Noname manuscript No.
(will be inserted by the editor)

How to Integrate Legal Requirements into A Requirements Engineering Methodology for the Development of Security and Privacy Patterns

Luca Compagna · Paul El Khoury · Alžběta Krausová · Fabio Massacci · Nicola Zannone

Received: date / Accepted: date

Abstract Laws set requirements that force organizations to assess the security and privacy of their IT systems and impose them to implement minimal precautionary security measures. Several IT solutions (e.g., Privacy Enhancing Technologies, Access Control Infrastructure, etc.) have been proposed to address security and privacy issues. However, understanding why, and when such solutions have to be adopted is often unanswered because the answer comes only from a broader perspective, accounting for legal and organizational issues. Security engineers and legal experts should analyze the business goals of a company and its organizational structure and derive from there the points where security and privacy problems may arise and which solutions best fit such (legal) problems. The paper investigates the methodological support for capturing security and privacy requirements of a concrete health care provider.

Keywords Security & Privacy Patterns, Legal Requirements, Organization, Pattern Validation, Healthcare.

1 Introduction

Protecting sensitive information is a critical success factor for any organization. The "security" reputation of a company is now becoming a critical asset as more and more customers are considering security and privacy practices an important factor for decision making when selecting a service provider.

Market pressure is not the only force. Privacy is a highly regulated area in Europe. There are strict regulations in place within the European Union that impose rules for the collection, handling, and processing of personal data (e.g., the Directive on data protection, 95/46/EC, supplemented with the Directive on privacy and electronic communications, 2002/58/EC). Organizations that manage personal data cannot escape the obligation to implement the requirements established by such regulations in their IT infrastructure. Unfortunately, there is a gap between legal language and computer language. This gap represents severe problem as legal obligations have to be converted into requirements to be enforced by the IT infrastructure [?].

An answer to these problems can be provided using design patterns that integrate legal and IT concerns. Security patterns [?,?,?] have been proposed to capture expertise of security experts and make it available for application designers that might not have a solid security background. Yet, such proposals are often tainted by two limitations: the exclusive focus on the IT system and their practitioner informal outlook.

The first limitation makes it difficult to full capture legal requirements. Legal requirements are imposed on organizations as a whole and often refer to humans and human-run procedures (as opposed to computer-run procedures). For example, the analysis of security policies compliant with the Italian privacy legislation [?] showed that the largest share of the security and privacy policy is concerned with the behavior of human beings as opposed to IT systems. It is also well known that security vulnerabilities are often a direct consequence of mistakes at the interface between the IT system and the policies and procedures adopted by organizations [?]. Therefore, it is necessary to model and analyze IT systems together with their organizational context.

The second limitation affects the reliability of patterns. Though a pattern might be the result of the collaboration between security and legal experts, this human collaboration can be error-prone. Apparently right but indeed wrong patterns might erode their acceptance from application designers. One of the most effective countermeasures is represented by the application of formal method approaches as a way to establish proof-of-concept evidence of the soundness of the proposed solutions. This would not seem to be a major problem as the use of formal methods and logic to capture legal reasoning has a long tradition: the idea of legal reasoning as a "formalized process of deduction in which concrete cases may be included under legal rule descriptions mechanically" [?] originated in 1908. This theory by Roscoe Pound called mechanical jurisprudence "presupposes that it is possible to develop stable and unambiguous legal concepts" [?]. Logical relations in law were further investigated by many researchers. An interesting study was presented in 1913 by Hohfeld who introduced a system of basic legal elements in [?]. Later, formal methods based on deontic logic were proposed to provide a means for reasoning about ideal and actual behavior [?]. A preliminary formalization was defined by Mally in 1926 [?]; it has been further refined by von Wright in 1951 [?]. Those works opened a new research field at the crossroads of Artificial Intelligence and Law, and attracted the interest of several researchers (e.g., [?,?,?,?]). However, logic is very good to provide a precise model of a system but very bad at communicating such a model [?] to legal and system experts that may not have background in formal methods. As such it is not very effective in providing this additional assurance that legal and security experts long for.

All together, security and legal experts need support for the design of security patterns in the form of automated and easy-to-use tools that, on the one side, assist experts in the specification of their solutions and, on the other side, provide formal techniques for a proof-of-concept analysis of these solutions.

1.1 Contribution of the paper

In this paper, we show how we can combine logic, agent-oriented methodologies, practical security engineering with legal reasoning to capture, model, and reason about practical legal patterns in a concrete industrial health care domain. Ideally, the pattern design and validation process (Fig. 1) require legal experts to describe patterns in natural language. Such a description is parsed by a natural language processor on the

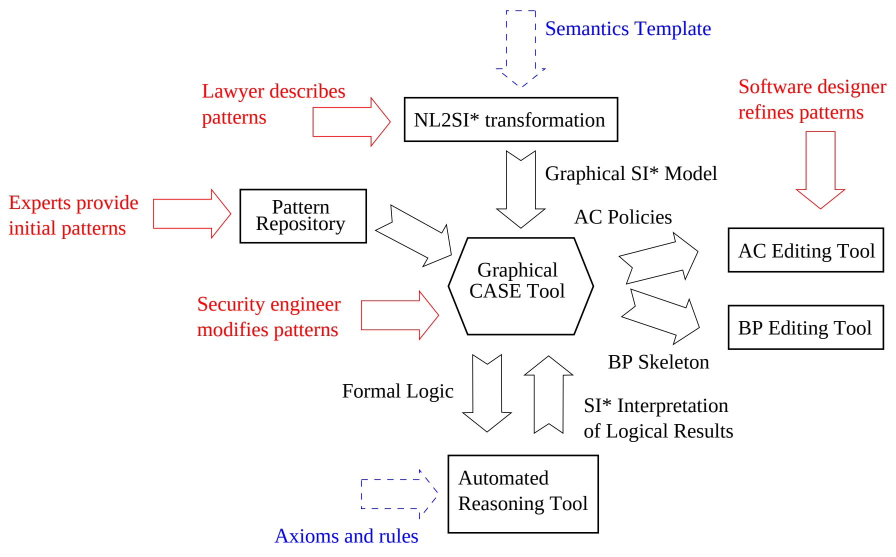


Fig. 1 Pattern Design and Validation

basis of a semantic template [**?**,**?**]. The annotated description is then used to automatically generate graphical models of patterns, which are revised by security engineers using a CASE Tool. Graphical patterns are translated into formal specifications that are verified using some automated reasoning tools. The results of the reasoning tool are interpreted by the CASE tool that graphically shows possible inconsistencies and errors in the patterns (if any). Security engineers together with legal experts revise the pattern in light of identified inconsistencies. Once the patterns are validated, the CASE tool could generate their implementation, for instance, in terms of access control (AC) policies or business processes (BP) [**?**,**?**,**?**]. Finally, the implementation of patterns is refined and adapted to a specific application domain by software designers.

The development of tools that support the process presented in Fig. 1 presents many challenges. In this paper, we propose a methodological approach that intends to support security and legal experts in the definition of security patterns. We start from a *graphical and easy-to-use modeling framework*, specifically the SI* modeling framework [**?**], to support system designers lacking deep security and legal knowledge to design and deploy systems where security and privacy solutions are enforced in accordance to regulations.

Next, we assess the suitability of this framework towards critical security and privacy issues, derived from legal regulations, that affect a health care system based on a smart items infrastructure. We show how to capture and generalize solutions to these issues into security and privacy patterns, and how to (proof-of-concept) validate them through an automated reasoning technique made available by the framework.

The security and privacy patterns described in this paper represent an excerpt of the pattern library we are populating in the context of the EU SERENITY project.[1] This pattern library is intended to serve as a reference in the design and deployment of systems sensitive to security, dependability, and privacy issues.

[1] EU-IST-IP 6th Framework Programme – SERENITY 27587 – http://www.serenity-project.org.

The remainder of the paper is structured as follows. In the next section (§2) we introduce the notion of security and privacy patterns. We then introduce a scenario that is used as a running example throughout the paper (§3). We discuss security and privacy requirements from a legal perspective (§4) and present a brief description of the language used to model patterns (§5). We present a number of security and privacy patterns (§6) together with a formal framework for validating them and assisting system designers in their application (§7). Finally, we conclude the paper with some directions for future work (§8).

2 Primer on Security Patterns

Patterns have been adopted into software engineering as a method for object-based reuse [**?**]. They capture solutions that designers tend to reuse in their software. In particular, security patterns have been proposed to assist designers in the identification and formulation of security solutions to recurrent security problems. Security patterns are essentially security best practices presented in a template format. They are defined as quadruples $\langle Context, Requirement, Solution, Consequence\rangle$, where *Context* defines the situation and conditions of applicability of the pattern, *Requirement* is the expression of the need for a system to enhance its privacy or security level, *Solution* is given as an abstract formal model whose application in the system-to-be guarantees the achievement of the requirement, and *Consequence* describes the potential effects of implementing the pattern. This format aids designers, included not security experts, in identifying and understanding security concerns, and in implementing appropriate security measures. A collection of security patterns is called security pattern system. Such a system stores a library of security patterns, together with guidelines for their implementation and the relationships among patterns. The proposed solutions should be verified and validated to guarantee that the security features are reliably implemented. The Common Criteria [**?**] defines seven evaluation assurance levels to assess the evaluation process and verify if the system meets all the requirements of its protection profile. As privacy is an extremely high risk concern for organizations, we should provide the highest evaluation assurance level that requires both formal verification and testing of the solution.

Schumacher [**?**] applies the pattern approach to security problems by proposing a set of security patterns to be applied during the software development process. Yoder and Barcalow [**?**] propose architectural patterns that are intended to increase the security level of an application. Fernández and Pan [**?**] design patterns for the most common security models such as authorization, role-based access control, and multilevel security. One of the main problems of these proposals is the lack of tools for the validation of patterns. Moreover, they are targeted to define IT solutions. In contrast, to understand the problem of security engineering, pattern designers also need to address security concerns of the organizational setting where the IT system operates.

Towards this direction, we can find the proposal by Mouratidis et al. [**?**]. Their security pattern language includes patterns addressing the communication between agents belonging to different agencies, the authentication of agents, and the provision of a restricted environment for unauthenticated agents. Unfortunately, the modeling framework adopted by the authors lacks fundamental concepts needed to capture security aspects of organizations and, once again, lacks an underlying formal framework for pattern validation. Models of interactions of agents can be found also in [**?**]. These models utilize negotiation policies which determine agents' behavior. Fernández et al. [?] propose security patterns for physical access control. These patterns are defined using UML class diagrams and sequence diagrams as the composition of basic patterns that address specific security aspects such as alarm monitoring, relays, and access control to physical structures. While the pattern entities are presented in detail, the authors do not provide any proof of validity for this pattern. The research work of IBM in 2003 on patterns tackled security risks when designing the system [?]. The focus of this work is on the implementation of efficient security and privacy solutions as integral part of a business process delivering value to its customers. Yet, those patterns are only available at the workflow (as opposed to organization) level and are not validated.

Most of above proposals also hardly make any reference to the underlying legal theory. This issue is partially addressed by Kienzle and Elder [?] who propose a template for specifying security patterns that includes information that may be not necessary to design patterns but is mandatory in a privacy and security context.

Another challenge concerns the application of security patterns. Yoshioka et al. [?] address this problem by applying security patterns to meet the security requirements for inter-company coordination systems. They propose to select security patterns at design time while still focusing on their efficient implementation. A problem of this approach is the lack of a formal framework that supports the analysis of the security requirements and determines precisely the context in which a pattern can be applied. Fernández and Yuan [?] proposed a methodological approach based on Semantic Analysis Pattern [?]. The idea underlying this work is to add instances of security patterns to the conceptual model of the application. Although this approach provides an interesting way to apply security patterns during the lifecycle of the application, it does not provide any formal analysis technique to drive system designers in the application of patterns.

3 Smart Items for Health Care

An area receiving significant attention from the Smart Items community is health care. The exploitation of Smart Items Infrastructures in the health care domain offers significant benefits in many situations ranging from regular monitoring of patients after hospitalization to emergency cases, where the life of patients is at risk. In particular, the health care domain requires designers to address a number of issues concerning access and integration of all available health care resources in order to offer a continuous, widespread, cooperative health care system and tools for personalized patient monitoring. Security and dependability concerns need to be carefully taken into account to make this exploitation successful. A Smart Items Infrastructures for health care has to deal with a variety of personal data (e.g., medical data of patients) subject to strict privacy regulations and confidentiality requirements.

Here we consider a case study showing how health care monitoring of patients after hospitalization might be managed through a Smart Items Infrastructures. This scenario focuses on Bob, a 69 years old widowed man who has been recently discharged from hospital after a cardiac arrest. Bob's health conditions need to be monitored 24 hours a day, and, for this purpose, Bob entered into a contract with the Health Care Centre (HCC), a provider of medical services. The HCC equips Bob with a smart T-shirt that incorporates a motion sensor providing an alert as soon as Bob becomes passive for two minutes and monitoring devices that regularly measure his heart rate, blood pressure, body temperature, etc. The smart T-shirt conveys all measured data to an e-

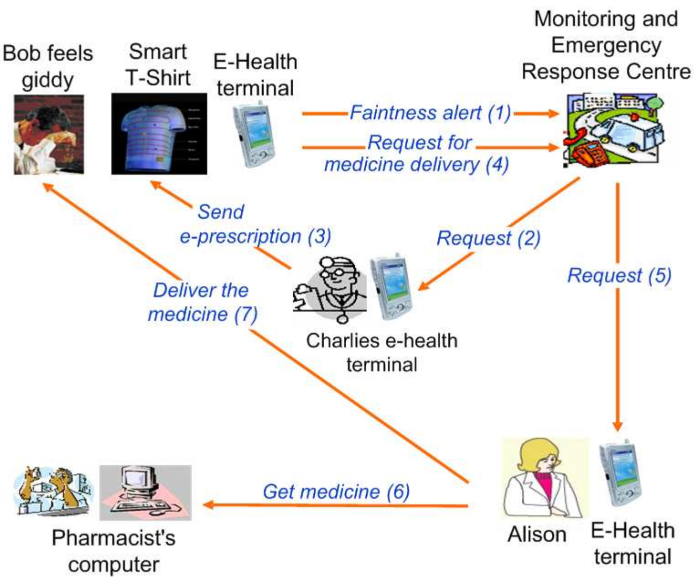


Fig. 2 Faintness alert and delivery of medicines

health terminal that allows Bob to promptly communicate them together with medical data to his doctor via the Monitoring and Emergency Response Centre (MERC), the department of the HCC responsible for receiving and handling patient requests for assistance. Last but not least, Bob has also subscribed to an experimental programme that aims, through a sensor network working behind the scene, to enhance his daily life at home and to provide additional data for better monitoring his health status. The sensor network is installed in Bob's house by the Sensor Network Provider (SNP), a contractor of the HCC. The SNP is also in charge of maintaining the sensor network in Bob's house.

Among the possible situations that can be envisaged in this scenario, we focus on a few of them, meaningful enough to highlight the security and privacy issues of interest for this paper. Figure 2 depicts the faintness alert and delivery of the medicine scenes, and Figure 3 shows the false alarm from the motion sensor installed on Bob's smart T-shirt scene.

Scene 1 (faintness alert) Bob feels giddy and proceeds by sending a request for assistance to the MERC through his e-health terminal. The MERC knows from its database that Bob's doctor is on vacation and thus starts a doctor discovery process that consists of sending a broadcasting message to a group of doctors able to substitute Bob's doctor. Dr. Charlie is the first to answer and so he is appointed by the MERC to substitute Bob's doctor for this request. Dr. Charlie interrogates the MERC via his e-health terminal to receive Bob's medical data from the smart T-shirt and medical history. He writes the electronic prescription on his e-health terminal and sends such a prescription to Bob's e-health terminal.

Scene 2 (delivery of the medicine) Bob feels weak and instead of driving to the pharmacy to get the medicine, he asks the MERC to accomplish this task. A

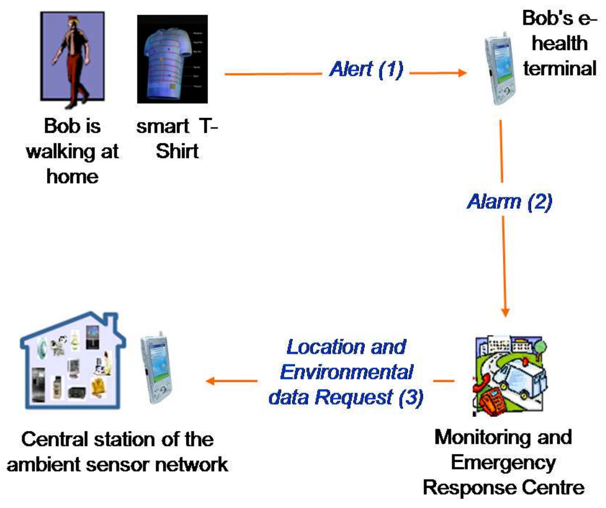

Bob is walking at home
smart T-Shirt
Alert (1)
Bob's e-health terminal
Alarm (2)
Location and Environmental data Request (3)
Central station of the ambient sensor network
Monitoring and Emergency Response Centre

Fig. 3 False alarm from Bob's smart T-shirt

social worker, Alison, receives a message from the MERC on her e-health terminal to go to the pharmacy and get the medicine to be delivered to Bob as an electronic credential. This message also contains the medical prescription issued by the doctor to Bob. Alison acknowledges the request and goes to the pharmacy. After a successful message exchange between Alison's e-health terminal and the pharmacist computer, the medicine is given to the social worker that proceeds in delivering it to Bob.

Scene 3 (false alarm) The motion sensor on Bob's smart T-shirt raises an alert stating that Bob has been totally passive for more than 2 minutes. Bob's e-health terminal gets this alarm and sends it to the MERC. The MERC queries the smart home to localize Bob. The home's central station receives the localization request and replies back Bob's position, movements and surroundings after interacting with the sensor network infrastructure installed by the SNP. Using this information the MERC has evidence that Bob is safe and that the motion sensor on Bob's smart T-shirt is probably not working properly.

It is clear that the Smart Items Infrastructures presented in this section has to integrate a variety of sensors and devices, as well as human actors and organizations. In this context, security and privacy requirements need to be carefully considered to comply and abide to data protection regulations.

4 Security & Privacy Requirements from a Legal Perspective

The spread of IT has facilitated the flow of information among individuals and companies. For instance, it is almost immediate from our running example that the HCC needs to collect, store and further process a large amount of data to provide health care services. This introduces several security and privacy issues in the IT landscape.

In this paper, we particularly focus on the European Directive on Data Protection [**?**], which provides guidelines for the collection, use, storage, distribution, and other ways of processing personal data in Member States.

The European Directive on Data Protection (hereafter referred to as the Directive) is considered a milestone in the history of the data protection in Europe. The motivation underlying the Directive is the need to promote the right to privacy and to harmonize data protection laws of Member States. The Directive defines a general legal framework that is technologically neutral: the principles and provisions established by the Directive are general enough to be applied to new technologies and situations, even if it may be necessary to translate those general rules into guidelines accounting for the specificity of those technologies [**?**].

The Directive defines various subjects whose rights and obligations are regulated in relation to processing of personal data (Article 2):

– A *data subject* is either a person who is already known or a person who can be identified, directly or indirectly by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity. In accordance with Recital 26 of the Directive, a person is identifiable when a controller or any other person can identify the person with help of any reasonably means. The data subject is granted various rights, such as right to be informed, right of access to data, or right to object to the processing of his personal data. In our scenario, Bob plays the role of data subject.
– A *data controller* is the natural or legal person, public authority, agency or other body, which alone or jointly with others determines the purposes and means of the processing of personal data. The Directive imposes various obligations upon data controllers, such as to ensure that all principles established by the Directive are observed (Article 6). Other important obligation of data controllers is to implement appropriate technical and organizational measures to protect personal data against accidental, unlawful or unauthorized destruction or loss, alteration, disclosure or access (Article 17, paragraph 1). In our scenario, the HCC acts as the data controller. Actually, the HCC is the entity who determines purposes (e.g., health care services) and means (e.g., smart T-shirt, eHealth terminal, etc.) of the processing of personal data.
– A *data processor* is defined as a natural or legal person, public authority, agency or any other body which processes personal data on behalf of the controller. The relationship between the data controller and the data processor must be governed by a contract or legal act binding the processor to act only on instructions set by the controller and to implement appropriate technical and organizational measures ensuring secure processing of personal data. This contract or legal act must be in a written form at least in those parts concerning data protection and security measures (Article 17). Due to the nature of this legal act, a data processor cannot be an employee of the data controller [**?**]. In our scenario, any legal subject that acts on behalf of the HCC is a data processor. For instance, the SNP plays the role of data processor. The SNP has the responsibility to maintain the sensor network (e.g., servers with databases of collected personal data) and thus it can access patient personal data.

Personal data may be processed only when one of the legitimate grounds allowing processing of personal data is given. In accordance with Article 7, there are generally six grounds justifying processing of personal data.

(p1) The processing is legitimate when the data subject has unambiguously given his consent.[2]

When such consent does not exist, personal data may be processed only if processing is necessary

(p2) to perform a contract to which the data subject is a party or to take steps at the request of the data subject prior to entering into a contract; or
(p3) to comply with a legal obligation imposed upon the controller; or
(p4) to protect the data subject's vital interests; or
(p5) to perform a task carried out in the public interest or in the exercise of official authority vested in the controller or in a third party to whom the data are disclosed; or
(p6) to pursue legitimate interests of the controller or the third party(ies) to whom the data are disclosed.

Data concerning a person's health are considered "sensitive personal data" by the Directive and shall be processed using special protection. For instance, the standard ISO/IEC 17799 [**?**] suggests to change passwords of accounts entitled to manage sensitive data more often than passwords for "normal" accounts. The processing of sensitive personal data is subject to additional rules. It is generally prohibited unless certain conditions are met (Article 8): either the data subject gives his explicit consent with such processing[3] or other conditions must be met. Among such conditions, we mention two that are relevant for our purposes:

(c1) "processing is necessary to protect the vital interests of the data subject or of another person where the data subject is physically or legally incapable of giving his consent" (Article 8, Paragraph 2(c));
(c2) "processing of the data is required for the purposes of preventive medicine, medical diagnosis, the provision of care or treatment or the management of health-care services, and where those data are processed by a health professional subject under national law or rules established by national competent bodies to the obligation of professional secrecy or by another person also subject to an equivalent obligation of secrecy" (Article 8, Paragraph 3).

Next we analyze some situations occurring in our health care scenario and discuss possible solutions along the lines suggested by the Directive. Notice, however, that additional and more specific requirements may be set in national regulations.

Example 1 The MERC appoints Dr. Charlie to provide medical care to Bob (Scene 1 in Section 3). However, Dr. Charlie is not Bob's doctor. In order to access Bob's medical records to achieve his duties Dr. Charlie must have Bob's consent. At first, Bob must agree with the proposed doctor as he has the right to choose his own physician [**?**]. After the consent is given, the MERC is entitled to disclose Bob's medical data to Dr. Charlie. According to this scenario, the HCC shall provide Dr. Charlie with the permissions necessary to access Bob's medical records. In case of Bob being too giddy and unable to give his consent the rules (p4) and (c1) (see above) normally apply.

[2] The consent of data subjects is defined as any freely given consent and informed indication of his wishes by which the data subject signifies his agreement to personal data to him being processed (Article 2, letter h).

[3] Notice that the laws of the Member State may provide that the prohibition of processing sensible personal data is not lifted by the data subject's giving his consent (Article 8, Paragraph 2(a)).

In this particular case, sensitive personal data may be revealed to the substituting doctor as these are medical data in accordance with Article 8, paragraph 3 of the Directive. However, the scope of application of this exception varies from Member State to Member State. The reason is that national laws define the terms used in this provision in their own way. If a data controller intends to process sensitive data for purposes that in accordance with a national law cannot be interpreted as preventive medicine, medical diagnosis, provision of care or treatment, or the management of health care services, the patient must give his explicit consent with such processing. For instance, the explicit consent of the patient is needed when the HCC appoints the SNP to monitor patient health status as shown in the following example.

Example 2 The HCC needs to collect information about patients to provide certain services. The HCC outsources the task of collecting information about Bob's position, movement and surrounding environment to the SNP. These personal data are sensitive personal data as they serve to determine the patient health status. As the data processing is not performed on the basis of the legal obligation to provide health care, the HCC needs the explicit consent of the patient to process the data. Moreover, the HCC needs to enter into a written contract with the SNP. This contract must describe the security measures taken by the SNP and a stipulation that the SNP acts only on instructions from the HCC.

Placement of the sensor network in the patient's house entails various legal problems. As the sensor network monitors all people entering the house, data concerning subjects different from the patient may be collected. The processing of these data differs depending on their nature. If such data make it possible to identify a person, their processing must be performed on legal basis. Specifically, if sensors monitor only the position, the processor is entitled to process such data as they are not sensitive personal data and their processing is necessary to protect the legitimate interest of the processor to run the sensor network. Monitored subjects, however, need to be informed about such processing. On the other end, if sensors collect sensitive personal data, for example body heating, the explicit consent of the identifiable data subject is necessary. The processing of personal data is also regulated by several principles (Article 6).

The Directive establishes that data processing has to be fair and lawful. Personal data must be collected for specified, explicit and legitimate purposes and may not be processed in a way incompatible with those purposes. Such data should be adequate, relevant and not excessive in relation to the purpose for which they were collected. Data must be accurate and kept up to date and kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the data were collected or for which they are further processed. For security reasons, persons in charge of the data processing shall only be able to access that part of the information which they *need to know* to carry out assigned duties.[4]

Example 3 Bob requests the assistance of the MERC to get medicines delivered to his house (Scene 2 in Section 3). The MERC appoints Alison to accomplish this task, who receives a precept to deliver certain medicines to Bob together with his personal data. However, Alison should not have access to more information than that strictly necessary to carry out her functions. For instance, she should not have access to Bob's medical records, but should only be informed about which medicine has to be delivered and to which destination.

[4] This correspond to the "least privilege" principle proposed by Saltzer and Schroeder [**?**].

As the Directive does not regulate all aspects of the data processing and as it allows Member States to lay down more specific regulation, some other security and privacy issues are dealt with in national regulations. For instance, according to Article 1228 of the Italian Civil Code ("Liability for acts of auxiliaries"), data controllers and processors, who avail third parties in the execution of data processing, are still liable for their malicious, fraudulent, or neglect acts, unless otherwise intended by the parties. Thereby, data controllers and processors need warranties that appointed employees have taken charge of the assigned duties.

Example 4 The HCC is responsible for the delivery of medicines to patients. The MERC appoints Alison to deliver medicine to Bob (Scene 2 in Section 3). Although the MERC avails another subject in the performance of the obligation, it is still liable for the failure of the service. Thus, the MERC may want warranties from Alison about her commitment to achieve assigned duties.

5 The SI* Modeling Framework

To understand why and where solutions to security and privacy problems have to be deployed, system designers must model the goals and assets of the stakeholders within the socio-technical system, the assignments of permissions and responsibilities among them, and the trust network. Indeed, the understanding of the organizational context supports system designers in making the correct decision about the security solutions to be adopted, because not all organization processes are under the control of the IT system. Agent- and goal oriented approaches are found effective in organizational modeling and in easing the elicitation of functional and non-functional requirements for developing socio-technical systems. In this work, we have adopted Secure Tropos [**?**], an agent- and goal-oriented security requirements engineering methodology designed to model and analyze security aspects of socio-technical systems.

The Secure Tropos methodology adopts the SI*[5] modeling language [**?**] for the acquisition of the requirements model. SI* employs the concepts of actor, goal, and resource: an *actor* is an intentional entity that performs actions to achieve goals; a *goal* is a strategic interest of an actor; a *resource* represents a physical or an informational entity. Every actor is defined along with a set of objectives, entitlements, capabilities: *objectives* are goals intended to be achieved or resources required by the actor; *entitlements* are goals and resources controlled by the actor; finally, *capabilities* are goals and resources that the actor is able to respectively achieve and furnish.

SI* also employs the notion of *permission delegation* to model the transfer of entitlements (the *delegatum*) from an actor (the *delegator*) to another actor (the *delegatee*), and the notion of *execution dependency* to model the transfer of objectives (the *dependum*) from an actor (the *depender*) to another actor (the *dependee*). Moreover, the language adopts the notions of *trust of permission* and *trust of execution* to model the expectation of one actor (the *trustor*) about the behavior of another actor (the *trustee*) on a goal or resource (the *trustum*). In particular, trust of permission is used to model the trustor's expectation that the trustee does not misuse the trustum, and trust of execution is used to model the trustor's expectation concerning the ability and dependability of the trustee in accomplishing the trustum. The graphical notation of the constructs comprising the SI* modeling language is presented in Fig. 4.

[5] SI* is read as "see star".

From a methodological perspective, Secure Tropos is based on the idea of building a model of the system that is incrementally refined and extended. Specifically, goal analysis consists of refining goals and eliciting new relations among actors. It is conducted from the perspective of single actors using goal refinement, contribution analysis, and means-end analysis:

Goal refinement analyzes and refines goals to build a finer goal structure in terms of AND and OR decompositions.

Contribution analysis studies the impact of the achievement of goals on the achievement of other goals. This impact can be positive (denoted by '+') or negative (denoted by '-').

Means-end analysis is intended to identify the goals that provide means for achieving a goal and the resources needed and produced by the achievement of a goal.

The above constructs allow designers to capture the requirements model of organizations together with their IT systems in a number of graphical diagrams, namely

Actor Diagram describing objectives, entitlements and capabilities of each actor. They are also analyzed using goal analysis techniques from the perspective of single actors.

Trust Diagram describing the trust network. In particular, this diagram represents the expectations of actors in terms of trust of permission and trust of execution.

Execution Dependency Diagram identifying the dependencies among actors and in particular to which actor the achievement of which goals or delivery of which resources has been assigned by which actor.

Permission Delegation Diagram identifying the transfers of rights between actors and in particular to which actor has been delegated the permission on which goals or resources by which actor.

The actor diagram identifies the main stakeholders and system actors involved in the system. The other diagrams enrich such a diagram by representing different views of the requirements model. Fig. 5 shows the requirements model of the health care scenario. For simplicity we show all diagrams in a single picture, but in more complicated scenarios separate diagrams might be more readable. In addition, notice that the figure depicts a fragment of a real health care scenario where goals and resources are presented at a quite high level of details. More refined and detailed vision of the scenario can be compiled. At first we need to assign tasks to the various actors so that the model reflects what they are supposed to do, i.e. the Execution Dependency Diagram.

Example 5 A patient, Bob, wants medical services (represented using a request relation) and depends on the HCC for achieving the goal provide medical services. To achieve this goal, the healthcare provider needs to handle patient request and perform administrative duties. This aspect is captured using an AND decomposition. To satisfy goal handle patient request, the HCC depends on the MERC that is in charge of managing the supply of health care service. In our example, we consider services manage faintness alerts and deliver medicine. In particular, goal manage faintness alerts consists in a doctor discovery process, which the MERC has the capabilities to perform (represented using a provide relation), and provide medical care, for which the MERC depends on Dr Charlie. The doctor needs patient data to achieve his duties and, consequently, depends on the MERC for goal provide patient data.

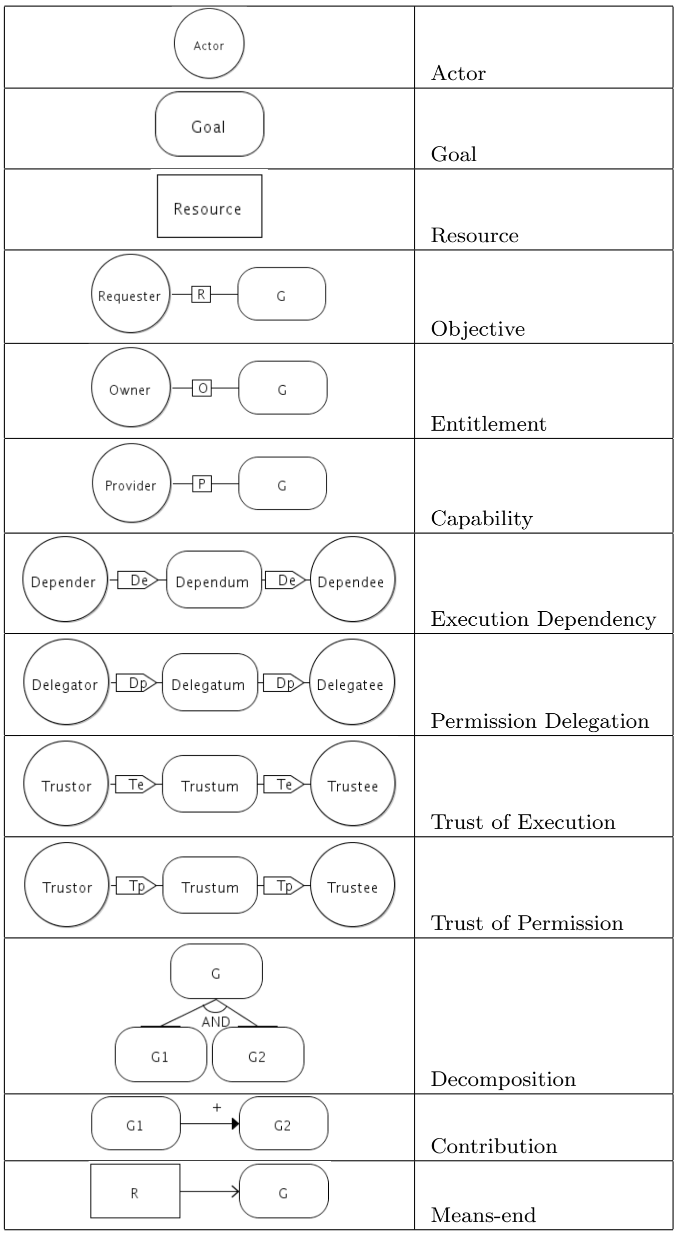


Fig. 4 SI* Graphical Notation

A cascade of authorizations from one actor to the next have to be added, i.e. the Permission Delegation Diagram.

Example 6 From a permission perspective, Bob, who is the data subject (represented using an ownership relation), gives the consent to manage patient data to the HCC (represented using a permission delegation relation). Manage patient data consists in

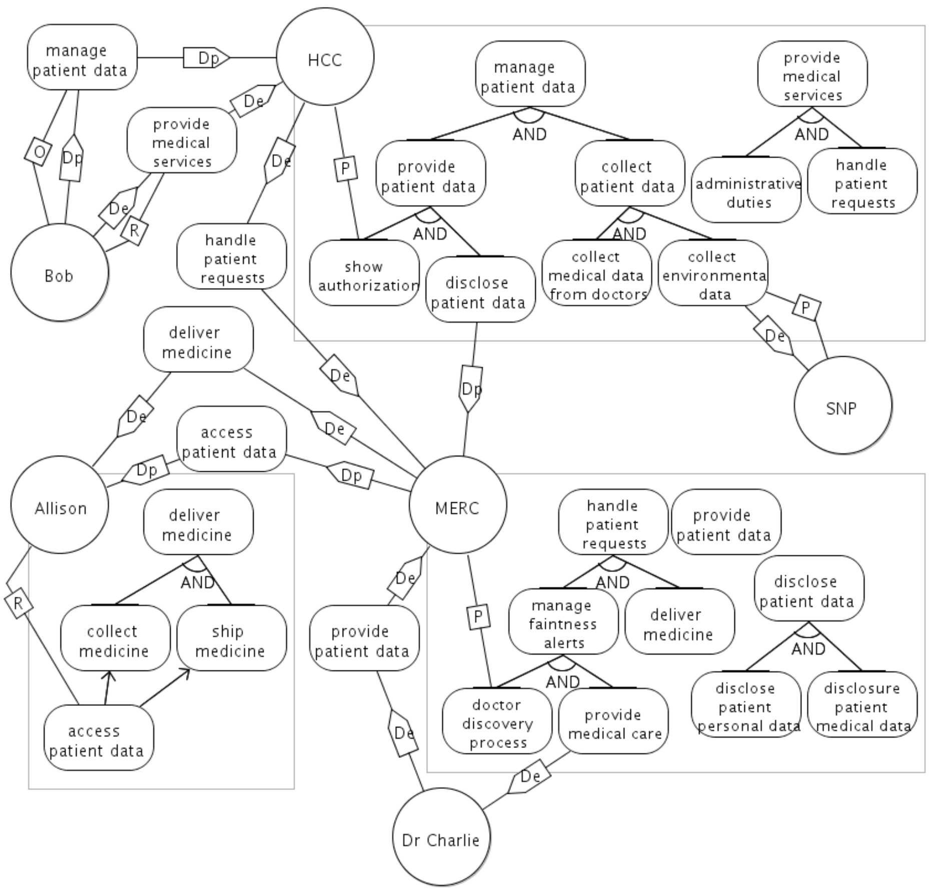


Fig. 5 SI* Model of the health care scenario

provide patient data and collect patient data. Patient data are collected from both the doctor, who has visited the patient, and the sensor network placed in the patient house. The HCC appoints the SNP to collect environmental data. Provide patient data consists in show authorization, which the HCC has the capability to achieve, and disclose patient data. The HCC authorizes the MERC to disclose patient data to those actors who need patient data to achieve assigned duties.

In many cases we need to transfer both the execution of a task and the explicit permission to do it (see discussion on Section 4 and the example below).

Example 7 To achieve goal deliver medicine, the MERC depends on Alison, who is a social worker. Assigned duties consist in collect medicine and ship medicine, and their achievement requires to access patient data. As the permission on deliver medicine does not imply the permission on access patient data, we represent the relations between goals access patient data and collect medicine and between goals access patient data and ship medicine using means-end relations.

Notice that Fig. 5 does not include any trust relation. Actually, the analysis of the case study has not identified such relations. This lack can require to put in place mechanisms to ensure the security and dependability of the socio-technical system. For instance, some patterns presented in this paper have been designed to address this problem.

6 Making Security Patterns Aware of Legal Issues

The EU SERENITY project proposes a pattern library populated by security and privacy patterns targeting different levels of abstraction. The library contains patterns covering the organizational level [?,?], workflow and services levels [?], and network and devices level [?]. This section addresses the problem of how to capture security patterns at the organizational level from legal requirements.

For the development and validation of security and privacy patterns that meet legal requirements, we have adopted the following approach:

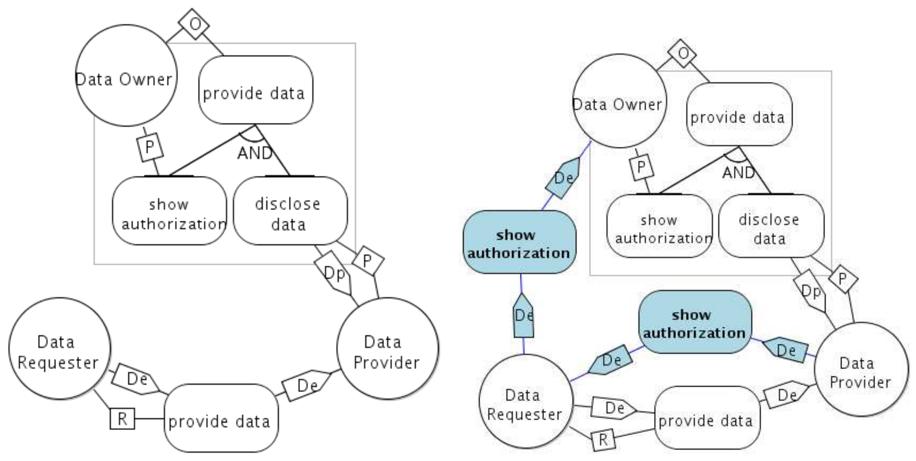
– The legal requirements have been analyzed and specified as logical statements. This required to go through the issues sketched in Section 4 and identify the relevant actors (e.g, data subject, data controller, etc.) as well as their right and responsibilities and the relations among them.
– The methodology presented in Section 5 has been employed to analyze the organizational contexts and identify situations in which legal requirements are not satisfied. Those situations have been used to define general contexts in which solutions for enforcing legal requirements are necessary.
– Pattern solutions have been captured using SI* diagrams and specify the changes to be brought to the organizational structure in terms of organizational goals, trust relations, and assignments of permissions and responsibilities between actors. They have been designed by taking into account best practices and existing protection measures.
– Patterns have been validated using the logical formalization by verifying that they satisfy legal requirements.

We have singled out some patterns to show the outcome of this process and the interplay of security and legal consideration. We recall that each pattern is defined in terms of the context in which the pattern is applied, the requirements to be ensured, the solution to be adopted, and the consequences of applying the pattern.

6.1 Access Control Pattern

Organizations specify access control policies to restrict access to data maintained in their IT systems. An access control policy, defined as a set of rules, states the actions that subject can perform on resources [?]. At the organizational level, we are of course not interested in the low level mechanism for access control but the high-level relations among actors. The SI* diagram should reflect the relations as they could emerge from a security policy document. So it should capture what is needed to achieve organizational goals such as resource allocation with the necessary authorizations, delegations, and trust relations.

The access control pattern is defined as follows:



(a) Context (b) Solution

Fig. 6 Access Control Pattern

Context The Data Requester wants access to certain data and depends on the Data Provider for it. The Data Owner has full disposition on who can access them.[6] The Data Owner delegates the permission to disclose data to the Data Provider. The Data Provider, however, can disclose the data only after the Data Requester has provided the necessary credentials (issued by the Data Owner). A SI* diagram representing the context is presented in Figure 6(a).

Requirement The Data Requester shall access information if he is an authorized actor.

Solution. Once the Data Provider receives an access request, he requests an authorization certificate from the Data Requester. The Data Requester forwards this request to the Data Owner. If the Data Owner provided the required evidence, the Data Provider discloses the data to the Data Requester. A SI* diagram representing the solution is presented in Figure 6(b) where the added elements and relations are represented in blue.

Consequences. The pattern solves the problem of granting the permission to access data to the Data Requester. However, new issues may arise after the application of this pattern. For instance, the application designers should verify whether the Data Requester actually needs that permission to achieve his duties. Other issues may arise due to the expectations of the Data Owner about the use of the data by the Data Requester.

If we consider the context from a purely technical perspective going through the data provider would be pointless. Indeed, the data provider could simply give an authorization to the data requester. The latter could then show this authorization when access to the data is needed. This is precisely what happens when we need to login to a web forum

[6] Notice that the Data Owner may be different from the Data Subject.

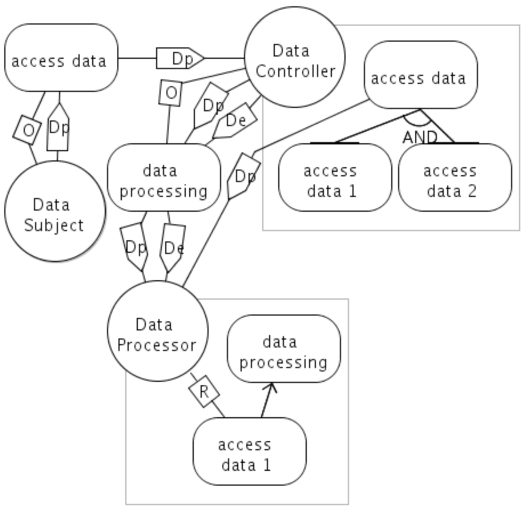
server: we just ask the administrator for a password. Legally speaking this situation is not acceptable for situations where personal data are disclosed: the authorization has to come from the data owner who is entitled to give authorization. This is represented in the picture by Ownership link (O) between the data owner and the "provide data" goal which is further decomposed into "show authorization" and "disclose data". In some cases the data owner is also able actually to provide it so that the data requester can ask him directly.

This pattern applies to many situations of our health care scenario, for instance, any time an actor has to access patient data. One example is Scene 1 where Dr. Charlie (i.e., the Data Requester) interrogates, through his e-health terminal, the MERC (i.e., the Data Provider), properly delegated by the HCC (i.e., the Data Owner), for Bob's medical data and medical history. Before granting the access, the MERC has to verify whether Dr. Charlie is authorized by HCC to receive Bob's medical data and medical history.

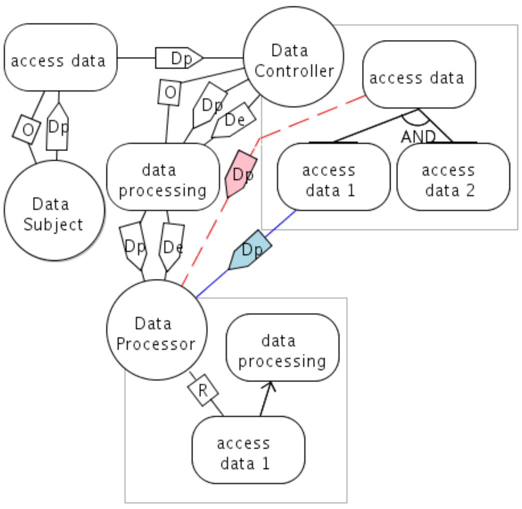
6.2 Need-To-Know Pattern

Among privacy and security principles, the *need-to-know* principle is one of the most fundamental. Legislation requires that the persons in charge of the processing shall have the permissions strictly necessary to achieve their duties. The practice of need-to-know limits the risk due to actors who abuse their trusted position within the organization, and failures in implementing it can cause severe damage to the organization. The need-to-know principle imposes a dual responsibility. When achieving assigned duties, data processors are expected to limit their requests for permission to that which they need to perform their duties. Under some circumstances, they may be expected to explain and justify their requests. Usually, this is the case when an individual lodges a claim concerning his rights related to the processing of personal data to a national supervisory authority. This supervisory authority then uses its investigative powers to examine the claim and may demand the above mentioned explanations. At the same time, controllers are expected to ensure that anyone to whom they give permission legitimately needs to have such permission. The implementation of the need-to-know principle at run time (as opposed to design time) can be problematic, because it is difficult to determine that a processor needs permission to achieve the assigned duties until they have been assigned. Below we present the pattern used to enforce the need-to-know principle when the data controller grants access to information that are not strictly necessary to perform the data processing.

Context. The Data Controller appoints the Data Processor to perform a data processing. The Data Controller also authorizes the Data Processor to perform it. To achieve the assigned duties, the Data Processor needs some information (i.e., goal "access data 1"). However, the Data Controller does not grant only the permission necessary to achieve the assigned duties. Rather the Data Controller authorizes the Data Processor to access all data (i.e., goal "access data") that include data that are not necessary to achieve assigned duties (i.e., goal "access data 2"). A SI* diagram of the context is presented in Figure 7(a).



(a) Context



(b) Solution

Fig. 7 Need-To-Know Pattern

Requirement. The Data Processor shall have only the permission necessary to achieve assigned duties.

Solution. The Data Processor does not need access to all information about the Data Subject, but only the part relevant to carry out assigned duties. It is up to the Data Controller to determine what access rights should be granted to the Data Processor and to ensure that the access rights unnecessary to carry out assigned duties are not granted. A SI* diagram representing the solution is presented in Figure 7(b). The red dashed line represent the relation deleted from the context and the blue line represent the new relation between the Data Controller and the Data Processor.

Consequences. After the application of this pattern, the Data Processor has the permission strictly necessary to achieve assigned duties.

An important feature of the need-to-know pattern lies in making visible to the business application designer the significance of clearly separating those data that are strictly necessary to accomplish a certain task from those that are not. It is not unusual that this aspect is, otherwise, either neglected or only partially tackled by means of common tools such as Non Disclosure Agreements (NDA) through which the data controller contractually bind the data processor to not disclose sensible information learned in executing its tasks. Obviously NDA are a good means, and sometime the only feasible one, to mitigate the risk of improper disclosure of sensible data, but assurance can only be achieved by stronger solution like Need-to-Know practices. Actually, Need-to-Know and NDA should be jointly used to provide that high level of protection required by applications processing critical sensible data.

This pattern fits in Scene 2 of our scenario. Bob (i.e., the Data Subject) is feeling weak and is so unable to go to the pharmacy himself. He requests the HCC (i.e., the Data Controller) for assistance. The HCC delegates this task (through the MERC) to Alison (i.e., Data Processor) who agrees to carry out this task. Alison should not be provided access to all of Bob's personal information, but only the information that is necessary for her to deliver the medicine. For example, Alison would need to know Bob's address, phone number, the prescribed medicine, dosage and how it is to be administered.

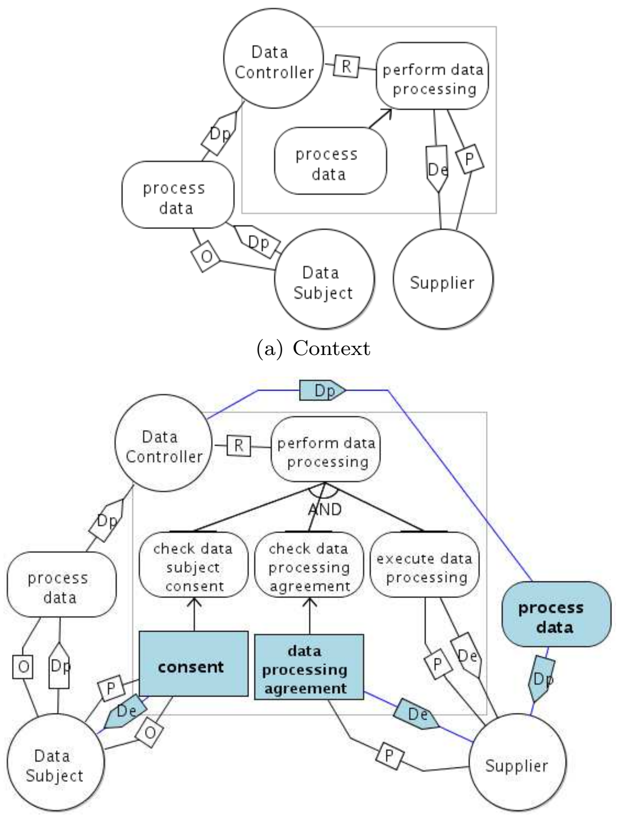
6.3 Outsourcing Pattern

Outsourcing is the transfer of management control of business activities to an outside supplier [?]. This business strategy is adopted to reduce costs, but has a strong impact on the security and privacy requirements of organizations. For instance, ISO 9001 [?] states that "where an organization chooses to outsource any process that affect product conformity with requirements, the organization shall ensure control over such processes". From a privacy perspective, the organization must demonstrate to have sufficient control over the outsourced data processing. Below we present a pattern for outsourcing.

Context. The Data Controller outsources the execution of data processing to an outside Supplier for which Data Subject's data are needed. However, in accordance with the Directive, only the Data Controller has a legal ground entitling him to process such data. A SI* diagram representing the context is presented in Figure 8(a).

Requirement. The Supplier shall have a legal ground on which basis he is allowed to process personal data.

Solution. When the Data Controller is entitled to process personal data by himself, he can then outsource processing of these data. This outsourcing is legal when the Data Controller enters into a written contract with the Supplier. This contract enforces the Supplier to act only on instructions set by the Data Controller and to adopt the specified security measures. All principles related to the data quality must be followed



(a) Context

(b) Solution

Fig. 8 Outsourcing Pattern

as well. The Data Subject has a right to be informed about how his personal data are processed. A SI* diagram representing the solution is presented in Figure 8(b) where the colored constructs represent the changes made to the context.

Consequences. The application of the pattern solves the problem of granting to the Supplier the permission necessary to perform outsourced data processing. The Data Controller may outsource data processing without any previous Data Subject's consent if he is entitled to process personal data by himself. For processing of personal data the Data Controller sometimes needs Data Subject's consent, sometimes he can process the data on basis of another legal grounds. At the same time, the application of the pattern introduces new issues. For instance, the Data Controller may want assurance that the Supplier does not repudiate the data processing agreement.

The outsourcing pattern is captured only for cases in which data controllers and data processors are subjects established under national laws of EU Member States. When soliciting consent from the data subject, the data controller must provide them with information specified in a relevant law. On the other hand, while planning later to outsource processing of personal data, the data controller has to look for a supplier *providing sufficient guarantees in respect of the organizational and technical security measures. Contracts between these two subjects usually contain clauses dealing with, but not limited to, management review, board oversight, and risk management. Each of these clauses necessitates a "periodic verification" and an "expiration date". Nevertheless, outsourcing of processing to data processors established in a non-EU country is subject to additional rules on transfer of data into third countries. Such outsourcing of processing is allowed only if law of the non-EU country in question ensures an adequate level of personal data protection. If not, then the outsourcing is possible only exceptionally, for instance if a data subject gives his unambiguous consent.*

This pattern fits in Scene 3 of our scenario. To monitor Bob's health, the HCC (i.e., the Data Controller) needs the assistance of the SNP (i.e., the Supplier), which keeps track and reports all information relating to Bob's environment. However, in this case the HCC cannot ask the SNP to monitor and collect Bob's environmental (sensor) information unless Bob has already given the consent for the collection of his personal data and been informed about why this information is being collected, to what extent and how it will be processed. Moreover, the HCC needs an agreement signed by the SNP on the use of collected data and the security measures adopted by the SNP to protect patient data.

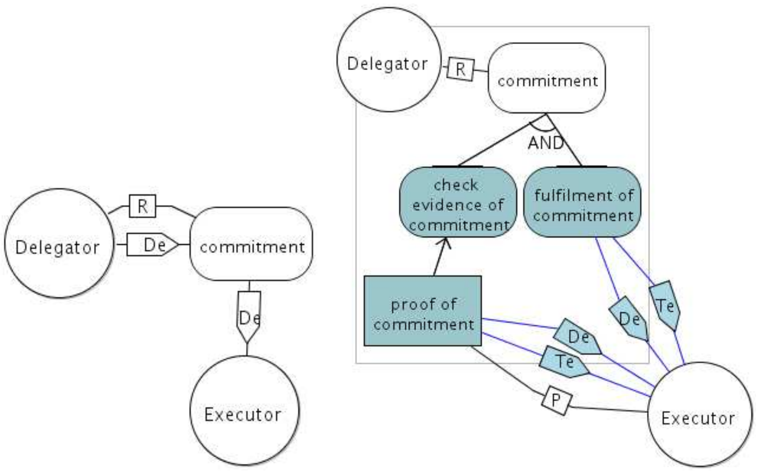
6.4 Non-Repudiation Pattern in Absence of Trust

To accomplish its daily tasks an organization exploits its social infrastructure by decomposing each task into sub-tasks that are then distributed/delegated among groups of actors having pre-defined relations. For those tasks and sub-tasks considered of critical importance for the organization, a simple statement of delegation is not sufficient to shift responsibility for the task performance from one actor to another. In these cases, a delegator might want a non-repudiable acknowledgment from the executor to have a proof that the latter has explicitly taken the responsibility for executing the task. On its side, the executor, after the fulfillment of the task, might want to have a non-repudiable acknowledgment from the delegator. Non-repudiation patterns address these issues. For sake of simplicity, we show here only the unilateral version of the non-repudiation pattern where the executor takes the responsibility of the task.

Context. The Delegator requests the achievement of a commitment and delegates its execution to the Executor, but the former has no warranties that the latter takes the responsibility of achieving the commitment. A SI* diagram representing the context is presented in Figure 9(a).

Requirement. The Delegator shall have evidence that the Executor cannot repudiate his commitment.

Solution. The Delegator refines the commitment into two sub-parts. The first part is used to check the evidence about responsibilities taken by the Executor and the second represents the actual desire of fulfilling the commitment. To achieve the commitment, the Delegator delegates the execution of the commitment to the Executor together with a request for a proof of commitment. Once received, this proof ensures the Delegator that the Executor has taken the responsibility of the commitment. The ultimate objective



(a) Context (b) Solution

Fig. 9 Non-Repudiation Pattern

of the pattern is to build a trust relation between the Delegator and the Executor for the achievement of the commitments. At the end of the day, in case the Executor fails to achieve the commitment, the Delegator can challenge the proof of commitment in front of the judge. A SI* diagram of the solution is presented in Figure 9(b) where the colored elements represent the changes added to the context. For the sake of simplicity, we have assumed a trust relation (**Te**) between the Delegator and the Executor for the proof of commitment. Actually, this relation implies additional mechanisms (e.g., a trusted third party which stores proofs of commitment) that justify its presence in the model.[7]

Consequences. The proof of commitment provided by the Executor needs to be cautiously kept by the Delegator. If the Executor does something wrong in the execution of the commitment, the proof of commitment is the only mean the Delegator can use to discharge itself from any responsibility.[8] Therefore, the application of the pattern requires the adoption of additional measures, such as backup infrastructures, to better protect the proof of commitment.

This pattern is particularly interesting because no permission is involved. So, if we made the mistake of equating legal requirements with permissions, it would look like a pattern where no legal concern was captured. On the contrary, as we noticed while introducing the pattern, there appears a legal issue of liability. When an actor is assigned the target of fulfilling a goal in the model such assignment does not correspond to the whims of the designer but to a precise commitment of the scenario we are trying

[7] Those mechanisms are strongly related to the specific application domain and their investigation falls outside the scope of the paper.

[8] If the Delegator is in a position of an employer and the Executor in a position of an employee, the proof of commitment may be used by the Delegator for claiming damages from the particular employee who failed to perform the task rather than releasing the Delegator from liability.

Pro1	$\leftarrow \mathsf{request}(A, S) \wedge \mathsf{not\ can\_achieve}(A, S)$
Pro2	$\leftarrow \mathsf{have\_perm}(A, S) \wedge \mathsf{not\ need\_to\_have\_perm}(A, S)$
Pro3	$\leftarrow \mathsf{need\_to\_have\_perm}(A, S) \wedge \mathsf{not\ have\_perm}(A, S)$
Pro4	$\leftarrow \mathsf{dependency}(A, B, S) \wedge \mathsf{not\ trustChain\_exec}(A, B, S)$

Table 1 Security and Privacy Constraints

to capture. Such commitments stems from implicit or explicit business, contractual or legal obligations. The actor in charge of the commitment might in some cases delegate the goal or subgoals to other actors but usually retains the responsibility in case the delegatee fails to deliver. Therefore he has two choices: either to trust the delegatee or to ask some proof of commitment (or proof of delivery) that can be used in court. The latter choice is precisely the case covered by the pattern.

This pattern fits into Scene 2 of our health care scenario where the medicine has to be delivered to the patient. For instance, the MERC (i.e., the Delegator) delegates the execution of the task to Alison (i.e., the Executor). Applying the above pattern, the MERC disposes of a proof stating that Alison acknowledged the request and accepted the responsibility of executing it.

7 Formal Analysis & Validation

Once security and privacy patterns have been designed by security and legal experts, we need some tool to validate them. In other words, we want assurance that the application of each pattern makes the system compliant with the requirements for which it has been designed.

As done in [?], we use the Answer Set Programming (ASP) paradigm [?] to accomplish our purposes. The ASP paradigm is based on the concepts of facts, rules, and constraints expressed as Horn clauses and evaluated using the stable model semantics. A fact consists of a relation symbol, called *predicate*, together with the appropriate number of well-typed arguments. Predicates are distinguished in two types: extensional and intensional. Extensional predicates correspond to edges and nodes of the graphical model defined by the system designer, whereas intensional predicates are derived by the reasoning system. Every graphical construct is then mapped into ground logical facts. *Rules* are used to define the semantics of SI* concepts and to derive the information needed for requirements analysis. *Constraints* encode security and privacy requirements in a form that is suitable for their verification. In particular, constraints specify conditions that must not be true in the model. In other words, constraints are formulations of possible inconsistencies.

Table 1 presents the formalization of the requirements addressed in this paper. Pro1 implements the access control requirement. It verifies if actors have requested data to actors that have both the capabilities and permission to provide them. Pro2 implements the need-to-know requirement. It verifies if actors who have a permission, actually need such permission in order to achieve a service assigned to them. Pro3 checks that actors are not trapped into a contrary-to-duty situation. It verifies if actors have the permission necessary to achieve the assigned duties. Pro4 implements untrusted delegation requirements. It verifies if actors are confident to achieve objectives assigned to other actors by detecting occurrences of untrusted execution dependency in the model.

If constraints are not satisfied, weaknesses or vulnerabilities may occur in the actual implementation of the system or in the policies and procedures adopted by the organization. It is up to the designer to decide whether or not such a failure compromises the system and adopt the adequate countermeasure. The selection of countermeasures depends on several factors such as context, risk, cost of the solution, compliance with legislation, etc. The patterns proposed in this paper are intended to assist designers, including those that are not security expert, in this decision making. Notice that the properties presented in Table 1 are general so that they can be used to verify the correctness of the model with other requirements for which other patterns should be applied. For instance, in [?] a monitor pattern has been proposed as surrogate for trust to cope with Pro4. Thereby, once the violation of a property has been spotted by the inference engine, the designer should analyze the context and the requirements demanded by stakeholders and apply the appropriate pattern. The analysis of our scenario reveals several violations of Pro4. Among them, the designer may recognize situations where the application of the non-repudiation pattern is necessary. This is the case, for example, when the HCC appoints Dr. Charlie and Alison to provide a certain service.

The same framework can be used by security and legal experts for the validation of patterns. In the proposed approach, the solution is encoded as a SI* model. This allows them to use the formal framework underlying SI* for verifying if the solution meets the requirement addressed by the pattern. Moreover, it is possible to identify the consequences of applying the pattern by analyzing the solution itself. Specifically, the framework verifies if the properties encoding requirements are not satisfied by the solution. This allows the identification of the relations among patterns by looking at which properties are not satisfied and browsing the pattern library for an appropriate pattern. For instance, we have discovered that in a particular configuration of the scenario the application of the outsourcing pattern requires the application of the non-repudiation pattern.

8 Conclusions

System designers usually are not security experts or legal experts so that they may have difficulties in deploying systems that comply with security and privacy requirements as defined in the current legislation. We address this issue by presenting an approach based on security and privacy patterns. The patterns proposed in this paper have been derived from a real case-study based on a Smart Items Infrastructures for health care systems and are part of a pattern library developed in the EU SERENITY project. The pattern library is composed of security patterns at different abstraction levels. For example, in [?,?] security patterns for Web Service, Workflow and Sensors have been proposed. In this paper, our focus is the definition of security patterns at the organizational level that intend to make the system compliant with legal requirements. Together with patterns, the paper also presents a formal framework that assists pattern designers in the validation of patterns and system designers in their application by supporting the analysis of organizational security and privacy requirements.

The framework presented in this paper is supported by the S&D Tropos Tool.[9] This tool is an Eclipse plugin developed to model and analyze security, privacy and

[9] The tool is available at http://sesa.dit.unitn.it/sistar_tool/

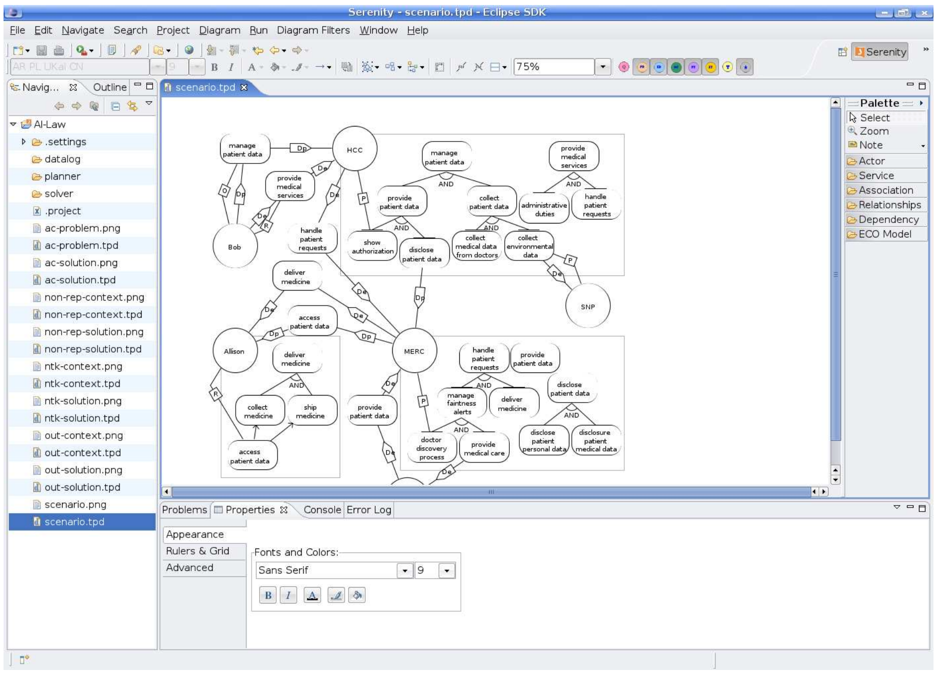


Fig. 10 S&D Tropos Tool

dependability requirements of socio-technical systems. The tool provides users with a graphical interface that allows them to draw SI* models (see Fig. 10). It also supports the automatic transformation of SI* graphical model into formal specifications expressed in ASP and provides a front-end to external ASP inference engines for tool supported requirements analysis.

Although formal verification is necessary to guarantee a high evaluation assurance level, the Common Criteria also requires testing to validate the solutions. In addition to proof-of-concept validation, the patterns in the SERENITY library have been validated by their use in real systems. However, the decision on the infrastructure used to implement the pattern depends on the specific context in which the pattern is deployed. For instance, the access control pattern proposed in the paper has been implemented using two different IT infrastructures. In a Service Oriented Architecture, the pattern has been implemented in XACML [?]. We have also implemented the access control pattern in a sensor-based architecture using encryption functions [?]. Recently, we finalized the development process covering an authorization pattern from conceptual level until implementation.[10] The demonstration describes the security engineering approach presented in this paper to address security and dependability issues concerning healthcare-related services in smart homes.

Currently, we are extending the pattern library by considering other scenarios (e.g., e-Business, e-Government, and Air Traffic Management) to (1) deploy general patterns

[10] A detailed walkthrough and demonstration are accepted to be presented at Information and Communication Technologies ICT 2008. Online description is available at `http://ec.europa.eu/information_society/events/cf/item-display.cfm?id=171`

that can be applied in different domains and (2) deploy patterns that cope with others security and privacy issues than the ones discussed for the scenario here presented.

We are also defining a pattern integration schema that will drive designers in the application of safe combinations of patterns where potential interferences and conflicts must be considered. The need of an integration schema is due to the fact that regulation compliance usually demand for the application of several security and privacy patterns to a particular context. For this purpose, we are developing a Prolog prototype based on Constraint Handling Rules (CHR) to check the consistency of pattern integration. In this setting, every security pattern (i.e., context, solution, requirement, and consequences) together with the union and intersection between their solutions and contexts are transformed into rules, and consistency among patterns are automatically verified before integration.

The security patterns defined in this paper intend to understand how the organizational setting should be modified to meet the legal requirements. Part of our on going work concerns the analysis of threats as a cross-cutting concerns for pattern validation. We are investigating how threats can compromise the system and determine countermeasures to mitigate the risks in organizational settings [**?**].

Finally, we are elaborating a tool supported methodology for extracting security- and privacy-related legal requirements from specification expressed in natural language. This work intends to address the first step of the process in Fig. 1. Specifically, we want to support legal experts in the generation of SI* models from textual description of security and privacy issues and the corresponding solutions.

Acknowledgements This work was partially funded by the projects IST-FP6-IP-SERENITY, IST-FP6-FET-IP-SENSORIA, and FIRB-TOCAI. We want to express our gratitude to all members of the SERENITY project for their feedback and useful scientific discussions.

References

1. Anderson, R.J.: Why cryptosystems fail. CACM **37**(11), 32–40 (1994)
2. Asnar, Y., Bonato, R., Giorgini, P., Massacci, F., Meduri, V., Riccucci, C., Saidane, A.: Secure and Dependable Patterns in Organizations: An Empirical Approach. In: Proc. of RE'07, pp. 287–292. IEEE Press (2007)
3. Asnar, Y., Moretti, R., Sebastianis, M., Zannone, N.: Risk as Dependability Metrics for the Evaluation of Business Solutions: A Model-driven Approach. In: Proc. of ARES'08, pp. 1240–1247. IEEE Press (2008)
4. Basin, D., Doser, J., Lodderstedt, T.: Model Driven Security: from UML Models to Access Control Infrastructures. TOSEM **15**(1), 39–91 (2006)
5. Bench-Capon, T.J.M., Robinson, G.O., Routen, T.W., Sergot, M.J.: Logic programming for large scale applications in law: A formalisation of supplementary benefit legislation. In: Proc. of ICAIL'87, pp. 190–198. ACM Press (1987)
6. Bench-Capon, T.J.M., Sartor, G.: A model of legal reasoning with cases incorporating theories and values. Artif. Intell. **150**(1-2), 97–143 (2003)
7. Breaux, T.D., Antón, A.I.: Analyzing regulatory rules for privacy and security requirements. TSE **34**(1), 5–20 (2008)
8. Breu, R., Popp, G., Alam, M.: Model based development of access policies. STTT **9**, 457–470 (2007)
9. Compagna, L., El Khoury, P., Massacci, F., Thomas, R., Zannone, N.: How to capture, communicate, model, and verify the knowledge of legal, security, and privacy experts: a pattern-based approach. In: Proc. of ICAIL'07, pp. 149–154. ACM Press (2007)
10. Cuevas, A., El Khoury, P., Gomez, L., Laube, A.: Security Patterns for Capturing Encryption-Based Access Control to Sensor Data. In: Proc. of SECURWARE'08, pp. 62–67. IEEE Press (2008)

11. Dibbern, J., Goles, T., Hirschheim, R., Jayatilaka, B.: Information Systems Outsourcing: A Survey and Analysis of the Literature. The DATA BASE for Advances in Information Systems **35**(4), 6–102 (2004)
12. Dijkstra, P., Prakken, H., de Vey Mestdagh, K.: An implementation of norm-based agent negotiation. In: Proceedings of the 11th international conference on Artificial intelligence and law, pp. 167–175. ACM (2007)
13. European Commission: Directive 95/46/ec on the protection of individuals with regard to the processing of personal data and on the free movement of such data. Official Journal of the European Communities **281**, 31 (1995)
14. European Commission: Communication on the follow-up of the work programme for a better implementation of the data protection directive. http://ec.europa.eu/justice_home/fsj/privacy/docs/lawreport/com_2007_87_f_en.pdf (2007)
15. Fernández, E.B., Ballesteros, J., Desouza-Doucet, A.C., Larrondo-Petrie, M.M.: Security Patterns for Physical Access Control Systems. In: Proc. of DBSec'07, LNCS 4602, pp. 259–274. Springer (2007)
16. Fernández, E.B., Pan, R.: A Pattern Language for Security Models. In: In Proc. of PLoP'01 (2001)
17. Fernández, E.B., Yuan, X.: Semantic Analysis Patterns. In: Proc. of ER'00, LNCS 1920, pp. 183–195. Springer (2000)
18. Fernández, E.B., Yuan, X.: Securing analysis patterns. In: Proc. of ACM Southeast Regional Conference, pp. 288–293. ACM Press (2007)
19. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley (1994)
20. Giorgini, P., Massacci, F., Zannone, N.: Security and Trust Requirements Engineering. In: FOSAD 2004/2005, LNCS 3655, pp. 237–272. Springer-Verlag (2005)
21. Guarda, P., Zannone, N.: Towards the Development of Privacy-Aware Systems. Information and Software Technology (2008). To appear.
22. Hofeld, W.N.: Fundamental legal conceptions as applied to judicial reasoning. Yale Law Journal **23**, 16–59 (1913)
23. IBM: Introduction to Business Security Patterns. IBM White Paper (2003)
24. ISO: Quality Management Systems: Requirements. ISO 9001:2000 (2000)
25. ISO/IEC: Code of practice for information security management. ISO/IEC 17799:2005 (2005)
26. ISO/IEC: Information technology-Security techniques-evaluation criteria for IT. ISO/IEC 15408:2005 (2005)
27. Kanger, S.: Law and logic. Theoria **38**(3), 105–132 (1972)
28. Kienzle, D.M., Elder, M.C.: Security Patterns for Web Application Development. Final Technical Report, University of Virginia (2002). Available at http://www.scrypt.net/˜celer/securitypatterns/final%20report.pdf
29. Kowalski, R.A., Sergot, M.J.: Computer Representation of the Law. In: Proc. of IJCAI'05, pp. 1269–1270. Morgan Kaufmann (1985)
30. Lamport, L.: How to write a long formula. Formal Aspects of Comp. **6**(5), 580–584 (1994)
31. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV System for Knowledge Representation and Reasoning. TOCL **7**(3), 499–562 (2006)
32. Mally, E.: Grundgesetze des Sollens: Elemente der Logik des Willens. Graz: Leuschner & Lubensky (1926)
33. Massacci, F., Mylopoulos, J., Zannone, N.: An Ontology for Secure Socio-Technical Systems. In: Handbook of Ontologies for Business Interaction, pp. 188–207. The IDEA Group (2007)
34. Massacci, F., Prest, M., Zannone, N.: Using a Security Requirements Engineering Methodology in Practice: The compliance with the Italian Data Protection Legislation. CSI **27**(5), 445–455 (2005)
35. Massacci, F., Zannone, N.: A Model-Driven Approach for the Specification and Analysis of Access Control Policies. In: Proc. of IS'08 (2008)
36. Meyer, J.J.C., Wieringa, R.J. (eds.): Deontic logic in computer science: normative system specification. John Wiley & Sons, Inc. (1994)
37. Mouratidis, H., Weiss, M., Giorgini, P.: Security Patterns Meet Agent Oriented Software Engineering: A Complementary Solution for Developing Secure Information Systems. In: In Proc. of ER'05, LNCS 3716, pp. 225–240. Springer (2005)
38. Room, S.: Data Protection & Compliance in Context. BCS (2007)

39. Saltzer, J.H., Schroeder, M.D.: The Protection of Information in Computer Systems. Proceedings of the IEEE **63**(9), 1278–1308 (1975)
40. Samarati, P., di Vimercati, S.D.C.: Access Control: Policies, Models, and Mechanisms. In: FOSAD 2001/2002, LNCS 2946, pp. 137–196. Springer (2001)
41. Sanchez-Cid, F., Muñoz, A., El Khoury, P., Compagna, L.: XACML as a Security and Dependability (S&D) pattern for Access Control in AmI environments. In: Proc. of AmI.d07, pp. 143–155. Springer (2007)
42. Schumacher, M.: Security Engineering with Patterns: Origins, Theoretical Models, and New Applications. Springer-Verlag (2003)
43. von Wright, G.H.: Deontic logic. Mind **60**, 1–15 (1951)
44. Wahlgren, P.: Automation of Legal Reasoning: A Study on Artificial Intelligence. Kluwer Law and Taxation Publishers (1992)
45. World Health Organization: A Declaration on the Promotion of Patients' Rights in Europe. http://www.who.int/genomics/public/eu_declaration1994.pdf (1994)
46. Yoder, J., Barcalow, J.: Architectural Patterns for Enabling Application Security. In: Proc. of PLoP'97 (1997)
47. Yoshioka, N., Honiden, S., Finkelstein, A.: Security Patterns: A Method for Constructing Secure and Efficient Inter-Company Coordination Systems. In: Proc. of EDOC'04, pp. 84–97. IEEE Press (2004)
48. Zeni, N., Kiyavitskaya, N., Cordy, J.R., Mich, L., Mylopoulos, J.: Annotating regulations using cerno: An application to italian documents - extended abstract. In: Proc. of ARES'08, pp. 1437–1442. IEEE Press (2008)