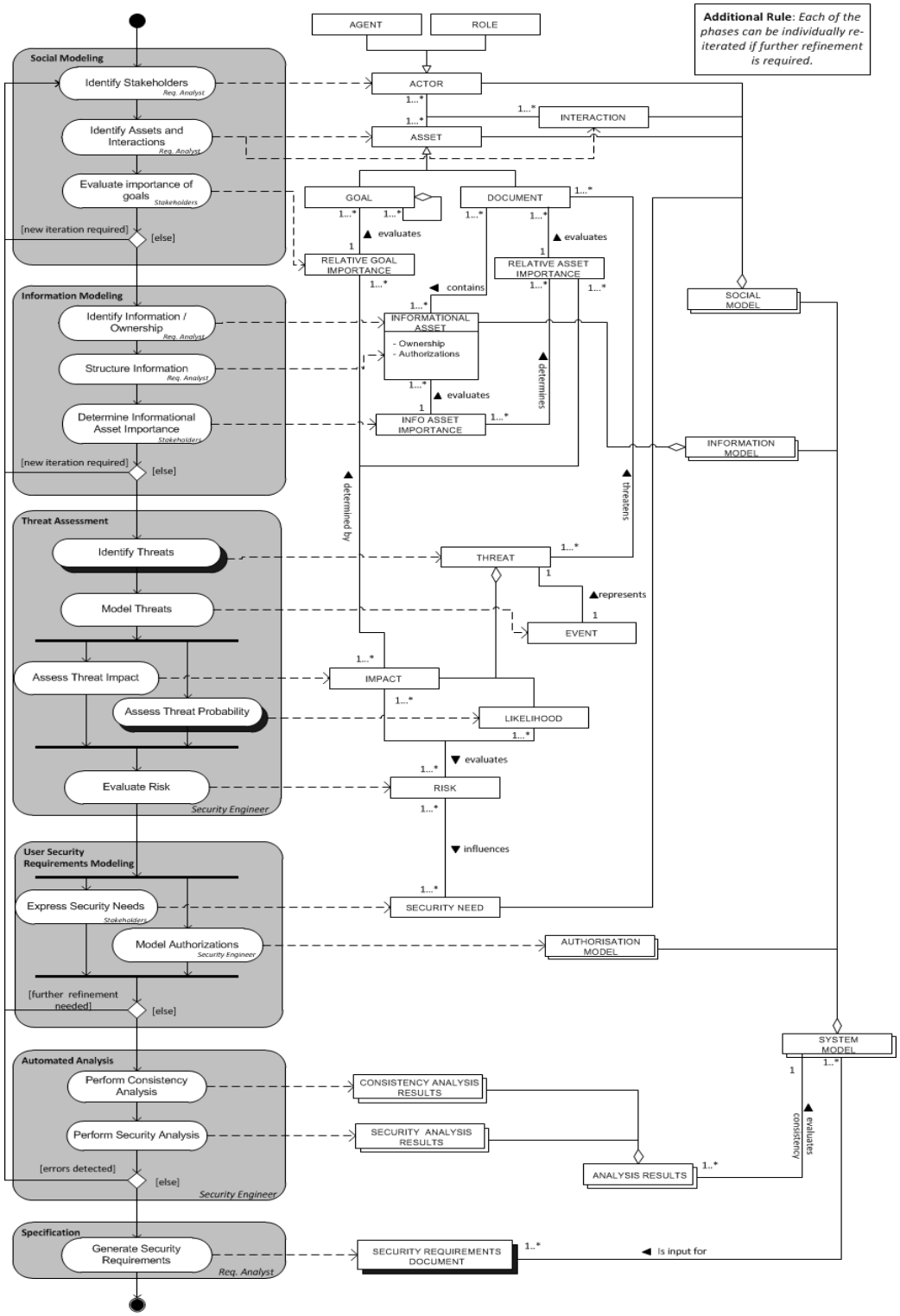


Figure 4.3: PDD of assembled method

Activity	Sub-Activity	Description	Role
Social modeling	Identify Stakeholders	The ACTORs participating in the system are identified.	Req. Analyst
	Identify Assets and Interactions	The ASSETs of ACTORs (GOALs and DOCUMENTs are identified and the interactions between them are modeled.	Req. Analyst
	Evaluate importance of goals	Values are provided for the RELATIVE GOAL IMPORTANCE of each of the system's GOALs	Stakeholders
Information Modeling	Identify Information /Ownership	The INFORMATIONAL ASSETs contained in the system's DOCUMENTs are identified and have their ownership assigned to ACTORs.	Req. Analyst
	Structure Information	The relationships between INFORMATIONAL ASSETs and DOCUMENTs are modeled at the INFORMATION MODEL.	Req. Analyst
	Determine Informational Asset Importance	Values are provided for the INFO ASSET IMPORTANCE of each of the system's INFORMATIONAL ASSETs.	Stakeholders
Threat Assessment	Identify Threats	THREATs to the system's security are identified.	Security Engineer
	Model Threats	Identified THREATs are modeled by EVENTs at the SOCIAL MODEL.	Security Engineer
	Assess Threat Impact	The IMPACT of each THREAT is identified by the RELATIVE GOAL IMPORTANCE and RELATIVE ASSET IMPORTANCE values of the ASSETs it affects.	Security Engineer
	Assess Threat Probability	The LIKELIHOOD of each THREAT is identified using external resources.	Security Engineer
	Evaluate Risk	The RISK that each THREAT poses to the system is evaluated using its IMPACT and LIKELIHOOD values.	Security Engineer
User Security Requirements Modeling	Express Security Needs	Each ACTOR expresses SECURITY NEEDs for each delegation of his GOALs and each transfer of his DOCUMENTs	Stakeholders
	Model Authorizations	Authorizations for the INFORMATION ASSETs of each ACTOR are modeled on the AUTHORIZATION MODEL.	Security Engineer
Automated Analysis	Perform Consistency Analysis	The consistency of the SYSTEM MODEL is automatically assessed.	Req. Analyst
	Perform Security Analysis	The satisfaction of SECURITY NEEDs imposed on the system is automatically assessed.	Req. Analyst
Specification	Generate Security Requirements	A SECURITY REQUIREMENTS DOCUMENT is automatically generated.	Req. Analyst

Table 4.1: Activities table of the new method

Concept	Description
ACTOR	Represents each autonomous participant of the system (Paja et al., 2012). Two types of actors are supported: agents, to refer to concrete participants, and roles, to refer to abstract actors.
ASSET	Anything that has value to the organization and is necessary for achieving its objectives (Dubois, Heymans, Mayer & Matulevičius, 2010). An asset can represent a goal or a document in the system's model.
INTERACTION	Represents the establishment and evolution of social relationships on the basis of the messages exchanged between actors. (Dalpiaz, Giorgini, & Mylopoulos, 2013). Interactions in the context of our method are limited to delegations or transmissions.
GOAL	A goal represents the strategic interests of an actor (Giorgini, Massacci, Mylopoulos & Zannone, 2005). It can be decomposed in sub-goals which represent individual processes that can be followed for the achievement of the goal.
DOCUMENT	A tangible resource that represents the concrete entities that actors exchange in order to achieve their goals. (Dalpiaz, Paja & Giorgini, 2011)
RELATIVE GOAL IMPORTANCE	The relative goal importance value refers to the degree to which each goal contributes to the organization's objectives. The relative importance may range from 0 to 1. (Suh & Han, 2003)
RELATIVE ASSET IMPORTANCE	The relative asset importance value refers to the degree to which the asset contributes to the objectives of the system (Suh & Han, 2003). The value may range from 0 to 1 and is derived from the importance values of the related information assets.
SOCIAL MODEL	Represents actors as social entities with goals they want to achieve and documents they may use or distribute to other actors while achieving these goals. (Paja et al., 2012)
INFORMATIONAL ASSET	Represents individual pieces of information which can be part of some document. (Paja, Dalpiaz & Giorgini, 2012)
INFO ASSET IMPORTANCE	The information asset importance refers to the criticality of each piece of information towards the achievement of the overall system's goals. The value may range from 0 to 1.
INFORMATION MODEL	Represents the information in the considered organization/setting together with the documents that represent such information, as well as the relationships among these informational entities or documents respectively. (Paja et al., 2012)
THREAT	A potential attack, carried out by an agent that targets one or more IS assets and that may lead to harm to assets. (Dubois, Heymans, Mayer & Matulevičius, 2010)
EVENT	Represents threats that negatively influence the capability of an actor to achieve one or more goals.
IMPACT	The potential negative consequence of a risk that may harm assets of a system or an organization, when a threat (or an event) is accomplished. (Dubois, Heymans, Mayer & Matulevičius, 2010)
LIKELIHOOD	The chance that a loss or harm will occur over the lifetime of an asset or within a specified period of time. (Suh & Han, 2003)
RISK	Risk is defined as the potential that a given threat will exploit vulnerabilities of an asset or group of assets and thereby cause harm to the organization. (ISO /IEC, 2004).
SECURITY NEED	Security needs are imposed by actors over interactions to constrain the way they take place within the system. (Paja et al., 2012)
AUTHORIZATION MODEL	Represents the permission flow from actor to actor, that is, the authorizations actors grant to others about information, specifying the operations actors can perform on the given information. (Paja et al., 2012)
SYSTEM MODEL	An aggregation of the social, information and authorization model. These different

	views provide a comprehensive picture of the setting which includes both business concerns and security aspects. (Dalpiaz, Paja & Giorgini, 2011)
ANALYSIS RESULTS	The outcome of the consistency and security automated analysis of the systems model. Used to identify inconsistencies and violations of the user's security needs at the system's model.
SECURITY REQUIREMENTS DOCUMENT	Contains the specifications of security requirements for the system-to-be, derived once the modelling is done and the security needs imposed by the actors are expressed. (Paja et al., 2012)

Table 4.2: Concepts table of the new method

4.2 Method description

The initial step of the method requires the *creation of a system model* including roles, goals and assets. The system model is created by the analyst, with the contribution of stakeholders, following the STS method. In order to illustrate our approach, a mock model was created in the STS-Tool and will be used throughout this section as an example (Figures 4.4, 4.5).

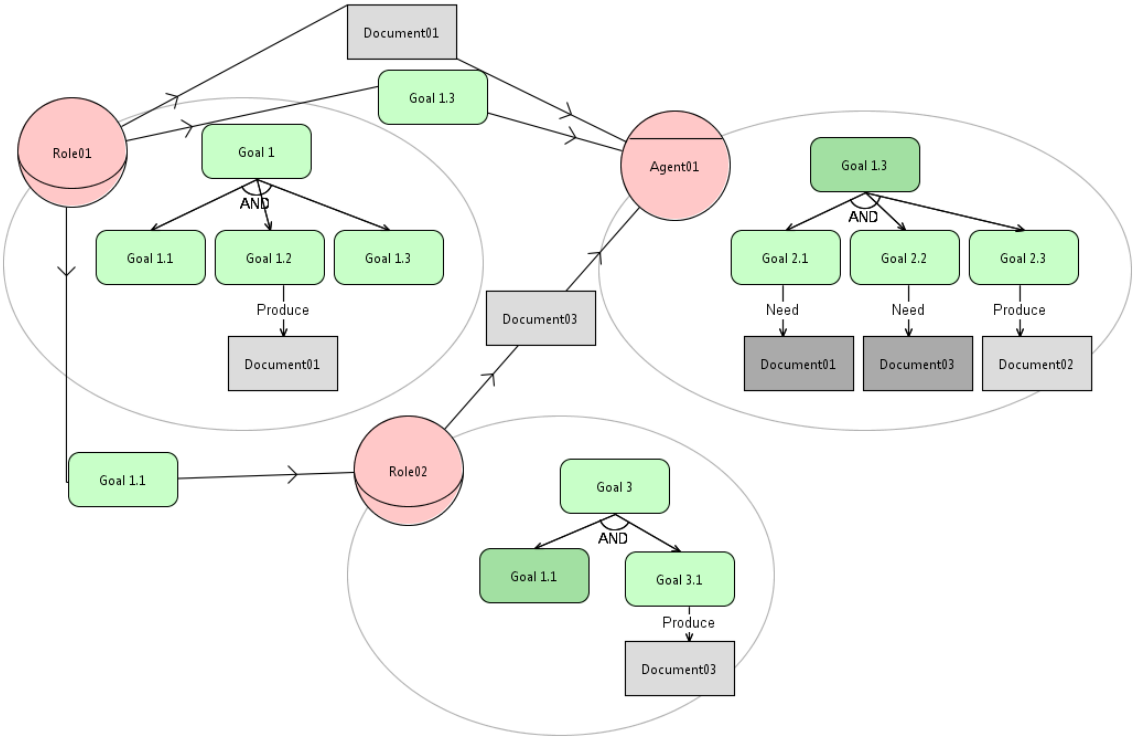


Figure 4.4: Social view example

The social view of our model (Figure 4.4) contains two actor roles (Role01, Role02) and a software agent (Agent01). The first actor (Role01) has a main goal which is decomposed to three sub-goals that can be either accomplished by the actor alone (e.g., Goal1.2) or delegated to others within the system (e.g., Goals 1.1, 1.3). A number of documents are produced and exchanged during the process, each containing different pieces of information necessary for the accomplishment of the system's goal. The matching of the informational assets to documents is illustrated by the Information view of the STS-ml model in Figure 4.5.

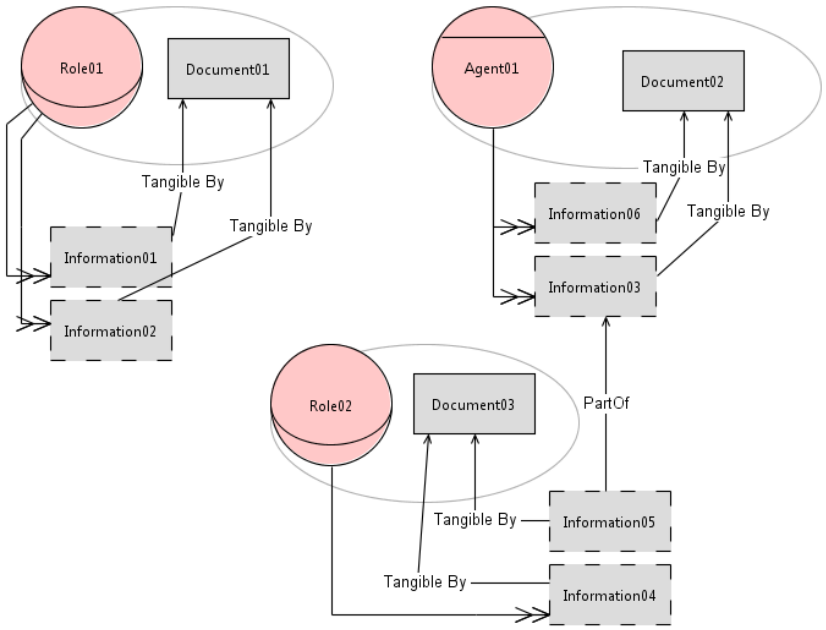


Figure 4.5: Information view example

A number of iterations may be required in order for the model to accurately illustrate the relationships within the system, but once the analyst confirms its maturity with the stakeholders the *prioritization process* can begin. During this stage the system stakeholders need to provide their input regarding the importance of each goal and informational asset of the system. First the system goals need to be ranked according to their importance. In order to do so, a value needs to be assigned to each sub-goal, which will represent its importance in the accomplishment of the top-level goal. In more detail, each top-level goal is considered individually and arithmetic values (FI_i) between 0 and 1 are assigned to each of its sub-goals. The aim here is to get a sense of ranking for the importance of each sub-goal so the values should be assigned in such a way that the sum for the decomposition always equals one (1) and a larger value represents a higher relative importance (Figure 4.6).By default, all sub-goals of a goal-decomposition are considered equally

important and have the same value (1 divided by the number of sub-goals), until an evaluation is performed resulting to new relative importance values. It may be easier for the stakeholders if they are asked to divide one hundred points (100) amongst the sub-goals of each top-level goal according how they perceive their importance and then those values can be converted to the appropriate format (from 0 to 1 with decimal digits) by the analyst.

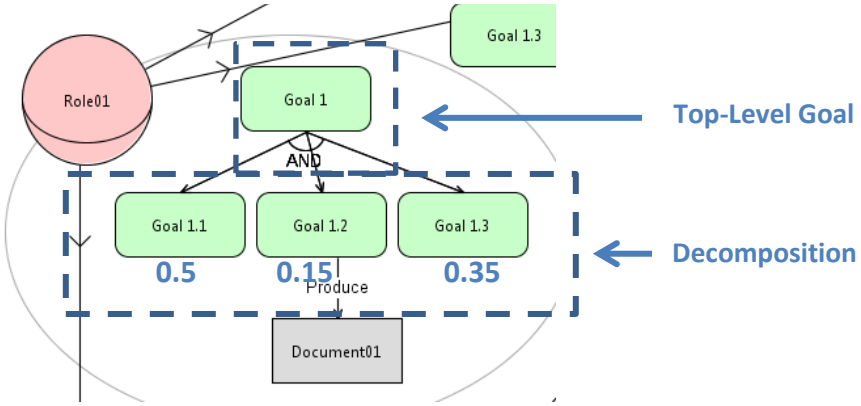


Figure 4.6: Example of assigning relative importance values to goal decompositions

For the purposes of our example we have assigned some mock relative importance values at every goal-decomposition of our model, as illustrated in Figure 4.7.

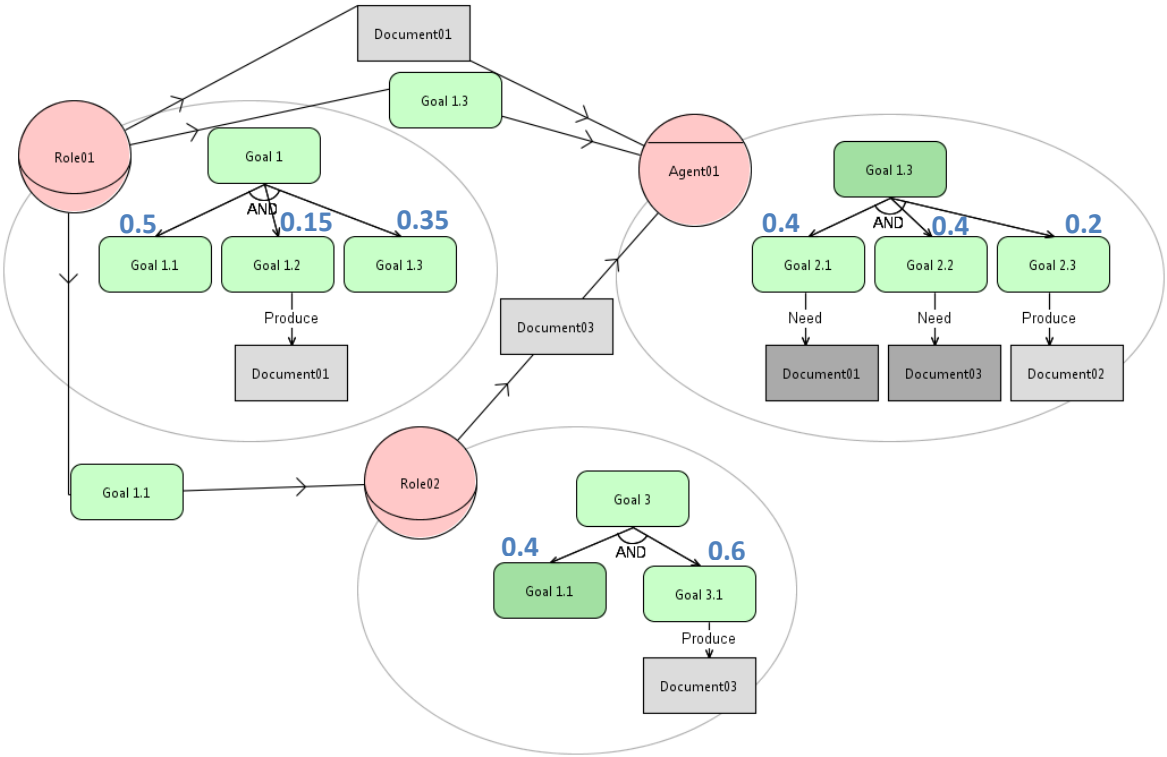


Figure 4.7: Social view with relative importance values assigned to goals

Next we turn our attention to the Information view model where the system stakeholders need, once again, to provide their input. Here the informational assets need to be evaluated according to their necessity for the achievement of the system's goals. Values from 0 to 1 are assigned to each informational asset (N_j), with a higher value representing information more critical for the achievement of the system's goals. What we aim to measure with this evaluation is how "useful" the stakeholders consider certain pieces of information for the completion of the overall process. Information receiving a higher necessity value will have a greater potential impact towards the achievement of certain goals if, for example, the document containing it was unavailable or missing.

When two informational assets are connected by a "Part of" relationship in our model, the necessity value of the asset on which the "Part of" relationship is directed should be at least equal to the highest necessity value of its components (e.g., at Figure 4.8, Information03 inherits the value of Information05 since they are connected with a "Part of" relationship). Once all informational assets are evaluated the document containing them inherits the highest of their necessity values (Figure 4.8). The rationale behind this decision lays in the fact that any given document is as important as the most critical piece of information it contains.

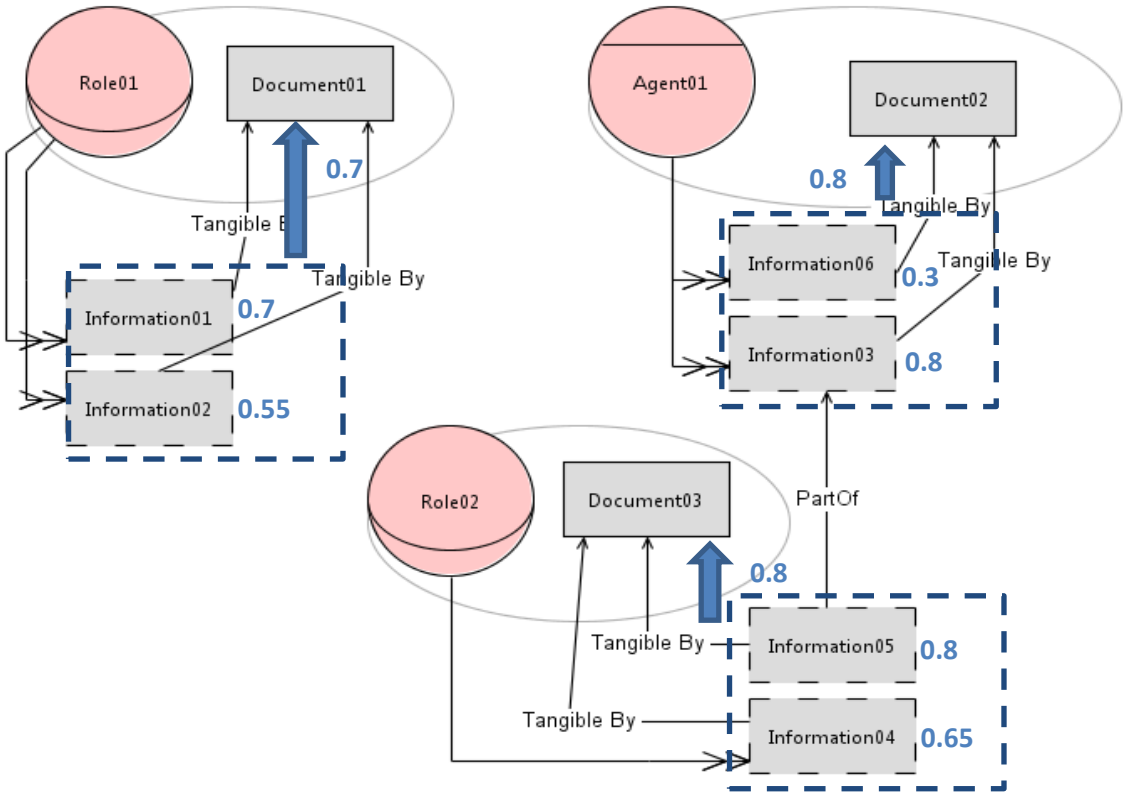


Figure 4.8: Information view with values assigned on informational assets

After all stakeholders have evaluated and ranked the goals and informational assets of the system, the analyst will *identify and rank potential risks*. In order to do so, threats to the informational assets of the system are first identified. The identification of these threats is left to the analyst's discretion, as there are a number of resources providing threat definitions to consult (e.g., ISO standards, threat repositories, etc.). In order to calculate the risk posed by each threat, its likelihood and impact values must be identified and plugged to the established risk formula, "Risk = Impact X Likelihood".

The likelihood of each event to threaten certain parts of the system is again calculated by the analyst after consulting various relevant resources available. The impact value however, can be easily calculated by the created model as follows. By following the goal-trees and delegations we multiply all the relative importance values (FI_i) of all the (sub-) goals between the main goal of the system and the threatened document. In order to identify the impact, we multiply the necessity value of the threatened document (N_j) with the product of the relative importance values identified before. In case of an event threatening multiple documents the same process is followed for each threatened document. The threat's individual impact values, for each threatened document, are identified and summed together to find out the overall impact of the threat (I_t). An example of this process is provided in Figure 4.9 where "Event01" threatens Document1 at Goal1.2 and Goal2.1. By following each of these goals to the main system goal (Goal 1) we multiply the relative importance values of all the goals in each path, which in our case are: $FI_{2.1}$ x

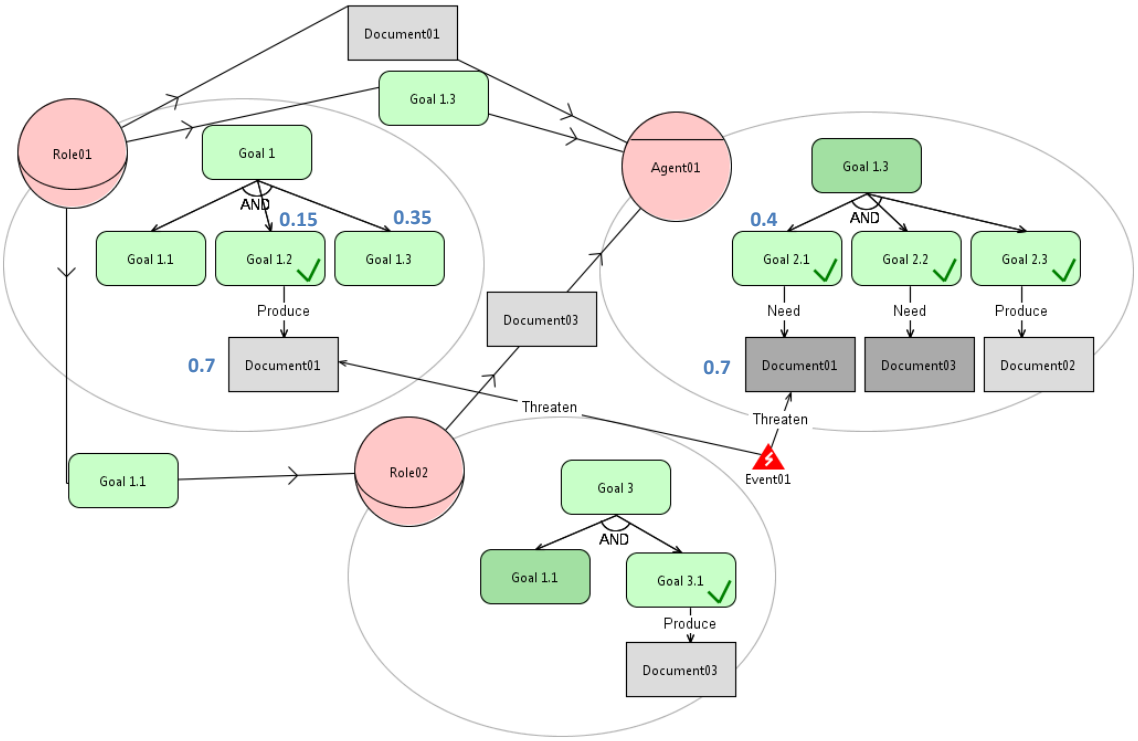


Figure 4.9: Example of threat impact identification through social view model

$FI_{1.3}$ (= 0.4 x 0.35) and $FI_{1.2}$ (= 0.15), and we multiply the sum by the necessity value of Document1 (N_1 = 0.7) to identify the impact of Event01, I_1 (= 0.7 x (0.4 x 0.35) + 0.7 x 0.15 = 0.203). For each event present in our model the same process is followed which, in the end, results in a list of impact values. The total risk value for each threat (R_t) is then calculated by multiplying the threat's impact (I_t) with its likelihood value (Li_t). Finally the risk values calculated can be ranked by highest to lowest to represent a ranking of threats from the most to the least severe for the system, as illustrated by the template provided in Table 4.3. This way the system stakeholders not only have a model of the system but also a clear overview of the most important threats to its functionality and security.

Threat	***Description***	**Impact** (I_t)	**Likelihood** (Li_t)	**Risk** (R_t)
"Event01"	*Brief description of threat*	0.203	0.3	0.0609
"Event03"	*Brief description of threat*	0.154	0.25	0.0385
"Event06"	*Brief description of threat*	0.085	0.4	0.0340

Table 4.3: Example of ranking threats

This simple process for calculating the impact of various threats may require some extra rules in order to better handle some unique situations that may arise in the created model. More specifically it is possible that the same sub-goal may be part of two or more separate goal decompositions (e.g., Goal1.1 in our example). When this happens the same sub-goal may have different relative importance values in each goal tree it is part of, thus complicating the impact calculation. To avoid this issue only the relative importance value at the goal-decomposition closer to the system's top-goal should be taken into account.

With all the identified threats ranked, the next step of the process requires from the stakeholders of the system to *express their security needs and provide the necessary authorizations* for their informational assets and delegated goals. In more detail, similarly to STS method, each stakeholder can define his security needs (non-repudiation, redundancy, non-delegation, trustworthiness, availability, integrity and confidentiality) for every document and goal it delegates to another actor in the system. Additionally, authorization needs to be provided for the usage, modification, production and distribution of informational assets exchanges through the system. The analyst then updates the model according to the input received by the stakeholders and adjustments can be made to the design to facilitate the expressed needs and authorizations.

This process may require a number of iterations until it produces a system design capable of satisfying all the requirements imposed by its stakeholders. The threat ranking earlier provided by our method, can assist an analyst to prioritize his efforts from the most to the least threatened system assets, during this demanding process. It can also be a helpful tool to support the decision making process of the analyst

when requirements or security needs overlap or conflict. In that case the threat ranking can be used by the analyst to assess which alternative is less risky for the overall design. Once the system design is finalized, the *automated analysis phase* follows, where, as in the STS method, the STS tool provides automated validation of the system design by identifying inconsistencies and errors. The same tool can finally produce a report with all the security requirements of the system which can be later used during the development stage.

4.3 Limitations

Certain issues with the developed method, as presented in the previous sections, need to be addressed. Firstly, as stated in the overview of the STS method, each goal can have an "AND" or "OR" decomposition to sub-goals in a socio-technical model. The goal evaluation process of our method can be successfully applied to "AND" goal-decompositions, by following the rules and process described in the previous section. However when an "OR" goal-decomposition occurs the established rules need to be adjusted. Since in "OR" goal-decompositions the top level goal can be reached by achieving only one of its sub-goals, there is no point to introduce relative importance values for its sub-goals. Nevertheless, our method can still accommodate such decompositions by giving a default value of one (1) as the relative importance of each sub-goal of an "OR" goal-decomposition. This way the rest of steps of the process and the underlying mathematical formulas can be applied without further modifications without sacrificing the quality of the outcome.

Another limitation of this method is the fact that it can assess risk and thus prioritize threats, only against the informational assets of a system. The STS method, which was the basis for the development of our method, allows for events to threaten goals and even actors in addition to informational assets. The process followed by our method requires a document (informational asset) at the lowest level of goal decompositions. This way a threat is tracked from a document at the lowest level all the way to the main system goal and its overall impact is assessed. Events directly threatening goals and actors require a new approach for risk assessment which is currently not covered by the capabilities of this newly developed method.

An additional limitation of this approach revolves around the accuracy of the stakeholders' evaluation of goals and informational assets of the system. Since they provide the values which are then used by the mathematical formulas of this method to calculate the risk, their input can heavily affect the accuracy of the results. As it is the case with any evaluation given by humans, errors and misjudgments can occur due to a variety of factors. In order to minimize the effects potential human errors it is important to include a large number of stakeholders in the evaluation

process, so over- or under-estimations affect less the average value assigned to each of the system's assets. It is also possible to follow pairwise comparisons or Delphi techniques, which compare the evaluations given by each stakeholder to each other and allow for modifications to be made until a consensus can be reached. However, the final decision is to be made by the analyst, according to the unique circumstances of each project.

It is also worth mentioning that since this is a newly created method, the support that existing CASE tools can provide is limited. While the modeling of the system can be covered by the STS Tool, which can also provide automated analysis, the calculations of the risk have to be performed manually for the time being. Another point that needs to be addressed is the dependency of this approach to external sources for threat definitions and their possibility of occurrence. These external sources are left to the discretion of the security analysts involved in the project and can vary from established international standards (e.g., ISO, NIST) to subjective personal experiences. Since the quality of the method's outcome is dependent on the quality of those external sources it is important that the analysts maintain high standards when selecting the resources which they will consult during these steps of the process. Nevertheless this limitation is not unique to our method, since the majority of methods in the area of risk management focus on assessing risks but use external approaches for the process of acquiring threat definitions and establishing their likelihood.

Chapter 5. Practical Evaluation

In this section the newly introduced method will be evaluated in practical settings in order to identify its level of performance and its shortcomings. Initially the process that will be followed for this evaluation will be overviewed and the criteria upon which the method will be judged will be introduced. Next some elaboration will be provided about the system that we selected for this case study. Its objectives, components and stakeholders will be presented and the rationale behind the system's selection will be elaborated. Since modeling the system is an essential phase of our method, an initial model of the selected system will be presented upon which our risk assessment process will be applied. Finally an overview of the output provided by our method and its overall performance will be discussed.

5.1 Method and evaluation criteria

In order to evaluate our method, its performance in practical settings should be assessed. In order to do so a software system has to be selected for the application of the method in a retrospective case study. The selected system should include a number of different agents (human or software) who interact with each other to accomplish certain goals. By applying our method, a social and informational model of the system will be created and goals and assets will be evaluated by the stakeholders in order to provide input for the risk assessment process. Finally, as an output of the application of our method, a complete list with the system's security requirements will be provided, accompanied by a prioritized threat list containing potential risks and a system model, where all interactions and their security needs are illustrated.

A number of different stakeholders need to be involved in the method's practical evaluation process. The role of the requirements analyst will be assumed by the authors of this work and creators of this method. The analyst is responsible for the creation of the system's model assisted by the input of various system stakeholders, who will provide documentation and their personal expertise, in order to achieve an accurate representation of the system. The stakeholders, as participants of the system, will also need to provide their input during the goal and asset evaluation, and additionally to express their individual security needs and authorizations which will be used in the later phases of the method to extract the security requirements of

the system. Finally, a security analyst will be required in order to identify threats towards the informational assets of the system and assess the likelihood of their occurrence. This role will be also assumed by the authors of this work, supported by literature of the field of information security (e.g., case studies, threat repositories, international standards, etc.) when necessary.

The *evaluation criteria* for the method's performance need to be defined, before we present the system selected for this case study. The *usability* of the developed method will be an important factor to be considered in this evaluation effort. Usability, in the context of our case study, represents the degree of difficulty faced by the analysts during the application of the method. In order to achieve a high usability rating, the method should be *efficient in its application* and provide clear instructions to the analyst for every step of the process. The capability of the method to sufficiently model, and therefore take into consideration, every security need of the system stakeholders is another evaluation criterion which can provide a solid indication of the method's modeling completeness. Finally, the method's *capability to uncover security requirements* will be another criterion for its evaluation. The setup of this retrospective case study allows us to compare the security requirements identified by our analysis and compare them to the security features already implemented in the system. If our developed method is able to provide a complete requirements list, containing requirements not uncovered during the initial system development, it will be an indication that our analysis produces a solid outcome, which can reveal overlooked aspects of the systems design.

5.2 Case study description

The system that will be studied during this retrospective case study involves the STRIPA software. STRIPA is described by its creators as "*a stand-alone decision support system that advices physicians during the pharmaco-therapeutic analysis*" (Meulendijk, Spruit, Jansen, Numans, & Brinkkemper, 2014). The Systematic Tool to Reduce Inappropriate Prescribing (STRIP) is a structured method aiming to assist physicians when dealing with patients receiving multiple medical drugs. The STRIPA software system aims to facilitate the application of the STRIP method in order to assist physicians optimize the patient's treatment plan and avoid over-prescription. The method follows a series of steps (Figure 5.1) in order to provide suggestions regarding adjustments to the medication the patient receives (e.g., removing redundant medication etc.) and its dosages, based on the patient's medical history and current symptoms.

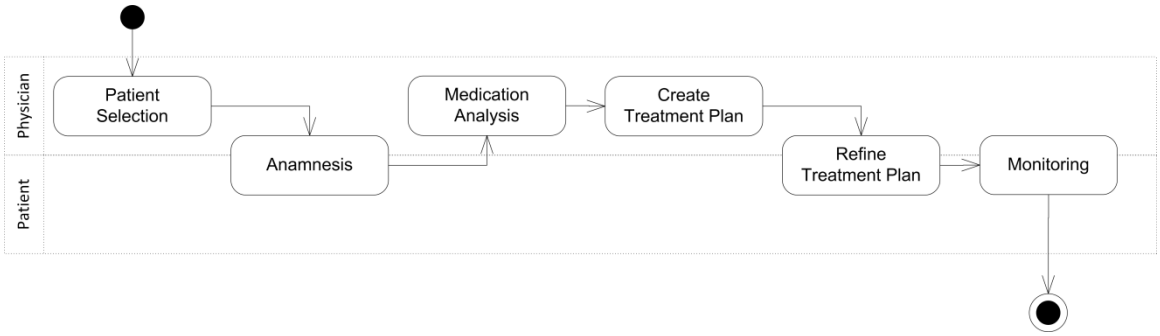


Figure 5.1: The STRIP method (Source: Meulendijk et al., 2014)

STRIPA functions as a standalone web-based application which communicates with the various information systems used by medical practitioners. This communication is bidirectional as the caregiver's IS provides patient information through medical records to STRIPA and STRIPA responds with treatment advice. Since the exchanged data contains highly sensitive personal information (e.g., patient's medical records) security is a major concern, thus making the extended socio-technical system around STRIPA a fitting candidate for the practical application of our newly developed method.

Another contributing factor is the fact the majority of the sensitive data is owned by the patient, who is also a stakeholder of the system, but is handled by the caregiver and exchanged with the STRIPA agent. Therefore the patient has to provide certain authorizations in order to make these information exchanges possible within the system. Since authorization management is an integral part of our method it will be interesting to be tested in real life settings.

Finally, since STRIPA is in the later stages of the development, with an existing prototype already available, it is a fitting candidate for our retrospective case study. The security features of the existing prototype can be used as a benchmark to compare against the security requirements produced from the application of our method. This allows us to get an immediate indication of the effectiveness of the method, within the timeframe of this thesis project, which would not be possible if a software system still in the early development stages was selected.

The first phase of our case study consists of the *system modeling*, where a representation of the main goals and information assets of the system is created (cf. Section 5.3). Throughout the modeling process the input of the architect and main designer of the STRIPA system is used for the refinement of the system's model. Once the modeling is complete several participants of the socio-technical system surrounding STRIPA provide their evaluation on the goals and information of the system. The stakeholders here range from medical practitioners to regular patients

and the evaluation provided by each of them will be taken into account during risk assessment. For our case study nine user evaluations are provided, six by healthcare practitioners familiar with the STRIPA system, one by the lead designer and developer of the STRIPA software agent, and two by non-technical subjects representing potential patients using the system. The template evaluation form used for this purpose can be found in Appendix A.3.

For the *threat assessment* process an independent security analysis will be performed by the authors, in order to identify a number of potential threats to the system and estimate their likelihood of occurrence, based on literature sources and similar past research attempts. These findings will be then used for the risk calculation and threat prioritization. Finally, some *user security requirements* need to be expressed and modelled, over each user's security needs and authorization preferences for his goal delegations and information exchanges within the system. Once the user preferences are inputted on the system model, the *automated analysis* provided by the STS Tool will check its consistency and completeness and extract a list with the *security requirements* expressed by the model. This requirements report, along with the system model will be provided to the creators of STRIPA and will also be used for the evaluation of the methods according to the criteria discussed in the previous section.

5.3 System modeling

Before risk analysis can take place, a model of the studied system has to be created, as indicated by the first two phases of the newly introduced method (*Social Modeling* and *Information modeling*). As already discussed in previous sections, the social and information view of the system need to be modeled by the analyst, according to the information provided by the systems stakeholders. Once the initial model is approved by them, the *Threat assessment* phase of the method can be initiated (cf. Section 5.4). The system modeling is performed by following the process, rules and notation established by the STS-ml, using the STS Tool. In this section the system model will be presented (Figures 5.2, 5.3) and its participants, assets and functionalities will be explained.

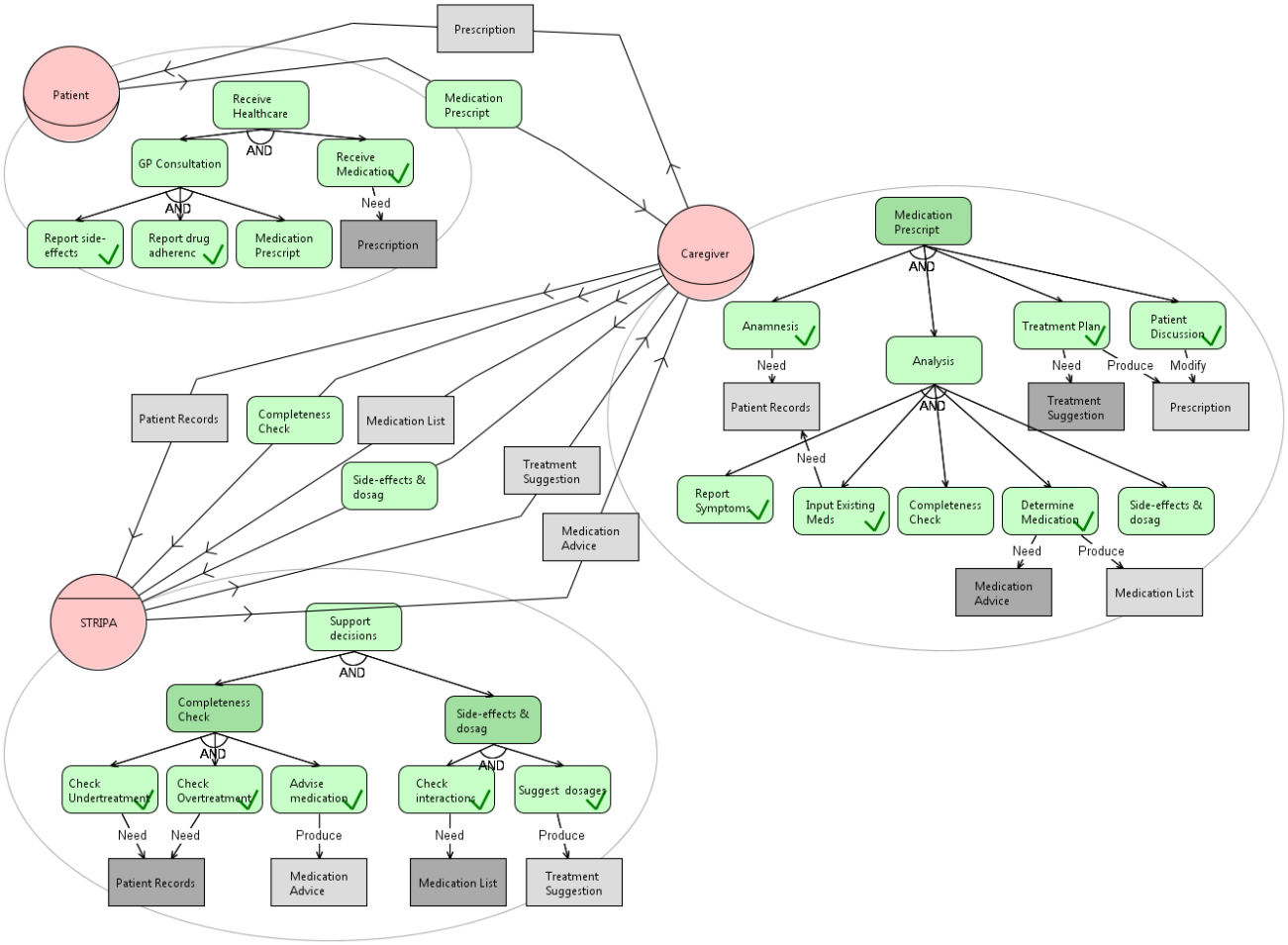


Figure 5.2: Social view modeling of the studied system

The first actor participating in the studied system represents the "Patient" who is receiving multiple medications for a number of medical conditions. The main goal of the patient is to receive healthcare and in order to do so he must consult a "Caregiver" to receive a prescription which he can then use to obtain the appropriate medication. During the consultation the patient must report the medication he is already receiving, his adherence to it and its potential side-effects before new medication can be prescribed by the caregiver.

The role of the "Caregiver" can be played by a range of medical practitioners, ranging from general practitioners to pharmacists. The caregiver has an individual information system which keeps and update the medical records of each of the caregiver's patients and communicates with the "STRIPA" agent in order to receive support during the decision making process of medication prescription. The process of prescribing medication requires a number of steps to be followed by the caregiver. During the "Anamnesis" phase the caregiver consults the patient records to

determine the patient's actual drug-use, reported side-effects and existing problems. During "Analysis" phase the symptoms that need to be treated and the existing medication of the patient are checked in order to identify the completeness of the already existing treatment plan. Here the caregiver's IS receives the input of the STRIPA system, which checks for over-treatment or under-treatment of the patient's symptoms and then suggests medication adjustments to correct the treatment plan. These suggestions are taken into consideration by the caregiver when determining the patient's new medication plan. As soon as the caregiver decides on the new medication plan, the STRIPA system automatically checks for counter-interactions between the selected drugs and their potential side-effects while also suggesting a dosage based on the patients individual characteristics.

This treatment suggestion produced by the STRIPA system is the input for the next phase of the process, "Treatment Plan". During this phase the suggested treatment plan produced with the support of the STRIPA system is evaluated by the caregivers in order to make a final decision and produce the prescription which the patient will use to obtain his medication. Once a final decision is made the final phase of the process, "Patient Discussion" takes place. Here the caregiver discusses the treatment plan with the patient and makes final adjustments according to the patient's needs. Finally the patient receives the prescription and obtains his medication.

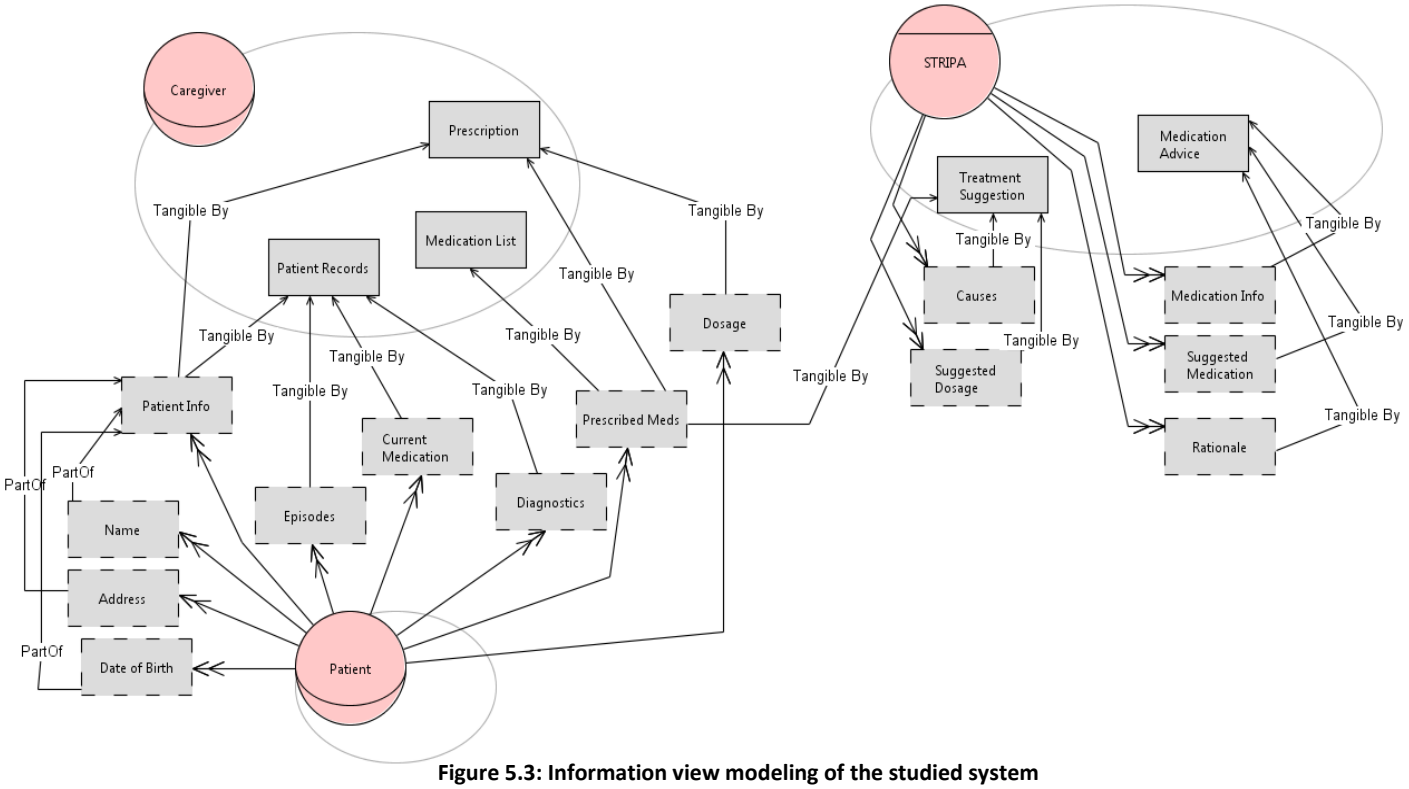


Figure 5.3: Information view modeling of the studied system

The documents exchanged during this process along with the information they contain and its ownership are modeled in the information view (Figure 5.3). The "Patient Records" document, stored in the caregivers IS, contains the following informational assets: personal information of the patient (name, address, date of birth), the patient's medical history (e.g., past episodes), diagnostics (e.g., past lab tests, x-rays, etc.) and current medication. The newly prescribed medication and its dosages are also informational assets belonging to the patient which are contained at the "Medication List" and "Prescription" documents created by the caregiver. The "Medication List" is an intermediate document created for the communication between the caregiver's IS and the STRIPA agent, in order to receive decision support. The "Prescription" is the final document issued by the caregiver to the patient in order for the latter to obtain the appropriate medication.

The STRIPA system also produces a number of documents containing certain information. The "Medication Advice" document contains informational assets regarding the suggested medication for the patient's symptoms, information about this medication (e.g., commercial name, active ingredient etc.) and the rationale behind those suggestions. The "Treatment suggestion" document is the final document produced by the STRIPA which includes the medication selected by the caregiver, the suggested dosages advised by the system, and possible causes of unwanted interactions between them. This document is used by the caregiver when making the final decision on the patient's treatment plan.

5.4 Threat assessment

The first activity during the threat assessment phase is to *identify potential threats* to the system. As earlier discussed, the threat identification is based on literature sources and international information security standards. Based on documented case studies of information security for medical information systems – namely (Samy, Ahmad & Ismail, 2010), (Vaast, 2007), (Maglogiannis & Zafiropoulos, 2006) and (Rindfleisch, 1997) – various threats to medical information systems were reviewed and the ones applicable to our system were selected. The same sources were used to obtain the *likelihood values for each threat*, which had to be subjectively adjusted by the authors, to fit the requirements of our method, as in some cases they were expressed in qualitative scales (e.g., high to low likelihood) instead of a numerical value between zero and one. The selected threats to be modeled, their definitions according to the ISO standards for information security management in health (ISO/IEC, 2008) and their likelihood estimations, adapted by literature but adjusted by the authors, are provided at Table 5.1

Threat	Definition	Likelihood
Communications infiltration	*"Communications infiltration of electronic communications occurs when an individual (a hacker, for example) tampers with the normal flow of data across a network."*	0.25
Technical failure of the host	*"These threats include hardware failures, network failures or failures in data storage facilities, […], the loss of availability of such systems can have life-threatening consequences for patients."*	0.87
User error	*"Error by users can, for example, result in confidential information being sent to the wrong recipient or having input errors."*	0.81
Masquerade by insiders	*"Masquerade by insiders consists of system use by those who make use of accounts that are not their own. […] As such, it constitutes a breakdown in secure user authentication."*	0.41

Table 5.1: Identified threats and likelihood of occurrence

The threats of "*Infiltration of Communications*" and "*Technical Failure of the host*" could affect all documents exchanged between the caregiver's IS and the STRIPA agent. Therefore, the events representing those threats are modeled on the social view of our system's model (Figure 5.4) connected with a "*Threatens*" relationship with four documents; Patient Records, Medication List, Medication Advice and Treatment Suggestion. The threat of "*User Error*" can threaten the documents created by the caregiver, as they are created manually and are prone to human errors. Therefore the document representing the Prescription, used during the patient discussion stage and later by the patient to acquire his medication and the Medication List document, submitted for validation at the STRIPA agent are potential targets for this threat. Finally, the "*Masquerade by insiders*" threat represents unauthorized access to sensitive documents through the caregiver's IS (in our case Patient Records and Prescriptions) by "*insiders*". Usually the unauthorized access does not have malicious motivations as the "*insiders*" can be colleagues, supporting staff or maintenance personnel, who make use of the caregiver's account to assist him. Nevertheless, sensitive patient information can still be potentially compromised.

In order to calculate the overall risk imposed on the system by each of the identified threats, their individual *impact* has to be identified. As previously described, the impact value of each threat is defined by multiplying the relative importance values of the information assets and linked goals affected by the threat, starting from a document and reaching until a top-level system goal. The importance values are calculated by averaging the evaluations provided by various system stakeholders. For our case study, nine stakeholders provided evaluations of the relative importance of the goals and information assets included in our system, which were averaged to the values included in the system model illustrated at Figures 5.4 and 5.5. Their individual evaluations as well as the average values are listed in Appendices A.4 and A.5 of this work.

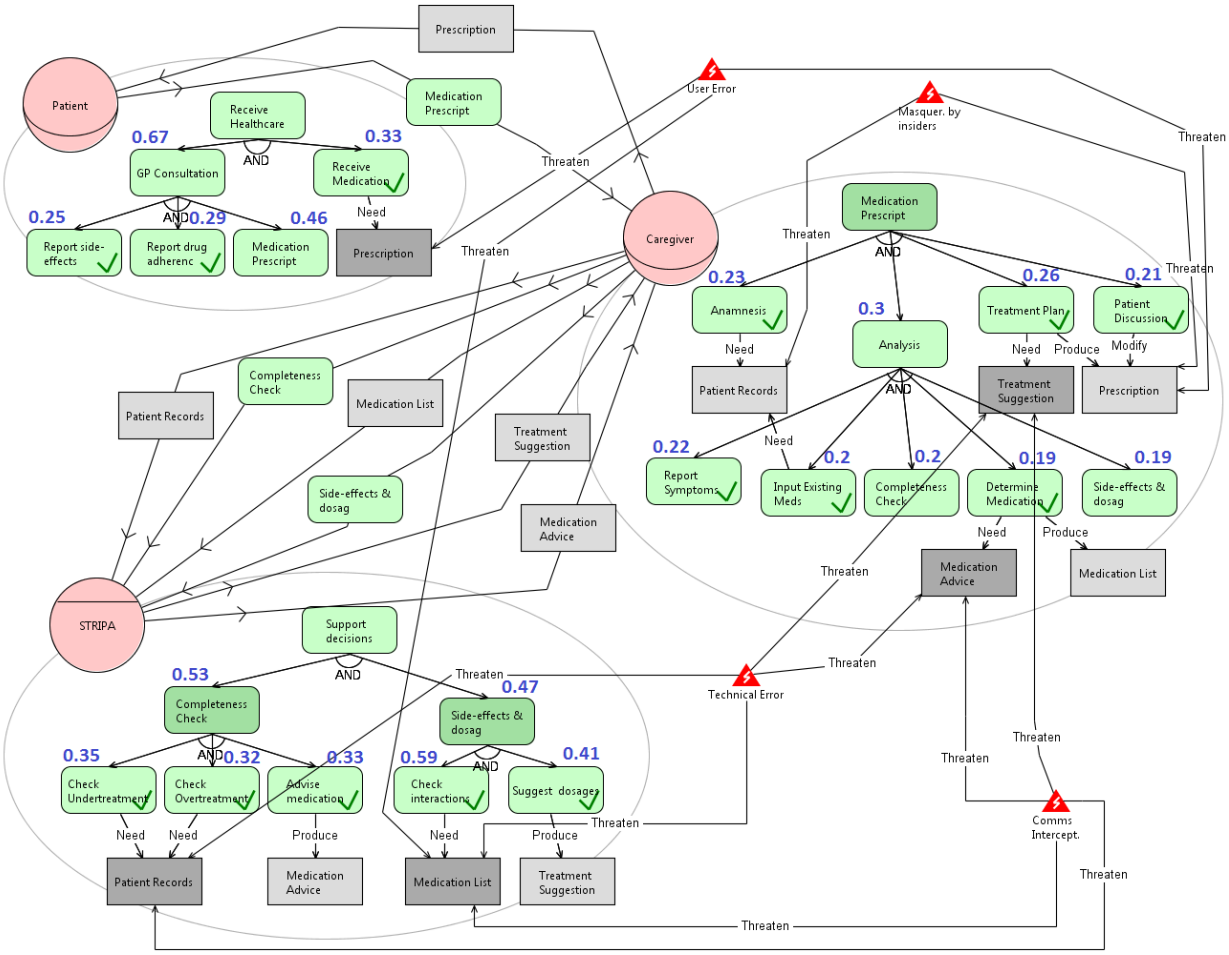


Figure 5.4: Social view of the system including relative importance values and threats

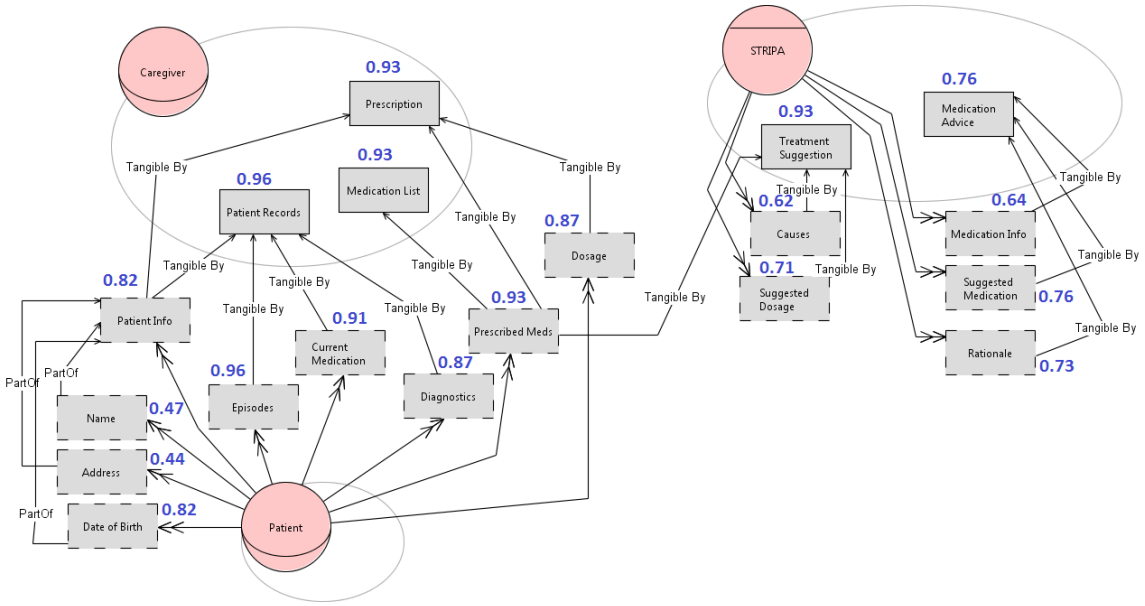


Figure 5.5: Information view of the system with relative importance values

By tracking all paths beginning from each event and reaching the top system goal ("*Receive Healthcare*") and adding their individual impact values calculated by the relative importance values as earlier described in section 4.2, the total impact of each threat can be identified. The total impact value for each threat of our system, along with its likelihood and its subsequent total risk value, are included in Table 5.2. The table can be used to rank threats according to their overall risk, which can be used as input during the next phase, when the users' security requirements (security needs and authorizations) are elaborated.

Threats	**Impact** (I_t)	**Likelihood** (Li_t)	**Risk** (R_t)
User error	0.377	0.81	0.3051
Technical failure of the host	0.109	0.87	0.0952
Masquerade by insiders	0.128	0.41	0.0526
Communications Infiltration	0.109	0.25	0.0274

Table 5.2: Threats ranked according to their potential risk towards the system

5.5 User security requirements modeling

Based on the identified threats and the security needs of the system's stakeholders, a set of constraints are modeled over the goal delegations and document provisions within the system. As previously discussed the security needs supported by STS-ml include non-repudiation, redundancy, no delegation, trustworthiness and availability for goal delegations and integrity, availability and confidentiality for document transmissions. In our case the security needs imposed on the system are influenced by the risk inherent to the system and its extended environment as illustrated at Figure 5.6. In more detail, the "*Prescription*" document exchange between the caregiver and the patient should have the *integrity* and *confidentiality* of its transmission guaranteed, since it can be threatened by events such as user errors and unauthorized access of the caregivers IS by insiders, as earlier discussed. Regarding the goal-delegation of "*Medication prescription*" between the same actors, the needs of *trustworthiness* is imposed, since a level of trust of trust between the two actors is a prerequisite for the completion of the healthcare process.

More importantly, regarding the document exchanges between the caregiver's IS and the STRIPA system, the need for *availability* is apparent since the communication between the two agents is threatened by potential technical errors of the host. Such event can disrupt the communications between the two agents making the provision of documents necessary for the process impossible. Therefore

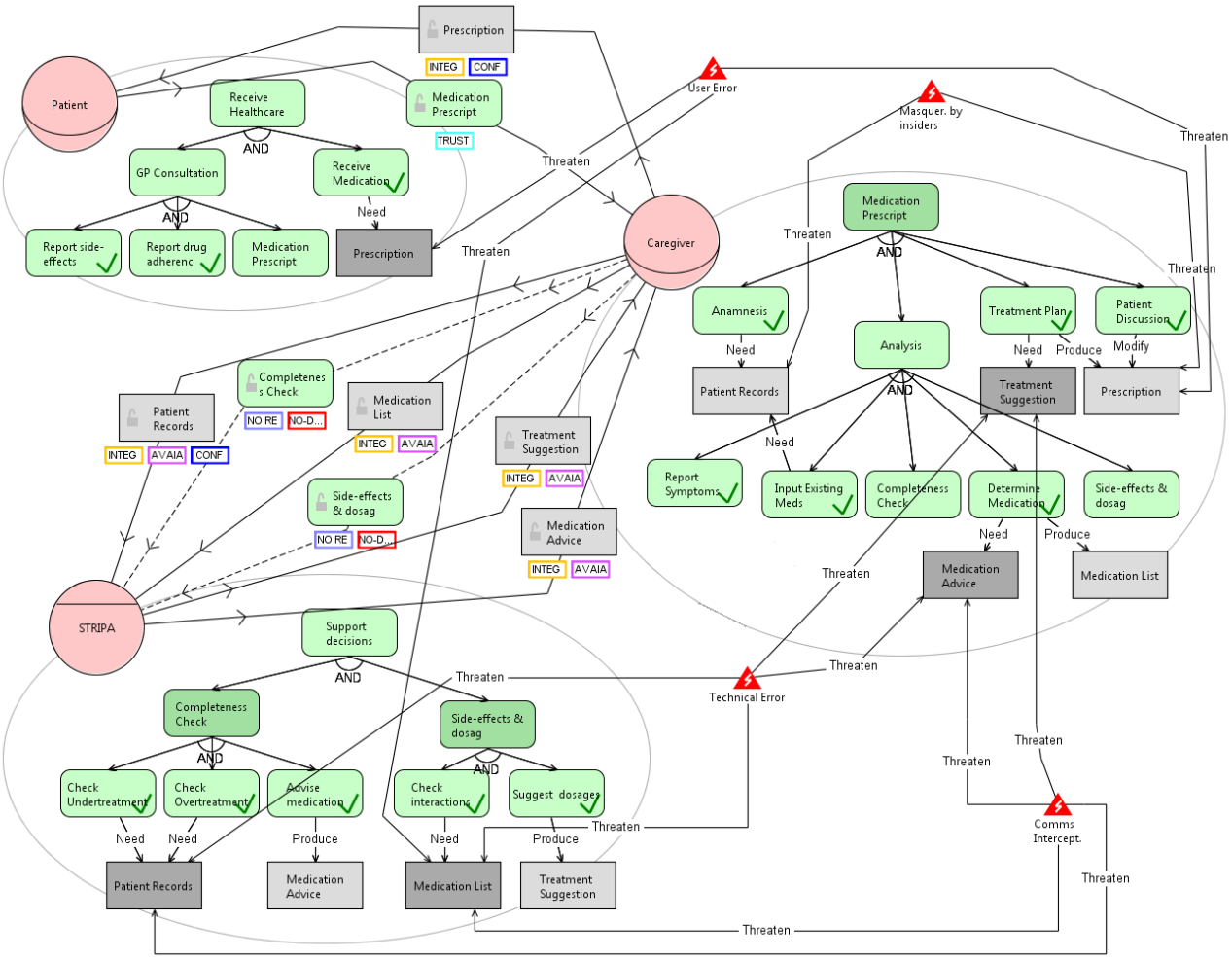


Figure 5.6: Security needs of users expressed on the social view of the system

the implementation of the system should provide mechanisms that guarantee the *availability* of the "*Patient records*", "*Medication List*", "*Treatment Suggestion*" and "*Medication advice*" documents. The *integrity* of the transmission of the same documents should also be guaranteed, as it can be threatened in the event of an infiltration to the communication channels between the two systems. In the case of the "*Patient Records*" document the need for *confidentiality* of its transmission, since it can be compromised by unauthorized access from a third party at the caregivers IS or the STRIPA system. For the delegation of the "*Medication completeness check*" and "*Side-effect and interactions check*" goals from the caregiver to the STRIPA

system the needs of *non-repudiation* and *no delegation* are imposed based on the same threats. The need for non-repudiation requires the adoption of security mechanisms that guarantee that the two sides cannot repudiate the occurrence of the delegation. The no-delegation security need ensures that the delegated goals are trusted to the STRIPA system to accomplish by itself, without allowing further delegation to other actors or agents for their completion.

Apart from the identification and modeling of the user's security needs, a representation of the user's authorizations is necessary to determine if information is exchanged and used in compliance with the restrictions imposed to the system. The modeling of such authorizations is included at the third view of the system's model according to the rules and notation introduced at STS-ml, as illustrated in Figure 5.7.

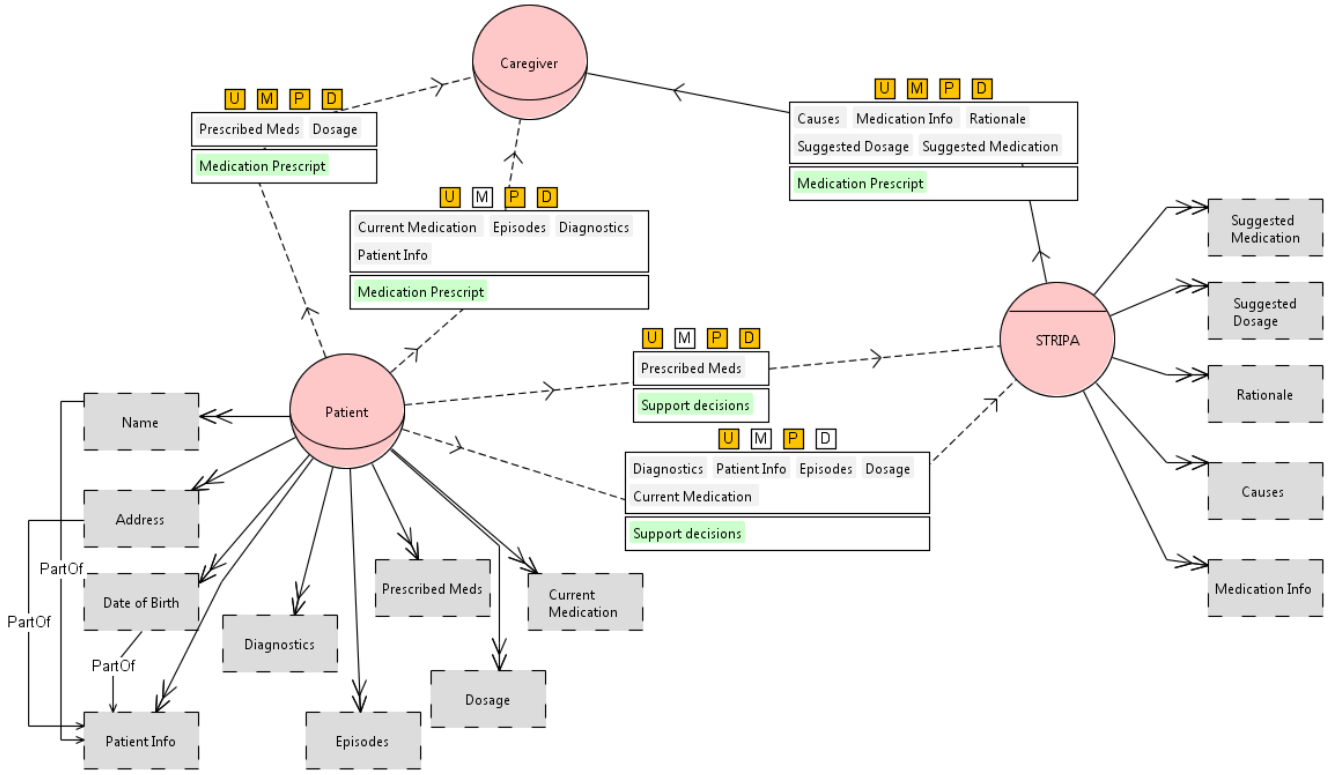


Figure 5.7: Authorization view of the studied system

In the above model of authorizations, each actor of the system provides the rights of usage, modification, production and distribution (U,M,P,D) of the certain pieces of information he owns, necessary for the completion of goals delegated to other actors or agents of the system. In our case the patient gives full authorization to the caregiver to use, modify, produce and distribute the patient's information regarding his newly *Prescribed Medication* and its *Dosage* in order to complete the "Medication

prescription" goal. The same actor also provides "use", "produce" and "distribute" authorizations for the rest of the information included in his personal records (*Episodes, Diagnostics*, *Patient Info* etc.) to the caregiver, though this time withholding the authorization to "modify" that information, as this is not necessary for the specific goal. The patient has to also provide authorizations towards the STRIPA system as it also makes use of certain pieces of his information when accomplishing its goal of supporting the decision making process of the physician. In more detail, the patient authorizes STRIPA to "use", "produce" and "distribute" his information regarding his newly *Prescribed Medication.* For the rest of his informational assets (*Episodes, Diagnostics*, *Patient Info* etc.) only the "use" and "produce" authorizations are required from the patient for the achievement of STRIPA's goals. Finally, the information owned by the STRIPA system, namely *Suggested Medication, Suggested Dosage, Rationale, Causes* and *Medication Info,* is given full authorization to be used, modified, produced and distributed by the caregiver in order to successfully complete the medication prescription process. After all the security needs and authorizations are identified the final phase our method takes place, automatically checking the consistency of the created model and extracting the security requirements described in it.

5.6 Automated analysis and specification

Initially the consistency check of our final system model (its different views presented at Figures 5.5, 5.6 and 5.7) reveal an authorization violation regarding the modification of "*Patient Info*" included in the *Prescription* document during the *Patient Discussion* activity. During this activity the caregiver can modify the prescribed treatment plan included in the Prescription document, according to the feedback received by the patient, but the violation occurs since the same document also contains the patient's personal information (name, address, date of birth) for which the patient has not provided the "modify" authorization to the caregiver. In order to resolve this issue a new iteration of information view of the system was created were the Prescription document is comprised of two interim documents, one containing the Patients Details, making the "*Patient Info*" informational asset tangible, and a second one containing the Patient's Treatment information, which can be modified by the caregiver. When the two documents are put together they create a Prescription document which the patient can then use to acquire his medication. This change in the structure of the Prescription document, illustrated at the information view of the model at Figure 5.8, does not cause any further modifications in the rest of the system's model so it is a relatively easy fix for this potential authorization inconsistency.

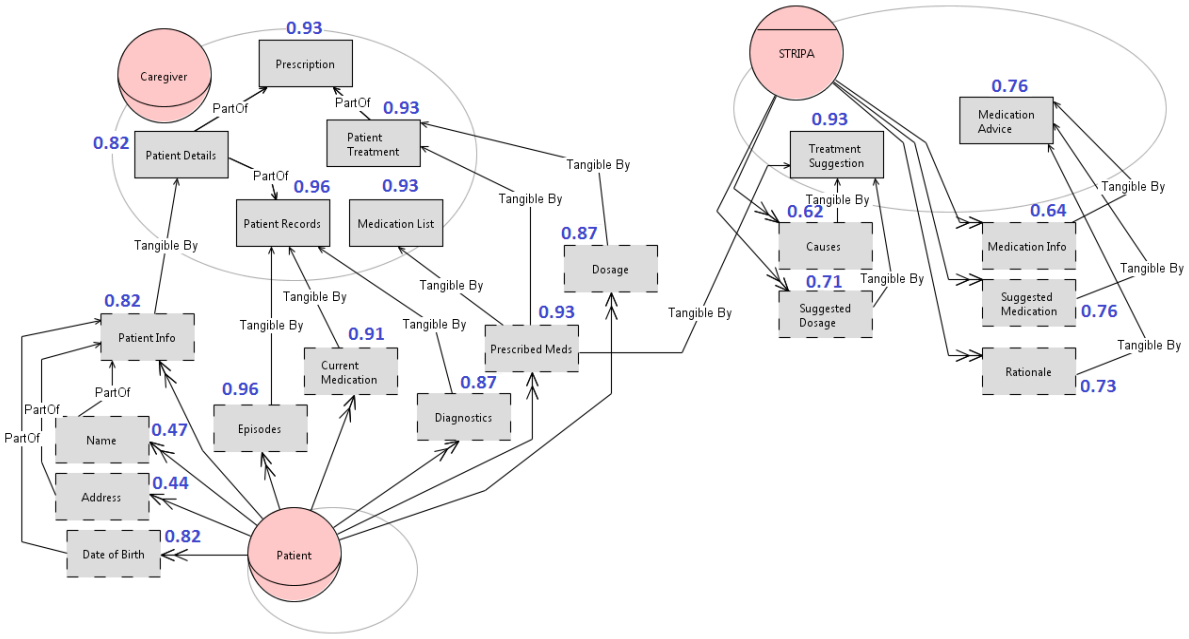


Figure 5.8: Final version of the information view model

The security requirements represented in our system's model will finally be extracted by the STS-Tool which can automatically generate a security requirements document, representing the final deliverable of our case study. For the overall system twenty five (25) security requirements were identified (cf. Appendix A.6 for the complete list), thirteen (13) out of which are the responsibility of the STRIPA system. They identified security requirements for the STRIPA system, are listed below at Table 5.3.

#	Description
1	Caregiver requires no-delegation for goal *Completeness Check*, when delegating *Completeness Check* to STRIPA.
2	Caregiver requires non repudiation-of-acceptance for goal *Completeness Check*, when delegating *Completeness Check* to STRIPA.
3	Caregiver requires no-delegation for goal *Side effects & dosage check*, when delegating *Side-effects & dosage check* to STRIPA.
4	Caregiver requires non repudiation-of-acceptance for goal *Side-effects & dosage* check, when delegating *Side effects & dosage check* to STRIPA.
5	Caregiver requires STRIPA to assure the availability the document *Medication List*.
6	Caregiver requires STRIPA to assure the availability the document *Patient Records.*
7	Caregiver requires STRIPA to ensure integrity of transmission over the provision of document *Medication Advice*, when STRIPA provides *Medication Advice* to the Caregiver.
8	Caregiver requires STRIPA to ensure integrity of transmission over the provision of document *Treatment Suggestion*, when STRIPA provides *Treatment Suggestion* to the Caregiver.

9	Patient requires STRIPA non-modification of Information: *Diagnostics, Patient Info, Episodes, Dosage* and *Current Medication*.
10	Patient requires STRIPA non-disclosure of Information: *Diagnostics, Patient Info, Episodes, Dosage* and *Current Medication*.
11	Patient requires STRIPA need-to-know of Information: *Diagnostics, Patient Info, Episodes, Dosage* and *Current Medication*, in the scope of goal *Support decisions*.
12	Patient requires STRIPA non-modification of Information: *Prescribed Meds*.
13	Patient requires STRIPA need-to-know of Information: *Prescribed Meds*, in the scope of goal *Support decisions.*

Table 5.3: Security requirements identified for the STRIPA system

5.7 Evaluation and conclusions

After the method's application at the socio-technical environment of the STRIPA system, a multi-faceted model of the system has been created and a list of security requirements has been produced. As discussed earlier (cf. Section 5.1), a series of criteria have to be applied in order to evaluate the method's performance.

The *first evaluation criterion* is the *usability of the method*, in terms of problems faced during its practical application. In our attempt the method was applied with minimal issues at the studied system. By following the method's steps and rules we were able to successfully construct a high-level model of the system, with the input and approval of its stakeholders. The asset evaluation was also performed without any issue by following the method's instructions. The only minor issue, not explicitly covered by the method, was the impact calculation when the same goal was part of more than one goal decompositions and therefore had different relative importance values at each goal-tree. This issue was resolved by taking into account the goal's relative importance value at the decomposition closest to our main system goal.

Another important point of our evaluation was the *capability* of the method to express and model all the security needs of the system's stakeholders. Our method was able to successfully express and model a number of security needs and authorizations, which were based on the newly-introduced threat prioritization process. Those user security requirements, tailored to the specific security needs of the system, were automatically checked against the created system model to evaluate its compliance. As already discussed in the previous section (cf. Section 5.6), a minor redesign of the structure of the Prescription document resulted from an authorization violation of the proposed design, as part of the automated model checking. This indicated that the security needs of the users were not only able to be expressed via our method, but also to result in modifications of the system's structure in order to be fully accommodated by the final design.

The *third and final evaluation criterion* was the security requirements uncovered by our method and how they compare against the security features already implemented in the studied system. Since our the STRIPA system, the focal point of our retrospective case study, has been already developed in a prototypical stage we were able to cross-reference our identified security requirements with the functionalities already implemented at the system. With the contribution of STRIPA's lead designer and developer, seven (7) out the thirteen (13) identified requirements – namely requirements number *2,4,5,6,7,8* and *12* from Table 5.3 - have not been, fully or at all, considered and implemented in the current version of STRIPA. Therefore the ability of our method to uncover previously unidentified security-related aspects of the studied system, provides a solid indication of its effectiveness in practice.

Finally, in addition to this case study's contribution to the method's evaluation, some of its limitations are also worth discussing. To begin with, our case study focused on one aspect of the STRIPA system's functionalities, specifically the provision of support in the decision making process of the caregiver during prescription. Nevertheless, there are a number of additional functionalities offered by the system, overviewed at Meulendijk et al. (2014), which were not elaborated in our study as they were outside the scope of this thesis project.

It is also worth noticing that the functionalities which were included in our case study could be analyzed in even greater granularity, adding to the precision of the produced design but producing larger models, too detailed for the purposes of this study. Nonetheless the current level of detail of our model proved to be more than adequate for identifying a number of security requirements and it could provide a solid starting point for later phases of the system's development.

A final limitation of the present case study lays in the identification and assessment of the threats of the studied system. The input of a security expert is required by our method for these activities, in order for a thorough identification of potential threats and their likelihood of occurrence. As this role was assumed by the authors of this work using mainly literature resources, a limited number of threats were identified to facilitate the application of this method and the likelihood of their impact was subjectively evaluated. Therefore, it is possible that more threats could surface, if a security expert with superior past experience and resources on the subject of information security repeated this process. Nevertheless, even with a sub-set of the overall threats identified, our method was able to rank them and provide some guidance regarding the priorities of the user security requirements identification. In future case studies, with a greater number of threats identified, the effects of their prioritization could become more apparent, when selecting which requirements to be implemented.

Chapter 6. Discussion

In this chapter, we present a discussion regarding the research findings and conclusions of this thesis. Initially we will discuss the outcome of each of the research questions introduced in the beginning of our research (cf. Section 1.3 – Research Questions). Since every research attempt presents its limitations, an evaluation of the weaknesses of our research will be provided next. Finally, we will elaborate on some future directions of our work in order to provide a direction and some objectives for similar research attempts in the future.

6.1 Conclusions

RQ1	*What does the literature suggest for identifying risks and security requirements in software systems?*
SQ 1.1	Which are the state-of-the-art methods for security requirement engineering?
SQ 1.2	Which are the state-of-the-art methods for risk analysis in software development?
SQ 1.3	What shared insights and empirical evidence can be found in the literature of those fields?

The first research question with its three sub-questions was answered via the findings of the literature study (cf. Chapter 2). Regarding the state-of-the-art in methods for security requirements engineering, initially, we discussed the contribution of modeling languages in security requirements engineering process. As security is a non-functional requirement, it cannot be quantified and measured as other functional requirements of a designed system. The creation of system models allows analysts to identify security issues via the system's design and therefore have a clear picture of what security requirements need to be imposed on the system, early in the development process, preferably during the requirements phase of the software development lifecycle. Since model-based approaches seem to be adequately equipped to handle complex system designs, they were a good fit for the needs of our project.

By filtering the literature of the area of model-based methods for security requirements engineering a number of established methods were identified and further grouped according to their origins. According to our findings, the earliest model-based methods are based on the concepts of the UML and are mainly used as a complementary tool to model basic user interactions with a software agent mainly using different variations of UML's use case diagrams. UML-based methods

identified through our literature review included abuse case modeling, misuse cases, anti-models and UMLsec. A common limitation of this category of methods was their inability to model complex interactions occurring in modern software systems were the complexity is increased as the system scales upwards with the introduction of more actors, agents and interactions between them. Another limitation of UML-based methods is the fact that they lack the tools and concepts needed to approach security from a social perspective.

Since the analysis of the social interactions taking place in software systems is a major factor that has to be considered during the design of modern systems, we turned our attention to goal-oriented modeling approaches to security requirements engineering. According to the literature of the area, goal-oriented methods are more adept at describing the complexity of interactions occurring in such systems, as they include concepts such as goals and actors that allow the modeling of high-level, social aspects of the system aligned with the strategic objectives of the organization, often aided by automated support tools. A number of goal-oriented methods were identified, Secure i*, Secure Tropos and STS-ml were overviewed in order to identify whether they offered concepts and techniques fitting with the goals of our research.

Risk management was another point of interest during the literature study part of our research, as reflected by the second sub-question. The literature of the area suggests that risk management is a process consisting of multiple phases that is applied to an existing system design to evaluate, control and minimize the consequences of potential events threatening it. Risk analysis, the initial phase of the risk management process, is mainly focused on the identification of potential threats and the estimation of their impact and likelihood of occurrence. Certain elements of these activities were already present, but less mature, in goal-oriented methods earlier identified. By examining a number of risk management methods - namely CORAS, CRAMM, Octave, ISRAM, ISRA-BM – we were able to identify useful fragments that can be added to a method for security requirements elaboration and will result in a more complete risk analysis process, thus facilitating the integration of risk management elements in the security elaboration process during the early stages of the development lifecycle.

Finally, the third sub-question for the literature part of our study aimed to identify critical factors and shared insights in the field of security requirements engineering and risk management. The early adoption of the process was a very important factor highlighted in a number of literature sources. Elaborating on security and potential threats of the system from the early stages of its development lifecycle leads to "*secure by design*" systems, with less chances of requiring costly redevelopments as result of poorly designed security features. Another critical point is the consideration of social and organizational aspects during the security elaboration process, which allows analysts to better understand and map the environment in which their

system to-be will function. The scalability of the selected approach for security was also considered as an important factor, as any method used should be able to handle the increasing complexity of interactions in modern socio-technical systems by producing comprehensible and intuitive system models. The existence of automated support tools for security requirements engineering model-based methods (e.g., CASE tools) is a contributing factor to the reduction of the complexity of the analysis as well as the time and manpower required for the process, since such tools offer automated consistency checks and requirement extraction from the created system models. By taking all the above factors into account while also considering the individual characteristics of each of the identified methods and their fit to our project needs, we were able to make a selection of relevant method fragments that were later used in the development of our method.

RQ2	*How are risks assessed and security requirements identified and specified in practice?*
SQ 2.1	When, how and by whom is security and risk elaborated during software development in practice?
SQ 2.2	Which factors influence the selection of a method for security requirements engineering and risk management, to be used in practice?
SQ 2.3	What is the gap between the state-of-the-art and current practices used for identifying risk and specifying security requirements?

The findings of our survey (cf. Chapter 3) provide substantial material to answer the second research question and its sub-questions. Regarding the first sub-question, the initiation of the security elaboration process, according to the experiences of our responders, is equally distributed between the requirements, design and implementation phases of the development lifecycle with a slight preference towards the later stages (implementation phase). Similarly, the initiation of the risk management process takes place more frequently at the second stage of the development lifecycle, during the design phase. Concerning the way the security requirements and risks are elaborated, practitioners indicated that systematic and structured approaches are only "*sometimes*" used instead of ad-hoc approaches according to the individual project's needs. Automated support tools were also indicated as "*rarely*" used in the process. Finally, when asked about the importance of individual roles during security elaboration, responders selected security experts as the most vital role to the process, but also recognized the importance of software architects, analysts, developers, stakeholders and end users.

For the second sub-question, a number of factors influencing the selection of the methods followed during security and risk elaboration was evaluated by the

responders. The situational needs of the implementation, similar past experiences and the suggestions of the stakeholders were rated as important influencing factors. The standardisability, scientific relevance and tool support of the selected method were also considered as contributing factors to the decision making process. Responders also indicated very positive opinions regarding the early elaboration of security from an organizational standpoint following a structured approach, instead of ad-hoc approaches focusing solely on the technical details of the implementation. The use of automated support tools and the integration of risk management elements in the security elaboration process were also perceived as beneficial.

These findings highlight significant gaps between the practical application of security requirements engineering and risk management and the literature findings of these areas. The first point of interest is the early adoption of the process in the development lifecycle. As previously discussed, literature highlights the need for security elaboration beginning from the requirements phase of the software development lifecycle. Practitioners responding to our survey, while recognizing the early adoption as highly beneficial, do not always practice it at the software projects they participate. In practice, as indicated by the survey's responses, security is often elaborated later in the lifecycle, during the design or even implementation phase. A similar phenomenon is observed for the adoption of systematic approaches, the benefits of which are presented in literature and recognized by the practitioners participating in our survey. Nevertheless, when it comes to the practical application systematic methods are not always favored over ad-hoc approaches. The use of automated support tools is also perceived as beneficial but the frequency of their usage in practice was reported as limited. All the above findings were taken into account during the development of our method and significantly influenced our decisions during the process.

RQ3	*How can we devise an effective method that integrates risk management in the security requirements engineering process?*
SQ 3.1	How can the findings of RQ1 and RQ2 contribute to the development of the new method?
SQ 3.2	Is this method effective and applicable enough to assist the security requirements definition and risk analysis process in real life settings? What are the limitations?

The development of our method was influenced by the literature findings (RQ1) and the analysis of the state-of-the-practice via our online questionnaire (RQ2). From the identified methods for security requirements engineering the STS method was selected as the main source of method fragments for the development of our new method. This selection was based on the adherence of the STS method to the critical

factors identified from the literature of the area, further elaborated at Section 2.4. Since the focus of our research is on the security of socio-technical systems, the selected method should also be able to model the complexity of the interactions inherent to such systems, produce comprehensive models of the designed system and be supported by automated tools. The goal-oriented approach followed by the STS method, supported by the STS modeling language (STS-ml) and the STS tool for modeling and automated analysis, made this method a perfect fit, since these features are aligned both with the identified practical needs and the context of our research.

The identified and overviewed risk management methods (cf. Section 2.3) were used as a source for the identification of the risk management elements integrated in our method. More specifically techniques and concepts used for risk analysis and in particular for threat identification, impact calculation and risk evaluation, included in the identified methods were examined. As already discussed in Section 2.4, the ISRA-BM method was selected as a result of the contextual similarities between its techniques and concepts and the needs of our new method. Examples of such techniques and concepts included hierarchical prioritization of assets, risk prioritization and asset-dependency diagrams, which were then adapted to improve or add new aspects to the risk analysis process already in place in the STS method. Additionally, factors such as the quantitative results provided by ISRA-BM, based on a well-established framework of mathematical formulae applied by security experts, added to the methods credibility and were in alignment with the needs of practitioners.

As a result of the findings of RQ1 and RQ2 the development of our method was based on solid principles, supported by both the scientific literature of the area and the consensus of practitioners. Consequently, the developed method, presented and overviewed at Chapter 4, approaches the issues of security and risk from the early stages of the system's development, providing the concepts and rules necessary for the analysis of the social aspects of the system from an organizational perspective. The design of the system is influenced by the outcome of the risk analysis process, which is used as an input during the selection of the security needs and authorizations that need to be imposed on the system to protect the informational assets of its stakeholders. After the iterative process of system modeling, threat assessment and user security requirement elaboration, the produced system model can be automatically checked and the security requirements described in it can be explicitly listed and used further into the system's development process.

From the application of our method in practical settings (cf. Chapter 5), conclusions were drawn regarding its effectiveness and shortcomings. As already discussed (cf. Section 5.7), the newly developed method was able to be implemented with minimal adjustments to its rules and processes, successfully expressed the security needs of

the system's stakeholders and revealed previously unidentified security aspects of the system via the produced security requirements. One important contribution of our method was the threat assessment phase, resulting in a list of prioritized threats which were then used as the basis from which the security needs of the users originated. Therefore our method facilitates the early elaboration of risk during the initial phases of the system's development, along with the identification of security requirements. Nevertheless, the lack of elaboration of countermeasures to defend against the identified threats is a shortcoming of our method and could be the subject of future works.

Another shortcoming of our case study was the limited number of threats that were identified, making the effects of the threat prioritization process less apparent overall. Nevertheless we believe that the contribution of threat prioritization to the identification of security needs and the tracing of security requirements to specific threats will become highly noticeable in future applications of the method (cf. Section6.3), involving larger and more complex systems, with numerous potential threats identified.

6.2 Limitations

Previously identified limitations for the different parts of this study (c.f. Sections 3.3, 4.3 and 5.7), need to be addressed. The first point of discussion focuses on the limitations of the survey of practice undertaken as part of our research. As discussed at Section 3.3, the population of our survey responders was limited to twenty four (24) subjects. The composition of this group also limited the analysis options, as the majority of responders had a research background compared to a few practitioners, thus not allowing for meaningful groupings for cross-group analyses of the responses. As a result, while the survey provided some useful insights about security and risk in practice, the recorded responses might be skewed towards opinions more favorable to researchers than practitioners of the field.

The developed method also presents some weaknesses, thoroughly discussed in Section 4.3. To summarize, the most important shortcomings have to do with the method's focus on informational assets of the system, therefore disregarding potential threats to other asset groups, as vulnerable and critical for the system's functionality as information. Another limitation lays on the reliability of integral parts of the method on subjective human evaluations, which are prone to bias and errors in their judgment. The input of stakeholders and system users plays an integral part in the creation of the system's model, the asset evaluation process and

the expression of user security needs. Therefore the quality of the method's output (e.g., system model and security requirements) can be affected by the subjectivity of the input provided by human stakeholders and users. Nonetheless, by using large and diverse groups of human stakeholders, discrepancies in their input would have a smaller effect on average.

The shortcomings of the method's practical application, addressed at Section 5.7, mainly had to do with the granularity and completeness of the analysis of the studied system. Our case study focused on specific functionalities offered by the system; a sub-set of its overall capabilities. Therefore the security requirements identified by our method concern only a part of the overall system's security. Similarly, a more experienced security engineer could provide a more complete list of potential threats towards the studied system. Since in our case, a smaller number of threats was identified, the potential benefits from their prioritization were less apparent in the definition of the user security needs and their implementation through the produced security requirements. Finally, another point worth addressing is the validity of the evaluation of the method's performance in practice, which took place at the end of our case study. Since the authors and creators of this method also played the role of the analyst, they were the ones that evaluated its usability, amongst other evaluation criteria, therefore some validity issues may arise regarding the objectiveness of this evaluation.

6.3 Future research

In this work we developed a structured method to integrate elements of risk management in the security requirements engineering process. By combining method fragments from established methods in the field of security requirements and risk management we developed a new method that uses the result of the risk evaluation and prioritization as input for the identification of user's security needs, creating a complete socio-technical model of the system to-be. We applied this method in practice via a retrospective case study, in order to evaluate its completeness and performance. Throughout our research, in our attempt to provide answers to our research question, a number of directions for future research in the topic were identified and will be shortly discussed here.

Since the evaluation of our newly created method was performed by its one application in the practical settings of the STRIPA system's extended environment, a number of additional case studies should be performed in the future, ideally in the settings of the software industry, in order to be able to extract more conclusions

about the performance of our method. By applying our method in more diverse and complex socio-technical environments there is a better chance of identifying a number of shortcomings, potential improvements and further expansions of the method's process and functionalities. Improvements in the threat impact identification technique could, for example, be identified by applying the method in complex systems with multi-level goal trees, including OR goal-decompositions; a current limitation of our method as discussed in section 4.3.

Another point of interest could be the expansion of our methods concepts and techniques in order to be able to accommodate a broader range of assets. Other than information assets, currently supported from our method, more asset groups can be introduced which could be interconnected, evaluated by the users and threatened by harmful events, putting the overall system at risk. Such asset groups can include infrastructure (e.g., hardware) or personnel (e.g., users of the system) with appropriate threats modeled for each of them.

An additional direction that could be followed by future research attempts on this topic could examine the expansion of our method's functionalities. For instance, the prioritization process could be expanded to prioritize, not only the potential threats to the system, but also the produced security requirements, which could be ranked according to the severity of the threat they are attempting to counter. Additional elements of risk management could also be introduced in order to include the provision of countermeasures for the threats identified. The integration of risk in the security requirements engineering process could also be explored using different methods as a baseline.

Finally, it would greatly improve the applicability of our method, if the STS-Tool, used for creating the system model and automatically checking it, was extended to cover the additional techniques (e.g., importance values assignments, threat impact value calculation), introduced by our method. This way the process of calculating and inserting relative importance values to the model and through them calculating the impact of threats could be automated and thus require less resources in order to be completed.

References

Alberts, C., Dorofee, A., Stevens, J., & Woody, C. (2003). *Introduction to the OCTAVE Approach*. Pittsburgh, PA: Carnegie Mellon University.

Basin, D., Clavel, M., & Egea, M. (2011). A decade of model-driven security. In *Proceedings of the 16th ACM symposium on Access control models and technologies* (pp. 1-10).

Basin, D., Doser, J., & Lodderstedt, T. (2006). Model driven security: From UML models to access control infrastructures. *ACM Transactions on Software Engineering and Methodology* (TOSEM), 15(1), 39-91.

Baskerville, R. (1992). The development duality of information systems security, *Journal of Management Systems* 4 (1) 1–12.

Best, B., Jurjens, J., & Nuseibeh, B. (2007). Model-based security engineering of distributed information systems using UMLsec. In *Proceedings of 29th International Conference on Software Engineering* (pp. 581-590).

Brinkkemper, S. (1996). Method engineering : engineering of information methods and tools. *Information and Software Technology*, 38 (4), 275-280.

Brinkkemper, S., Saeki, M., & Harmsen, F. (1999). Meta-modelling based assembly techniques for situational method engineering. *Information Systems*, 24(3), 209-228.

Chung, L. & B.A. Nixon. (1995). Dealing with non-functional requirements: three experimental studies of a process-oriented approach. In *Proceedings of the 17th international conference on Software engineering* (pp. 25-37).

Dalpiaz, F., Giorgini, P., & Mylopoulos, J. (2013). Adaptive socio-technical systems: a requirements-based approach. *Requirements engineering*, 18(1), 1-24.

Dalpiaz, F., Paja, E., & Giorgini, P. (2011). Security requirements engineering via commitments. In *Proceedings of 1st Workshop on Socio-Technical Aspects in Security and Trust (STAST),* (pp. 1-8).

Darke, P., Shanks, G., & Broadbent, M. (1998). Successfully completing case study research: combining rigour, relevance and pragmatism. *Information Systems Journal*, 8 (4), 273-289.

Dubois, É., Heymans, P., Mayer, N., & Matulevičius, R. (2010). A systematic approach to define the domain of information system security risk management. In S. Nurcan, C. Salinesi, C. Souveyet & C. Ralyte (eds.) *Intentional Perspectives on Information Systems Engineering* (pp. 289-306). Berlin- Heidelberg: Springer.

Elahi, G., & Yu, E. (2007). A goal oriented approach for modeling and analyzing security trade-offs. In C. Parent, K.D. Schewe, V.C. Storey & B. Thalheim (eds.) *Conceptual Modeling-ER 2007* (pp. 375-390). Berlin Heidelberg: Springer.

Eric, S. K., Giorgini, P., & Maiden, N. (2011). *Social modeling for requirements engineering*. Cambridge, MA: Mit Press.

Fabian, B., Gürses, S., Heisel, M., Santen, T., & Schmidt, H. (2010). A comparison of security requirements engineering methods. *Requirements engineering*, 15(1), 7-40.

Giorgini, P., Massacci, F., Mylopoulos, J., & Zannone, N. (2005). Modeling security requirements through ownership, permission and delegation. In *Proceedings of the 13th IEEE International Conference on Requirements Engineering,* (pp. 167-176).

Giunchiglia, F., Mylopoulos, J., & Perini, A. (2003). The tropos software development methodology: processes, models and diagrams. In F. Giunchiglia, J. Odell & G. Weiss (eds.) *Agent-Oriented Software Engineering III* (pp. 162-173). Berlin Heidelberg: Springer.

Greenhalgh, T., & Peacock, R. (2005). Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources. *Bmj*, 331(7524), 1064-1065.

Haley, C. B., Laney, R., Moffett, J. D., & Nuseibeh, B. (2008). Security requirements engineering: A framework for representation and analysis. *IEEE Transactions on Software Engineering*, *34*(1), 133-153.

ISO/IEC. (2004). 13335-1 - Information technology – security techniques – management of information and communications technology security. *International Organization for Standardization*, Geneva

ISO/IEC. (2008). 27799: Health informatics. Information security management in health using ISO/IEC, 27002. *International Organization for Standardization*, Geneva

Jürjens, J. (2002). UMLsec: Extending UML for secure systems development. In J.M. Jezequel, H. Hussmann & S. Cook (eds.) *UML 2002— The Unified Modeling Language* (pp. 412-425). Berlin Heidelberg: Springer.

Karabacak, B., & Sogukpinar, I. (2005). ISRAM: information security risk analysis method. *Computers & Security*, *24*(2), 147-159.

Kim, J., Kim, M., Park, S. (2005). Goal and scenario based domain requirements analysis environment, *The Journal of Systems and Software*, *79*, 926–938.

Liu, L., Yu, E., & Mylopoulos, J. (2003). Security and privacy requirements analysis within a social setting. In *Proceedings of 11th IEEE International Requirements Engineering Conference* (pp. 151-161).

Maglogiannis, I., & Zafiropoulos, E. (2006, August). Modeling risk in distributed healthcare information systems. In *Proceedings of the IEEE Engineering in Medicine and Biology Society*, EMBS'06. (pp. 5447-5450).

McDermott, J., & Fox, C. (1999). Using abuse case models for security requirements analysis. In *Proceedings of the 15th Annual Computer Security Applications Conference, (ACSAC'99),* (pp. 55-64).

Mellado, D., Blanco, C., Sánchez, L. E., & Fernández-Medina, E. (2010). A systematic review of security requirements engineering. *Computer Standards & Interfaces*, *32*(4), 153-165.

Mellado, D., Fernández-Medina, E., & Piattini, M. (2007). A common criteria based security requirements engineering process for the development of secure information systems. *Computer standards & interfaces*, *29*(2), 244-253.

Meulendijk, M.C., Spruit, M.R., Jansen, P.A.F., Numans, M.E., & Brinkkemper, S. (2014). STRIPA: A rule-based decision support system for medication reviews in primary care. *Manuscript submitted for publication*.

Mouratidis, H., Giorgini, P., & Manson, G. (2003). Modelling secure multiagent systems. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems* (pp. 859-866).

Mouratidis, H., & Giorgini, P. (2007). Secure tropos: a security-oriented extension of the tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, *17*(02), 285-309.

Mouratidis, H. (2011). Secure Software Systems Engineering: The Secure Tropos Approach. *Journal of Software (1796217X)*, *6*(3).

O'Brien, J. A. (2002). *Introduction to information systems: Essentials for the e-business enterprise*. New York, NY: McGraw-Hill, Inc.

Paja, E., Dalpiaz, F., Poggianella, M., Roberti, P., & Giorgini, P. (2012). Modelling Security Requirements in Socio-Technical Systems with STS-Tool. In *Proceedings of CAiSE Forum,* (pp. 155-162).

Paja, E., Dalpiaz, F., & Giorgini, P. (2012). Identifying conflicts in security requirements with STS-ml. Trento : Università degli Studi di Trento.

Paja, E., Dalpiaz, F., & Giorgini, P. (2013a). Managing security requirements conflicts in socio-technical systems. In W. Ng, V.C. Storey, & J. Trujillo (eds.), *Conceptual Modeling* (pp. 270-283). Berlin, Heidelberg: Springer.

Paja, E., Dalpiaz, F., & Giorgini, P. (2013b). Designing Secure Socio-Technical Systems with STS-ml. In *Proceedings of the 6th International i* Workshop (iStar 2013), CEUR Vol-978.*

Pavlidis, M., & Islam, S. (2011). SecTro: A CASE Tool for Modelling Security in Requirements Engineering using Secure Tropos. In *Proceedings of CAiSE Forum,* (pp. 89-96).

Peltier, T. R. (2005). *Information security risk analysis*. Boca Raton, FL: CRC press.

Rindfleisch, T. C. (1997). Privacy, information technology, and health care. *Communications of the ACM*, 40(8), 92-100.

Rumbaugh, J., Jacobson, I., & Booch, G. (2004). *The Unified Modeling Language Reference Manual,* Upper Saddle River, NJ: Pearson Higher Education.

Samy, G. N., Ahmad, R., & Ismail, Z. (2010). Security threats categories in healthcare information systems. *Health Informatics Journal*, 16(3), 201-209.

Sindre, G., & Opdahl, A. L. (2005). Eliciting security requirements with misuse cases. *Requirements Engineering*, *10*(1), 34-44.

Stølen, K., den Braber, F., Dimitrakos, T., Fredriksen, R., Gran, B. A., Houmb, S. H. & Aagedal, J. Ø. (2002). Model-based risk assessment–the CORAS approach. In *Proceedings of 1st iTrust Workshop*.

Stoneburner, G., Goguen, A., & Feringa, A. (2002). Risk management guide for information technology systems. *Nist special publication*, *800*(30), 800-30.

Streeton, R., Cooke, M., & Campbell, J. (2004). Researching the researchers: Using a snowballing technique, *Nurse Researcher*, 12(1), 35-46.

Suh, B., & Han, I. (2003). The IS risk analysis based on a business model. *Information & Management*, 41(2), 149-158.

Vaast, E. (2007). Danger is in the eye of the beholders: Social representations of Information Systems security in healthcare. *The Journal of Strategic Information Systems*, 16(2), 130-152.

van de Weerd, I., & Brinkkemper, S. (2008). Meta-modeling for situational analysis and design methods. In I. van de weerd, & s. Brinkkemper (eds.), *Handbook ot research on modern systems analysis and design technologies and applications* (pp. 35-54). Mankato, Minnesota State University.

van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *Proceedings of Fifth IEEE International Symposium on Requirements Engineering* (pp. 249-262).

van Lamsweerde, A. (2004). Elaborating security requirements by construction of intentional anti-models. In *Proceedings of the 26th International Conference on Software Engineering* (pp. 148-157).

Verdon, D., & McGraw, G. (2004). Risk analysis in software design. *Security & Privacy, IEEE*, *2*(4), 79-84.

Vorster, A., & Labuschagne, L. (2005). A framework for comparing different information security risk analysis methodologies. In *Proceedings of the 2005 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries* (pp. 95-103).

Whitman, M. E. (2003). Enemy at the gate: threats to information security. *Communications of the ACM*, *46*(8), 91-95.

Yazar, Z. (2002). A qualitative risk analysis and management tool–CRAMM.*SANS InfoSec Reading Room White Paper*.

Youseef, A., & Liu, F. (2012). A New Framework to Model a Secure E-Commerce System. *World Academy of Science, Engineering and Technology 6*(2), 6-12.

Yu, E., & Mylopoulos, J. (1998). Why goal-oriented requirements engineering. In *Proceedings of the 4th International Workshop on Requirements Engineering: Foundations of Software Quality* (pp. 15-22).

Zave, P. (1997). Classification of research efforts in requirements engineering. *ACM Computing Surveys (CSUR)*, *29*(4), 315-321.

Appendix

A1: Survey form

Engineering Secure Software / Information Systems

In the Department of Information and Computing Sciences at Utrecht University (the Netherlands), research is being conducted, concerning the development of secure software and information systems.

This survey aims to gather insights regarding how security is considered and analyzed throughout the software development process. We are interested in inputs from practitioners, and also knowing the perception of researchers. We hope that these insights will help us determine the needs of the industry and will ultimately support the creation of more secure systems.

Answering the survey will take approximately 10 minutes of your time. The survey is anonymous and all answers will remain confidential throughout the process of their collection and analysis. In the final page of the survey you can optionally provide your contact information, if you wish to receive the results of this survey once our analysis is completed.

For further information, remarks or suggestions please contact us. Thank you in advance for your participation in our study!

Nikos Argyropoulos (n.argyropoulos@students.uu.nl)
Dr. Fabiano Dalpiaz (f.dalpiaz@uu.nl)

Utrecht University, 2014

* Required

Background Information

What is your age group? *

In which region do you mainly conduct your job / your research? *

Which of the following characterizes best your background in the field of software/information systems (SW / IS) engineering?

- ○ Researcher
- ○ Practitioner
- ○ Student
- ○ Other:

How many years of experience do you have in the field of SW / IS engineering? *

How many years of experience do you have in the field of SW / IS security? *

How recently were you last involved in a SW / IS development project? *

Security during the system development life cycle

Please answer to the following questions. Whenever you don't know how to answer, due to lack of expertise or knowledge, either do not answer or choose the "Don't know" option

In your experience, how often do you consider the social aspects of the system, when discussing and analyzing the security of new SW / IS?

e.g., relationships between users, organizational structure, etc.

	1	2	3	4	5	
Never	○	○	○	○	○	Always

In your experience, how often do you consider other technical systems already in place, when discussing and analyzing the security of new SW / IS?

	1	2	3	4	5	
Never	○	○	○	○	○	Always

How beneficial would you consider analyzing system security from an organizational, instead of a purely technical, perspective?

	1	2	3	4	5	
Very harmful	○	○	○	○	○	Very beneficial

In your experience, how often do you discuss and analyze security at each phase of the development life cycle? *

	Never	Rarely	Sometimes	Often	Always	Don't know
Requirements Phase	○	○	○	○	○	○
Design Phase	○	○	○	○	○	○
Implementation Phase	○	○	○	○	○	○
Testing Phase	○	○	○	○	○	○
Deployment Phase	○	○	○	○	○	○
Maintenance Phase	○	○	○	○	○	○

How beneficial would you consider the adoption of security at the requirements definition phase rather than later in the development life cycle?

	1	2	3	4	5	
Very harmful	○	○	○	○	○	Very beneficial

Who is involved, what are the influences?

How important do you consider each of the following roles, regarding their contribution to the elaboration of security? *

	Not important at all	Not very important	Neutral	Very Important	Essential	Don't know
Software Architect	○	○	○	○	○	○
Requirements Analyst	○	○	○	○	○	○
Security Expert	○	○	○	○	○	○
Software Developer	○	○	○	○	○	○
Management / Stakeholders	○	○	○	○	○	○
End Users	○	○	○	○	○	○

How important do you consider the following factors when deciding on how to approach the security of the new system? *

	Not Important at all	Not very important	Neutral	Very important	Essential	Don't know
Similar past experiences	○	○	○	○	○	○
Popularity/Reputation of approach among professionals	○	○	○	○	○	○
Standardisability	○	○	○	○	○	○
Situational needs of the implementation	○	○	○	○	○	○
Stakeholders suggestions	○	○	○	○	○	○
Scientific/Academic relevance of the approach	○	○	○	○	○	○
Tool support availability	○	○	○	○	○	○

Systematic approaches and tools for security

How often do you follow a systematic approach/method when elaborating on security requirements?

	1	2	3	4	5	
Never	○	○	○	○	○	Always

How beneficial do you consider a systematic approach towards security requirements rather than ad-hoc approaches?

Note: You can skip this one if you answered "Never" in the previous question.

	1	2	3	4	5	
Very harmful	○	○	○	○	○	Very beneficial

How often do you use automated tools when elaborating on security requirements?

	1	2	3	4	5	
Never	○	○	○	○	○	Always

How beneficial do you consider the use of automated support tools regarding the quality of the obtained security requirements?

Note: You can skip this one if you answered "Never" in the previous question.

	1	2	3	4	5	
Very harmful	○	○	○	○	○	Very beneficial

Name any method for security requirements that you have used in past projects.

Note: Optional question

Assets and Risk

In your experience, how often do you analyze risk at each phase of the development lifecycle? *

	Never	Rarely	Sometimes	Often	Always	Don't know
Requirements Phase	○	○	○	○	○	○
Design Phase	○	○	○	○	○	○
Implementation Phase	○	○	○	○	○	○
Testing Phase	○	○	○	○	○	○
Deployment Phase	○	○	○	○	○	○
Maintenance Phase	○	○	○	○	○	○

How important for the overall security of the system do you consider each of the following assets? *

	Not important at all	Not very important	Neutral	Very important	Essential	Don't know
Data / Information	○	○	○	○	○	○
Applications / Software	○	○	○	○	○	○
Infrastructure / Physical assets	○	○	○	○	○	○
Organizational assets / Personnel	○	○	○	○	○	○

Do you integrate risk analysis methods with the security requirements elaboration process?

	1	2	3	4	5	
Never	○	○	○	○	○	Always

How often do you use systematic approaches (established methods) to perform risk analysis?

	1	2	3	4	5	
Never	○	○	○	○	○	Always

Do you perform risk assessment using qualitative or quantitative methods?

Note: You can skip this one if you answered "Never" in the previous question. - e.g., Qualitative result: "low, medium, high" - Quantitative result: 0.98, 0.66

	1	2	3	4	5	
Only qualitative	○	○	○	○	○	Only quantitative

Name any risk management method(s) that you have used in past projects.

Note: Optional question

Follow up and results

If you want to receive the results of this survey, once our research is completed, please enter you email address in the following field:

Submit

A2: Overview of method activities origin and modifications

Activity	Origin	Modifications / Comments
Identify Stakeholders	STS-ml	*None*
Identify Assets and Interactions	STS-ml	*None*
Evaluate importance of goals	ISRA-BM	Changed to accommodate *goals* instead of business functions.
Identify Information /Ownership	STS-ml	*None*
Structure Information	STS-ml	*None*
Determine Informational Asset Importance	ISRA-BM	Limited to *informational assets only*
Identify Threats	*new*	Can be performed in different ways (e.g., ISO standards, past experience, etc.)
Model Threats	STS-ml	*None*
Assess Threat Impact	*new*	Through the process introduced by our method (cf. Section 4.2).
Assess Threat Probability	*new*	Can be evaluated in different ways (e.g., ISO standards, system environment, etc.)
Evaluate Risk	ISRA-BM	Only *part of formulas* for risk calculation used.
Express Security Needs	STS-ml	Moved *after* the identification and prioritization of threats.
Model Authorizations	STS-ml	Moved *after* the identification and prioritization of threats.
Perform Consistency Analysis	STS-ml	*None*
Perform Security Analysis	STS-ml	*None*
Generate Security Requirements	STS-ml	*None*

Table A.1: Origin and modifications of method fragments

A.3: Template of user evaluation input form

Part 1: Goal evaluation

Instructions: Rank the activities of each of group below, according to their *relative importance* to each other, by dividing 100 point amongst them. (More points = higher importance)
Hint : If you consider all activities of a group equally important, divide the 100 points equally amongst them.

Groups / Activities	Activity Description	Importance
Group 1: *Receive Heathcare Service*	***Points left:***	*100*
GP consultation	*Visiting a caregiver for medical consultation*	
Receive medication	*Buy/obtain the prescribed medication*	
	Total:	**0**
Group 2: *Visiting a caregiver for medical consultation*	***Points left:***	*100*
Report side effects	*Report observed side effects to physician*	
Report drug adherence	*Report adherence to prescribed drugs to physician*	
Receive prescription for meds	*Receive adequate prescription from physician*	
	Total:	0
Group 3: *Medication prescription process*	***Points left:***	*100*
Anamnesis	*Patient records consultation to determine medical history*	
Analysis	*Analysis of current condition & possible treatment options*	
Treatment Plan	*Selection of treatment from available options*	
Patient Discussion	*Discussion of the treatment plan with the patient*	
	Total:	0
Group 4: *Analysis of current condition & possible treatment options*	***Points left:***	*100*
Report symptoms	*Input of patient's symptoms*	
Input existing medication	*Input of patient's current medication*	
Completeness check	*Check for untreated or overtreated conditions*	
Determine medication	*Selection of new medication*	
Determine interactions & dosage	*Check for unwanted interactions and correct dosage*	
	Total:	0
Group 5: *Decision support through a software*	***Points left:***	*100*
Completeness check	*Check for untreated or overtreated conditions*	
Interactions & dosage check	*Check for unwanted interactions and correct dosage*	
	Total:	0
Group 6: *Check for under- or over-treatment and suggest appropriate medication*	***Points left:***	*100*
Check for overtreatment	*Determine if too much medication prescribed*	
Check for undertreatment	*Determine if too little medication prescribed*	
Advise medication	*Suggest medication adjustments or new medication*	
	Total:	0
Group 7: *Check for unwanted interactions and correct dosage*	***Points left:***	*100*
Check for interactions	*Check for unwanted interactions between meds*	
Suggest dosage	*Suggest appropriate dosage for each medication*	
	Total:	0

Figure A.1: Goal evaluation, user input template

Part 2: Asset evaluation

Instructions: Evaluate how *important* you consider each of the following pieces of information for the overall success of the process of the diagnosis and treatment selection.
Select the appropriate rating from 1 to 5, for each information asset presented below.

Informational Asset	Information Description	Importance
Patient Name	*The name of the patient*	
Patient Address	*Patient's geographical location*	
Date of Birth	*Date of birth of the patient*	
Episodes	*Medical history of patient*	
Current medication	*Medication the patient is currently receiving*	
Diagnostics	*Lab results, X-rays, etc.*	
Prescribed Medication	*Medication prescribed to the patient for current condition*	
Dosage	*Dosage of prescribed medication*	
Medication Info	*Commercial name, active ingredients, etc.*	
Suggested medication	*Medication suggested for a specific condition*	
Rationale	*Rationale behind medication selection*	
Suggested dosage	*Dosage suggestion for medication*	
Causes	*Causes of unwanted interactions between suggested medication*	

Figure A.2: Information assets evaluation, user input template

A.4: User evaluations of relative goal importance

Average values

Groups / Activities	Activity Description		Importance
Group 1: *Receive Heathcare Service*		***Points left:***	*0*
GP consultation	*Visiting a caregiver for medical consultation*		67,14
Receive medication	*Buy/obtain the prescribed medication*		32,86
		Total:	100
Group 2: *Visiting a caregiver for medical consultation*		***Points left:***	*0*
Report side effects	*Report observed side effects to physician*		25,00
Report drug adherence	*Report adherence to prescribed drugs to physician*		28,89
Receive prescription for meds	*Receive adequate prescription from physician*		46,11
		Total:	100
Group 3: *Medication prescription process*		***Points left:***	*0*
Anamnesis	*Patient records consultation to determine medical history*		22,78
Analysis	*Analysis of current condition & possible treatment options*		30,56
Treatment Plan	*Selection of treatment from available options*		26,11
Patient Discussion	*Discussion of the treatment plan with the patient*		20,56
		Total:	100
Group 4: *Analysis of current condition & possible treatment options*		***Points left:***	*0*
Report symptoms	*Input of patient's symptoms*		22,11
Input existing medication	*Input of patient's current medication*		19,89
Completeness check	*Check for untreated or overtreated conditions*		20,44
Determine medication	*Selection of new medication*		19,33
Determine interactions & dosage	*Check for unwanted interactions and correct dosage*		18,22
		Total:	100
Group 5: *Decision support through a software*		***Points left:***	*0*
Completeness check	*Check for untreated or overtreated conditions*		53,33
Interactions & dosage check	*Check for unwanted interactions and correct dosage*		46,67
		Total:	100
Group 6: *Check for under- or over-treatment and suggest appropriate medication*		***Points left:***	*0*
Check for overtreatment	*Determine if too much medication prescribed*		34,89
Check for undertreatment	*Determine if too little medication prescribed*		32,56
Advise medication	*Suggest medication adjustments or new medication*		32,56
		Total:	100
Group 7: *Check for unwanted interactions and correct dosage*		***Points left:***	*0*
Check for interactions	*Check for unwanted interactions between meds*		58,89
Suggest dosage	*Suggest appropriate dosage for each medication*		41,11
		Total:	100

Figure A.3: Average goal evaluation values

User Input

Importance	Importance	Importance	Importance	Importance	Importance	Importance	Importance	Importance
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
60	60	60	60	60	60	60	80	90
40	40	40	40	40	40	40	20	10
100	100	100	100	100	100	100	**100**	**100**
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
20	30	10	30	30	25	25	40	15
30	30	40	30	35	25	25	20	25
50	40	50	40	35	50	50	40	60
100	100	100	100	100	100	100	100	100
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
30	25	30	15	30	15	20	20	20
30	25	30	25	40	35	20	30	40
20	25	30	25	15	30	40	30	20
20	25	10	35	15	20	20	20	20
100	100	100	100	100	100	100	100	100
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
20	40	20	24	20	20	20	15	20
20	15	20	24	20	20	20	20	20
20	15	20	19	20	25	20	25	20
20	15	20	19	25	20	20	15	20
20	15	20	14	15	15	20	25	20
100	100	100	100	100	100	100	100	100
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
50	50	50	60	60	60	50	50	50
50	50	50	40	40	40	50	50	50
100	100	100	100	100	100	100	100	100
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
34	40	35	30	40	30	30	40	35
33	40	25	30	30	30	30	40	35
33	20	40	40	30	40	40	20	30
100	100	100	100	100	100	100	100	100
0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*
60	50	50	50	60	50	60	70	80
40	50	50	50	40	50	40	30	20
100	100	100	100	100	100	100	100	100

Figure A.4: Overall goal evaluation values

A.5: User evaluations of relative information importance

Average values

Informational Asset	Information Description	Importance
Patient Name	*The name of the patient*	2,33
Patient Address	*Patient's geographical location*	2,22
Date of Birth	*Date of birth of the patient*	4,11
Episodes	*Medical history of patient*	4,78
Current medication	*Medication the patient is currently receiving*	4,56
Diagnostics	*Lab results, X-rays, etc.*	4,33
Prescribed Medication	*Medication prescribed to the patient for current condition*	4,67
Dosage	*Dosage of prescribed medication*	4,33
Medication Info	*Commercial name, active ingredients, etc.*	3,22
Suggested medication	*Medication suggested for a specific condition*	3,78
Rationale	*Rationale behind medication selection*	3,67
Suggested dosage	*Dosage suggestion for medication*	3,56
Causes	*Causes of unwanted interactions between suggested medication*	3,11

Figure A.5: Average information assets evaluation values

User Input

Importance	Importance	Necessity	Importance	Importance	Importance	Importance	Importance	Importance
1	5	2	3	1	2	1	4	2
2	2	3	2	2	2	2	2	3
4	5	4	4	5	4	3	4	4
5	5	4	5	5	5	4	5	5
4	5	5	5	5	4	4	5	4
5	4	3	5	4	5	4	5	4
4	5	5	5	5	4	5	5	4
3	4	5	4	5	4	5	5	4
4	3	3	3	3	3	3	3	4
4	3	5	3	4	3	4	4	4
5	3	4	3	4	2	3	4	5
3	3	4	3	4	3	4	4	4
2	3	2	3	4	2	2	5	5

Figure A.6: Overall information assets evaluation values

A.6: Security Requirements List

Responsible	Description
Caregiver	STRIPA requires Caregiver to assure the availability of document *Medication Advice*.
	STRIPA requires Caregiver to assure the availability of document *Treatment Suggestion*.
	Patient requires Caregiver to ensure integrity of transmission over the provision of document *Prescription*, when Caregiver provides *Prescription* to Patient.
	Patient requires Caregiver to ensure confidentiality of transmission over the provision of document *Prescription*, when Caregiver provides *Prescription* to Patient.
	STRIPA requires Caregiver to ensure integrity of transmission over the provision of document *Medication List*, when Caregiver provides *Medication List* to STRIPA.
	STRIPA requires Caregiver to ensure integrity of transmission over the provision of document *Patient Records*, when Caregiver provides *Patient Records* to STRIPA.
	STRIPA requires Caregiver to ensure confidentiality of transmission over the provision of document *Patient Records*, when Caregiver provides *Patient Records* to STRIPA.
	STRIPA requires Caregiver need-to-know of Information *Causes , Medication Info , Rationale, Suggested Dosage* and *Suggested Medication*, in the scope of goal Medication Prescription.
	Patient requires Caregiver non-modification of Information *Current Medication , Episodes, Diagnostics* and *Patient Info*.
	Patient requires Caregiver need-to-know of Information *Current Medication, Episodes, Diagnostics* and *Patient Info*, in the scope of goal Medication Prescription.
	Patient requires Caregiver need-to-know of Information *Prescribed Meds* and *Dosage*, in the scope of goal Medication Prescription.
Patient	Patient will delegate to Caregiver that has a high trustworthiness level.
STRIPA	Caregiver requires no-delegation for goal *Completeness Check*, when delegating *Completeness Check* to STRIPA.
	Caregiver requires non repudiation-of-acceptance for goal *Completeness Check*, when delegating *Completeness Check* to STRIPA.
	Caregiver requires no-delegation for goal *Side effects & dosage check*, when delegating *Side-effects & dosage check* to STRIPA.
	Caregiver requires non repudiation-of-acceptance for goal *Side-effects & dosage* check, when delegating *Side effects & dosage check* to STRIPA.
	Caregiver requires STRIPA to assure the availability the document *Medication List*.

	Caregiver requires STRIPA to assure the availability the document *Patient Records.*
	Caregiver requires STRIPA to ensure integrity of transmission over the provision of document *Medication Advice*, when STRIPA provides *Medication Advice* to the Caregiver.
	Caregiver requires STRIPA to ensure integrity of transmission over the provision of document *Treatment Suggestion*, when STRIPA provides *Treatment Suggestion* to the Caregiver.
	Patient requires STRIPA non-modification of Information: *Diagnostics, Patient Info, Episodes, Dosage* and *Current Medication*.
	Patient requires STRIPA non-disclosure of Information: *Diagnostics, Patient Info, Episodes, Dosage* and *Current Medication*.
	Patient requires STRIPA need-to-know of Information: *Diagnostics, Patient Info, Episodes, Dosage* and *Current Medication*, in the scope of goal *Support decisions*.
	Patient requires STRIPA non-modification of Information: *Prescribed Meds*.
	Patient requires STRIPA need-to-know of Information: *Prescribed Meds*, in the scope of goal *Support decisions.*

Table A.2: Security requirements identified for the overall system