Protecting privacy in system design: the electronic voting case

Evangelia Kavakli
Cultural Informatics Laboratory,
Department of Cultural Technology and Communication,
University of the Aegean, Mytilene, Greece

Stefanos Gritzalis
Information and Communication Systems Security Laboratory,
Department of Information and Communications Systems Engineering,
University of the Aegean, Samos, Greece, and

Kalloniatis Christos
Cultural Informatics Laboratory,
Department of Cultural Technology and Communication,
University of the Aegean, Mytilene, Greece

Abstract
Purpose – The purpose of the paper is to present Privacy Safeguard (PriS) a formal security requirements engineering methodology which, incorporates privacy requirements in the system design process and to demonstrate its applicability in an e-voting case.

Design/methodology/approach – PriS provides a methodological framework for addressing privacy-related issues during system development. It provides a set of concepts for formally expressing privacy requirements (authentication, authorisation, identification, data protection, anonymity, pseudonymity, unlinkability and unobservability) and a systematic way-of-working for translating these requirements into system models. The main activities of the PriS way-of-working are: elicit privacy-related goals, analyse the impact of privacy goals on processes, model affected processes using privacy process patterns and identify the technique(s) that best support/implement the above-process patterns.

Findings – Analysis of a number of well known privacy-enhancing technologies, as well as of existing security requirement engineering methodologies, pinpoints the gap between system design methodologies and technological solutions. To this end, PriS provides an integrated approach for matching privacy-related requirements to proper implementation techniques. Experimentation with the e-voting case suggests that PriS has a high degree of applicability on internet systems that wish to provide services that ensure users privacy, such as anonymous browsing, untraceable transactions, etc.

Originality/value – The paper proposes a new methodology for addressing privacy requirements during the design process. Instead of prescribing a single solution, PriS guides developers to choose the most appropriate implementation techniques for realizing the identified privacy issues. In addition, due to its formal definition it facilitates control of the accuracy and precision of the results and enables the development of automated tools for assisting its application.

Keywords Privacy, Data security, Automation

Paper type Research paper

1. Introduction
Privacy as a social and legal issue, traditionally, has been the concern of social scientists, philosophers and lawyers. However, the extended use of electronic
government applications in the context of basic public services (e.g. health care, social security, taxation, etc.), sets additional technology-related requirements for protecting the electronic privacy of individuals.

Most of today’s e-services are relying on stored data, identifying the customer, his preferences and previous record of transactions. However, combining such data will in many cases constitute an invasion of privacy. Protecting privacy is especially important in e-government applications, since the greater collection and storage of personal data by government, have the potential to decrease the level of democratic trust.

Privacy-related issues are many and varied, as privacy itself is a multifaceted concept. Privacy comes in many forms, relating to what it is that one wishes to keep private. Review of current research, highlights the path for user privacy protection in terms of eight privacy requirements namely identification, authentication, authorization, data protection, anonymity, pseudonymity, unlinkability and unobservability (Fischer-Hübner, 2001; Cannon, 2004; Koorn et al., 2004). The first three requirements are basically security requirements but they are included due to their key role in the privacy protection. Addressing these requirements is of vital importance when one aims to minimize or eliminate the collection of user identifiable data. The need for a methodology that will address these privacy issues is immense.

This paper describes Privacy Safeguard (PriS), a methodology for incorporating privacy requirements into the system design process. PriS models privacy requirements in terms of organisational goals and uses the concept of privacy process patterns for describing the impact of privacy goals onto the organisational processes and the associated software systems supporting these processes. In addition, Formal PriS provides a formal definition of the PriS way-of-working, i.e. it formally defines the processes of:

- analysing the impact of privacy requirement on organisational goals, subgoals and process; and
- suggesting of appropriate system implementation technique(s) for realising these requirements.

The paper is structured as follows. Section 2 provides a brief overview of related work. Section 3 presents the PriS conceptual framework and way of working. Also, it introduces privacy process patterns and explains how they can be used in order to identify appropriate privacy implementation techniques. Formal PriS is presented in Section 4. The applicability of the methodology is demonstrated in an e-voting project, in Section 5. Finally, Section 6 concludes with pointers to future work.

2. Related work
A number of requirement engineering methodologies have been proposed for managing security issues in the design level. For better representing the related work being accomplished in this filed, existing methodologies are presented in the following paragraphs categorized by their common vulnerabilities. Also, a table is presented for clearly showing the drawbacks of the existing methodologies and the necessity for introducing a new methodology.

NFR (Chung, 1993; Mylopoulos et al., 1992), Tropos (Liu et al., 2003; Mouratidis et al., 2003a, b), KAOS (van Lamsweerde and Letier, 2000), i* (Liu et al., 2002), RBAC (He and Antón, 2003), M-N framework (Moffett and Nuseibeh, 2003), GBRA (Antón, 1996;
Antón and Earp, 2000) are a number of security requirements engineering methodologies. One common vulnerability of these methodologies is that they address privacy along with other security goals; however, as it is discussed in this paper, privacy should be considered as a separate design criterion in the whole system’s development process. Furthermore, the majority of the proposed methodologies (with the exception of GBRAM) consider the elicitation of security requirements from business goals but do not address how these requirements are translated into system components, nor do they suggest any relevant implementation techniques. The RBAC methodology is the only one, which considers the generation of system policies based on the elicited security requirements. However, it does not suggest any systematic way for eliciting and managing these requirements.

Bellotti and Sellen (1993) developed a framework for privacy aware design in ubiquitous computing. This framework proposes a procedure designers may follow through a set of questions in order to evaluate a system. The evaluation is accomplished by identifying a set of new requirements, which must be implemented by the developers. A recent variation of this framework is proposed by Hong et al. (2004). In spite of the fact, that these frameworks are inexpensive to use and not very time consuming, a number of common disadvantages exist. Firstly, they do not address/suggest any implementation techniques for realizing the identified requirements. A gap between design and implementation exists since they do not suggest a way for guiding the developer from the design to the implementation level. Also, these frameworks produce a static set of vulnerabilities (which the current system must overcome) and leave the designer to re-evaluate the entire system since they do not take iteration into account as part of the design process. Changing one part in the system’s design may affect multiple other parts in terms of privacy. Based on the afore-mentioned vulnerabilities these frameworks are more likely to be employed once at the end of the design cycle rather than become a part of the design process.

The STRAP framework proposed in Jensen et al. (2005) takes a further step compared to the previous frameworks. Specifically, it is based on the above frameworks while borrowing methods from requirements engineering and goal-oriented analysis. In particular, at the beginning STRAP performs a goal-oriented analysis of the system for identifying the relevant actors, goals and major system components. Then a list of vulnerabilities is produced by asking a number of questions similar to the ones proposed in Bellotti and Sellen (1993) and Hong et al. (2004) on every goal and sub-goal. Vulnerabilities are categorized based on the four Federal Information Practices presented in The Code of Fair Information Practices (1973). Once vulnerabilities are identified the steps of refinement, evaluation and iteration follow. While STRAP successfully combines goal-oriented analysis and heuristic-based frameworks for addressing privacy vulnerabilities, it does not take the next step of discovering/suggesting the relevant implementation techniques needed for eliminating these vulnerabilities. A detailed overview of these methodologies can be found in Kavakli et al. (2006). Table I presents the above-mentioned methodologies along with their drawbacks and their vulnerabilities.

From an implementation perspective, a number of privacy-enhancing technologies (PETs) have been developed for realising privacy. The PETs described below, focus on the software implementation alone, irrespective of the organisational context in which the system will be incorporated.
In other words, there is no obvious link between the organisational processes that are constrained by the privacy requirements and the supporting software systems. This lack of knowledge makes it difficult to determine which software solution best fits the organisational needs or to evaluate alternatives. Some examples of PETs follow providing also a brief description of their way of working. Anonymizer (Boyan, 1997) is a third-party web site, which acts as a middle layer between the user and the site to be visited providing user’s anonymity. Crowds is an agent that has been designed also for protecting user’s anonymity. It is based on the idea that people can be anonymous when they blend into the crowd (Reiter and Rubin, 1998, 1999). Onion Routing is a general-purpose infrastructure for private communications over a public network. It provides anonymous connections that are strongly resistant to both eavesdropping and traffic analysis (Reed et al., 1998; Goldschlag et al., 1999). Dining Cryptographers Network (DC-Net) proposed in Chaum (1985, 1988) allows participants to send and receive messages anonymously in an arbitrary network. It can be used for providing perfect sender anonymity. Mix-Networks is another technique introduced in Chaum (1981) and further discussed in Pfitzmann and Waidner (1987). It realises unlinkability of sender and recipient as well as sender anonymity against recipient and optionally recipient anonymity. Hordes is a protocol designed for utilising multicast communication for the reverse path of anonymous connections, achieving not only anonymity but also sender unlinkability and unobservability. A detailed description of Hordes is given in Shields and Levine (2000). GAP (GNUnet’s Anonymity Protocol) presented in Bennett and Grothoff (2003) achieves anonymous data transfers. However, GAP is customised to the functionality of a peer-to-peer network. Finally, Tor, presented in Dingledine et al. (2004), is an architecture based on the Onion Routing architecture with an improved way of working. A detailed overview of the above-mentioned technologies can be found in Kavakli et al. (2005) and Gritzalis (2004).

3. The PriS methodology

3.1 PriS conceptual framework

As mentioned above, PETs focus on the software implementation alone, irrespective of the organisational context in which the system will be incorporated understanding the relationship between the user needs in the organisational domain and the capabilities of the supporting software systems is of critical importance.
To this end, PriS methodology provides a set of concepts for modelling privacy requirements in the organisation domain and a systematic way-of-working for translating these requirements into system models. The conceptual model used in PriS is based on the enterprise knowledge development (EKD) framework (Loucopoulos and Kavakli, 1999; Loucopoulos, 2000), which is a systematic approach to developing and documenting organisational knowledge. This is achieved through the modelling of:

- organisational goals, that express the intentional objectives that control and govern its operation;
- the “physical” processes, that collaboratively operationalise organisational goals; and
- the software systems that support the above processes.

In this way, a connection between system purpose and system structure is established.

Based on this framework, PriS models privacy requirements as a special type of goal (privacy goals) which constraint the causal transformation of organisational goals into processes. From a methodological perspective, reasoning about privacy goals comprises of the following activities:

- **Elicit privacy-related goals.** The first step concerns the elicitation of the privacy goals that are relevant to the specific organisation. This task usually involves a number of stakeholders and decision makers (managers, policy makers, system developers, system users, etc). Therefore, elicitation of privacy goals is described the following activities: perform stakeholder analysis and organise stakeholder workshop; identify privacy issues; and agree on a structures set of privacy goals. Identifying privacy issues is guided by the basic privacy concerns identified in Section 1. The aim is to interpret the general-privacy requirements with respect to the specific application context into consideration.

- **Analyse the impact of privacy goals on organisational processes.** The second step is to analyse the impact of privacy goals on processes and related support systems. Answering this question involves the following tasks: identify the influence of privacy goals on organisational goals and analyse the impact on processes. A summary of this process is shown in the Figure 1. For each privacy goal, PriS identifies the impact it may have on other organisational goals. This impact may lead to the introduction of new goals or to the improvement/adaptation of existing goals. Introduction of new goals may lead to the introduction of new processes while improvement/adaptation of goals may lead to the adaptation of associated processes accordingly. Repeating this process for every privacy goal and its associated organisational goals leads to the identification of alternative ways for resolving privacy requirements. The result of this process modelled in the spirit of and extended AND/OR goal hierarchy.

- **Model affected processes using privacy process patterns.** Having identified the privacy-related processes, these are modelled based on the relevant privacy-process patterns. Privacy-process patterns are generalised process models, which include activities and flows connecting them, presenting how a business should be run in a specific domain Kalloniatis et al. (2007). In particular, PriS defines seven-process patterns corresponding to the seven basic
privacy requirements. Each pattern is described in two levels. The first level, shown in Figure 2, is more abstract and is the same for all privacy process patterns. It describes the general situation where a user is related to the system through a specific process “the privacy related process”. The system is usually a computer system but it could also be a person, a process or a service.

In the second level, the privacy-related process is analysed with respect to the specific privacy requirement constraining that process. For example, Figure 3 shows the process pattern for addressing the authentication requirement, which describes the relevant activities needed to realise that process.

As shown in Figure 3, every time a user submits a request to the system, the system should check that request and if authentication is needed the user should provide them with the proper authentication data or else a deny access.

**Figure 1.** Analyse the impact of privacy requirements on business processes

**Figure 2.** First level of pattern

**Figure 3.** Authentication pattern
Identify the technique(s) that best support/implement the above processes. The last step is to define the system architecture that best supports the privacy-related process identified in the previous step. Once again, through the pattern analysis, PriS is able to suggest the proper implementation technique(s) that best support/implement these processes.

In particular, every process pattern highlights the specific privacy-related activities that should be implemented thus, indicating to the developer where the privacy implementation technique needs to be introduced in order to ensure that the process is privacy compliant. Naturally, the choice of the appropriate implementation technique depends on the privacy requirement(s) under consideration.

Existing privacy implementation techniques can be classified in six categories, namely:

1. administrative tools;
2. information tools;
3. anonymiser products, services and architectures;
4. pseudonymiser tools;
5. track and evident erasers; and
6. encryption tools.

An overview of these categories can be found in Gritzalis (2004).

Each category includes a number of technologies-methodologies (Kavakli et al., 2006; Gritzalis, 2004). For example, administrative tools include: Identity management, Biometrics, Smart Cards, Permission Management and Monitoring and Audit tools. Anonymiser tools include: Browsing Pseudonyms, Virtual E-mail addresses, Surrogate Keys as well as a number of PETs such as Crowds, Onion Routing, Gap, Tor, etc.

Different tools in each category implement specific-privacy process patterns. The correspondence between privacy process patterns and implementation tools is shown in Figure 4. Using this table, a developer can choose for every process pattern which is/are the best implementation technique(s) among the ones available, always based on the privacy requirement(s) that needs to be realised, as well as the specific business domain in which it will be implemented.

<table>
<thead>
<tr>
<th>Administrative Tools</th>
<th>Information Tools</th>
<th>Anonymizer Products, Services and Architectures</th>
<th>Pseudonymizer Tools</th>
<th>Track and Evident Erasers</th>
<th>Encryption Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity Management</td>
<td>Biometrics</td>
<td>Privacy Compliance</td>
<td>Scanning</td>
<td>CBM Personalization</td>
<td>Application Data Management</td>
</tr>
<tr>
<td></td>
<td>Smart Cards</td>
<td>Monitoring</td>
<td>Browsing Pseudonyms</td>
<td>SDF Deletion</td>
<td>Browser Cleaning Tool</td>
</tr>
<tr>
<td></td>
<td>Biometrics</td>
<td>Privacy Policy Generators</td>
<td>Virtual Email Addresses</td>
<td>Trash Eater</td>
<td>Activity Traces Eraser</td>
</tr>
<tr>
<td></td>
<td>Privacy Policy</td>
<td>Privacy Compliance</td>
<td>Trusted Third Parties</td>
<td>Harddisk Data Eraser</td>
<td>Encrypting Email</td>
</tr>
<tr>
<td></td>
<td>Readers</td>
<td></td>
<td>Surrogate Keys</td>
<td>Encrypting Documents</td>
<td>Encrypting Transactions</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>Crowds</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>Onion Routing</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>DC-Nets</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>Mix-Net</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>Hordes</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>GAP</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>Tor</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
<tr>
<td></td>
<td>Privacy Compliance</td>
<td></td>
<td>CRM Personalization</td>
<td>Encrypting Documents</td>
<td>Encrypting Documents</td>
</tr>
</tbody>
</table>

| Authentication | X X X X X |
| Identification | X X X X X |
| Data Protection| X X X X X |
| Anonymity and/or pseudonymity | X X X | X X X X X X X X X X X X X X X X X X |
| Unlinkability | X X X | X X X X X X X X X X X X X X X X X X X X |
| Unobservability| X X X | X X X X X X X X X X X X X X X X X X X X |

Figure 4. Matching privacy patterns with implementation techniques
It should be mentioned that alternative system implementation architectures may be used depending on the privacy requirement that one wishes to achieve. Therefore, instead of prescribing a single solution PriS identifies and suggests a number of implementation techniques and architectures that best support the realisation of each privacy-related process in the system’s development phase. The developer is then responsible for choosing which architecture is best for the developing system based on organisation’s priorities such as, cost, systems efficiency, etc.

4. Formal PriS
This section provides a formal definition of the PriS way of working. Formal PriS aims to provide consistent, unambiguous and precise representations of all PriS concepts as well as to provide the basis for useful tool support for PriS activities. The following sections formally describe the 4 PriS activities described in Section 3.

4.1 Elicit privacy related-goals (goal-level)
Let us start with the formal definition of the PriS goal model.

**Definition 1.** Every goal belongs to a set of goals called G set. G contains every system’s goal and subgoal as it is defined from the stakeholders’ analysis based on the functionalities which the system should accomplish.

\[ G = \{G_1, G_2, G_3, \ldots, G_{n-1}, G_n\}. \]

In every goal, hierarchy abstract goals are further decomposed into more specific goals until reaching the level of operationalised goals which do not need further decomposition thus being able to be realised from one or more processes.

For realising the connections between every goal and its immediate subgoals the terms parent goal and child goal are defined. Every goal in the goal hierarchy has a parent goal and a number of child goals.

**Definition 2.** The parent goal of a goal \( G_i \) is the goal in the hierarchy where the \( G_i \) is derived from. The parent goal is always at a higher level that the \( G_i \) in the hierarchy.

**Definition 3.** A child goal(s) of a goal \( G_i \) is one of the goals used for the decomposition of the \( G_i \). Child goal is always at a lower level in the hierarchy.

Usually, every goal \( G_i \) is decomposed into several subgoals. Thus, the definition for every goal regarding its position in the goal hierarchy will be as follows:

**Definition 4.** Every goal \( G_i \) belonging in the G set, has a parent goal and a number of child goals.

\[ G_i = \{G_{\text{parent}}, G_{\text{child1}}, G_{\text{child2}}, \ldots, G_{\text{childk-1}}, G_{\text{childk}}\}. \]

where \( k \) the number of child goals for the goal \( G_i \).

However, in Definition 4, two exceptions exist.

**Exception 1.** When a goal has no parent goal (usually the more abstract goal in the hierarchy) the \( G_{\text{parent}} \) is equal to 0 (zero).

**Exception 2:** When a goal has no child goal(s) (usually the operationalised goals which do not need further decomposition) the \( G_{\text{child}} \) attributes are equal to 0 (zero).

For the proper representation of the goal hierarchy in the formal model, the first goal (more abstract goal) as well as the operationalised goals (goals that do not need further decomposition) need to be identified since in the first goal no parent goals
should be assigned and in the operationalised goals no child goals can be assigned either. Exceptions one and two solve this problem.

The next step is to define which of the goals in the G set are affected by which privacy requirement(s). To achieve this, the seven privacy requirements are represented by specific variables which can take only two values, 0 and 1.

The aim is to apply these variables on every privacy affected goal by assigning seven values, respectively. The variables which have the value 1 represent the privacy requirements that affect the specific goal. In Table II, the variables’ names of each privacy requirement are shown.

Definition 5. Every privacy requirement is expressed by a variable which can only accept two values, 0 and 1. Every goal $G_i$ is assigned by seven values which represent which privacy requirements affect the specific goal and which not. If $G_i$ is not an end goal (no child goals) then the privacy requirements that affect goal $G_i$ also affect all $G_{i,child}$ goals of $G_i$.

$$G_i = \{PV1, PV2, PV3, PV4, PV5, PV6, PV7\}$$

For realising the goal hierarchy in formal PriS, a number of procedures and functions need to be defined.

(1) $\text{Assign\_hier} \left(G_i\right) = (G_{\text{parent}}, G_{\text{child1}}, \ldots, G_{\text{childk-1}}, G_{\text{childk}})$

The Assign_hier procedure assigns the parent and child goals of goal $G_i$.

(2) $\text{Assign\_pv}(G_i) = (PV1, PV2, PV3, PV4, PV5, PV6, PV7)$

The Assign_pv procedure assigns seven values to the respective privacy variables for goal $G_i$ regarding which privacy requirements affect the specific goal. For example, if goal $G_i$ needs to realise authentication and unlinkability requirements then the Assign_pv function will be as follows:

$$\text{Assign\_pv}(G_i) = (1, 0, 0, 0, 1, 0)$$

(3) $\text{Read\_pv} \left(G_i\right)$ function.

The specific function receives as input the name of one goal from the goal hierarchy, e.g., $G_i$ and returns the values of the seven privacy variables this goal has been assigned to. Through function Read_pv it is always known, which are the privacy requirements that affect every privacy-related goal in the goal hierarchy.

<table>
<thead>
<tr>
<th>Privacy variable</th>
<th>Expressed privacy requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>PV1</td>
<td>Authentication</td>
</tr>
<tr>
<td>PV2</td>
<td>Authorisation</td>
</tr>
<tr>
<td>PV3</td>
<td>Identification</td>
</tr>
<tr>
<td>PV4</td>
<td>Data protection</td>
</tr>
<tr>
<td>PV5</td>
<td>Anonymity and pseudonymity</td>
</tr>
<tr>
<td>PV6</td>
<td>Unlinkability</td>
</tr>
<tr>
<td>PV7</td>
<td>Unobservability</td>
</tr>
</tbody>
</table>

Table II. Privacy variables representing every privacy requirement.
Based on the above example, the result of the function $\text{Read_pv}(G_i)$ would be $(1,0,0,0,1,0)$. Based on the table above, it is derived that the specific goal is a privacy-related goal and the privacy requirements that affect it are authentication and unlinkability.

4.2 Analyse the impact on organizational processes (process-level)

Processes realise the operationalised goals defined in the previous step. Every process may realise more than one operationalised goals.

Definition 6. Every process belongs to a set of processes the $P$ set. $P$ contains all the processes identified that realise the system’s operationalised goals.

$$P = \{P_1, P_2, P_3, \ldots, P_{k-1}, P_k\}$$

The first step is to identify and create a link between the privacy related operationalised goals and the respective processes that realise these goals. For achieving that, the procedure $\text{Match_G}_P(G_i) = (P_s)$ is introduced. This procedure receives as parameters the privacy related operationalised goal and is assigned by the process among which a link is created for declaring that the specific goal is realised by the specific process. At the end of this step, two tasks are accomplished. The identification of privacy-related processes and the creation of the links between the privacy-related operationalised goals and these processes.

After the identification of privacy-related processes, PriS identifies which privacy process patterns need to be applied not only for modelling these processes but also for relating them with the proper implementation techniques.

For accomplishing this task, a table is created which maps every process pattern with a specific variable (Table III). Process pattern variables share the same logic like privacy requirement variables but their aim and way of working is different as it is derived from the following description.

Every process is assigned by seven values which are the values of the seven process pattern variables. On every process pattern variable, two values can be assigned. 0 and 1. 0 indicates that the respective process pattern will not be applied on the specific process while value 1 indicates the opposite.

Thus the following definition occurs:

<table>
<thead>
<tr>
<th>Privacy process pattern variable</th>
<th>Expressed privacy process pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>PP1</td>
<td>Authentication pattern</td>
</tr>
<tr>
<td>PP2</td>
<td>Authorisation pattern</td>
</tr>
<tr>
<td>PP3</td>
<td>Identification pattern</td>
</tr>
<tr>
<td>PP4</td>
<td>Data protection pattern</td>
</tr>
<tr>
<td>PP5</td>
<td>Anonymity and pseudonymity pattern</td>
</tr>
<tr>
<td>PP6</td>
<td>Unlinkability pattern</td>
</tr>
<tr>
<td>PP7</td>
<td>Unobservability pattern</td>
</tr>
</tbody>
</table>

Table III. Privacy process pattern variables representing every privacy process pattern
Thus, the following definition occurs:

**Definition 7.** Every process $P_i$ is assigned by seven values which represent which privacy process patterns are applied on the specific process and which not.

$$P_i = \{PP1, PP2, PP3, PP4, PP5, PP6, PP7\}$$

### 4.3 Model affected processes using privacy process patterns (process-level)

The selection of the privacy process patterns that will be assigned on every process is based on the privacy requirements that constraint the operationalised goal which is realised by the specific process. The privacy requirements which constraint every goal can be identified from the respective values of the privacy requirement variables of each goal as it was mentioned before.

Despite the fact that the values of every goal’s privacy requirements’ variables are assigned as one set, a classification among these variables exists. Specifically, the first four privacy requirements are related with identification issues, while the last three has to do with anonymity issues. In other words, the first four requirements focus on protecting privacy by identifying each subject and granting privileges regarding the rights of this subject to the data that it tries to access, while the last three requirements focus on protecting the privacy of each subject by ensuring its anonymity or by preserving the reveal of its personal data by malicious third parties.

Based on this classification, the seven privacy requirements’ values of every operationalised subgoal are examined separately and different rules exist when selecting the proper privacy process patterns.

Specifically, for the first four privacy requirements, the selection of the privacy process pattern is based on the last requirement in the sequence that has the value of 1. As it can be derived from the privacy process pattern description, the respective patterns of the first four privacy requirements (authentication, authorization, identification and data protection) are interrelated in a sense that the next in the sequence “contains” also the previous ones. Thus, when a goal, for example, is assigned to achieve authentication and data protection requirements then the process pattern which will be assigned on the respective process that realise this goal will be only data protection pattern because data protection is later in the sequence meaning that authentication is already “covered” by this pattern. The specific example is expressed formally as follows:

$$G_i = \{1, 0, 0, 1, 0, 0, 0\} \quad \text{The privacy requirements of goal } G_i$$

$$\text{Match}_G_P(G_i) = \{P_k\} \quad \text{Process } P_k \text{ realises goal } G_i$$

$$P_k = \{0, 0, 0, 1, 0, 0, 0\} \quad \text{Privacy process patterns assigned to process } P_k$$

The following Table IV presents which privacy process patterns are applied on every process depending on the values of the operationalised goal’s first four privacy requirement variables.

As it can be seen on the table above, the process pattern that is applied on every process, regarding the first four privacy requirements is always the latest in the sequence where the respective requirement has a value of 1.
The selection of the appropriate process pattern for the last three requirements (anonymity and pseudonymity, unlinkability and unobservability) is totally related on the values of the respective privacy requirements of the operationalised goal. Specifically, when a privacy requirement variable assigned to a goal has a value of one then the respective process pattern is applied on the relevant process which realises that operationalised goal.

However, four cases exist where the selection of a process pattern is uniquely determined and does not follow the above-mentioned rule for these three requirements. These cases are shown in Table V.

PriS combines the above cases and rules and returns as a result the values of the seven process pattern variables for every privacy related process. Thus, for example, if an operationalised goal $G_i$ has the following seven values $(0,1,0,1,0,1,0)$ as the values of the privacy requirements’ variables, PriS identifies that in the process $P_k$ that realises goal $G_i$ the respective process patterns that will be applied are data protection and unlinkability.

As it was mentioned before, every process may realise more than one operationalised goals. In this case, before the selection of the proper process patterns that will be applied on the specific process, PriS identifies the maximum values between every privacy requirement variable of each subgoal and creates a virtual goal $G'$ that contains all seven maximum values. Thus, for example, if goal $G_i$ with values $(0,1,0,1,0,0,0)$ and goal $G_t$ with values $(1,1,0,1,1,0,1)$ are realised by the same process $P_k$, a new virtual goal $G'$ will be created with values $(1,1,0,1,1,0,1)$. The selection of the proper process patterns will be based on the seven values of the $G'$ goal. In this way, process $P_k$ will satisfy both goals, $G_i$ and $G_t$.

<table>
<thead>
<tr>
<th>PV1</th>
<th>PV2</th>
<th>PV3</th>
<th>PV4</th>
<th>Process pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Authentication</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Authorisation</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Authorisation</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Identification</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Identification</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Identification</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Identification</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Data protection</td>
</tr>
</tbody>
</table>

Table IV. Application of process patterns based on the first four privacy requirements variables

<table>
<thead>
<tr>
<th>PV5</th>
<th>PV6</th>
<th>PV7</th>
<th>Process pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Anonymity and pseudonymity</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Unlinkability</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Unobservability</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Unobservability</td>
</tr>
</tbody>
</table>

Table V. Application of process patterns based on the last three privacy requirements variables
Definition 8. \( \forall G_i \in G \), and are realised by process \( P_k \), a new goal \( G' \) is created and is defined as follows:

\[
G' = G_i \land G' \land \ldots \land G^k = [PV_i^j \land PV_j^i \land \ldots PV^k]
\]

where \( k \) – the number of operationalised goals realised by one process; \( l - 1, 2, \ldots 7 \) (seven privacy variables for every goal)

Based on the above definition, PriS takes the maximum value of every operationalised goal’s privacy variable and creates \( G' \) which constitutes of the maximum values of every privacy variable.

Beside the \( \text{Match}_G \left(P(G_i) = (P_k) \right) \) procedure a number of other processes and functions need to be defined in order to address the above mentioned way of working.

Procedure \( \text{Init}(P_k) \). Receives as input the name of the process and initializes (assigns the value of zero) the values of the process pattern variables for that process.

Function \( \text{Locate_pp}(G_i) \). This function receives as input the name of the privacy-related operationalised goal (or the \( G' \) if there are more than one goals) and returns which process patterns will be applied on the relevant process that realises \( G_i \) or \( G' \). The returned values are assigned to four variables namely \( \text{id} \), \( \text{an} \), \( \text{unlink} \), \( \text{unob} \). Below an analysis of these variables is presented.

Function \( \text{Get_pr}(i, \text{Read_pv}(G_i)) \). This function returns the value of the \( i \)-th privacy requirement variable of goal \( G_i \).

Procedure \( \text{Assign_pp}(P_k, \text{id}, \text{an}, \text{unlink}, \text{unob}) \). This procedure receives as input a process name and the four variables which were taken as a result from the function \( \text{Locate_pp()} \) and applies the selected process patterns on the respective process \( (P_k) \).

In order to better understand the function \( \text{Locate_pp}(G_i) \), its implementation is described below.

\[
\begin{align*}
\text{Locate_pp}(G_i) \\
\text{an} &= 0 \\
\text{id} &= 0 \\
\text{unlink} &= 0 \\
\text{unob} &= 0 \\
\text{for } i &= 1 \text{ to } 4 \\
& \quad \text{if Get_pr}(i, \text{Read_pv}(G_i)) = 1 \text{ then id} = i \\
& \quad \text{end for} \\
& \quad \text{if Get_pr}(5, \text{Read_pv}(G_i)) = 1 \text{ then an} = 1 \\
& \quad \text{if Get_pr}(6, \text{Read_pv}(G_i)) = 1 \text{ and Get_pr}(7, \text{Read_pv}(G_i)) = 1 \text{ then} \\
& \quad \text{unlink} = 0 \\
& \quad \text{unob} = 1 \\
& \quad \text{else} \\
& \quad \text{unlink} = \text{Get_pr}(6, \text{Read_pv}(G_i)) \\
& \quad \text{unob} = \text{Get_pr}(7, \text{Read_pv}(G_i))
\end{align*}
\]

The function \( \text{Locate_pp()} \) uses the procedure \( \text{Get_pr} \) for reading the relevant value of every privacy requirement assigned to goal \( G_i \) and acts accordingly to the rules that were described before. The variable \( \text{id} \) can be assigned by one of the four values (one to four) regarding the last privacy requirement which has the value of one. As it was mentioned before the process pattern that will be selected, regarding the first four
privacy requirements, depends on the last requirement in the sequence that has the value of one. The variable \( an \) is assigned by the value one only when anonymity and pseudonymity requirement has an impact on goal \( G_i \), meaning that the respective requirement variable will also have been assigned with the value one. The same implies for the variables unlink and unob. However, as it is mentioned on Table V, an exception exists where unlinkability and unobservability requirements variables are equal to one. In that case, only unobservability pattern is applied. Function Locate_pp() considers the above rules and assigns the right values to the respective variables for the selection of the privacy-related process patterns.

The procedure Assign_pp\((G_i)\) is applied on every privacy-related process and is implemented as follows:

\[
\text{Assign}_\text{pp}(P_k,\text{id},an,\text{unlink},\text{unob})
\]

\[
\text{Init } (P_k)
\]

\[
P_k(\text{id}) = 1
\]

\[
P_k(5) = an
\]

\[
P_k(6) = \text{unlink}
\]

\[
P_k(7) = \text{unob}
\]

It is obvious that the procedure Assign_pp follows the execution of the Locate_pp(). The aim of Assign_pp is to assign the correct values to the corresponding privacy process pattern variables of every privacy related process \( (P_k) \), thus addressing which of the seven process patterns will be applied and which not. Specifically, at the beginning, the seven process pattern variables are initialized through the Init procedure. Then, the \( id \)-th variable in the sequence receives the value of one indicating that this was the last privacy requirement (for the first four) that has an impact on the operationalised goal, and based on the previous description the respective pattern is the only one that needs to be applied. In the 5th, 6th and 7th process pattern variable the values of an, unlink and unob are assigned based on the outcome of the Locate_pp() function.

4.4 Identify the technique(s) that best support/implement privacy-related processes

Having decided which of the privacy-process patterns will be applied on every privacy-related process, the last step is to identify and suggest to the developer a number of implementation techniques that realise each pattern, based on Figure 4 which matches every process pattern with the respective technologies that realise it.

Definition 9. Every implementation technique belongs to a set of implementation techniques, the IT set.

\[
\text{IT} = \{IT_1, IT_2, IT_3, \ldots, IT_{j-1}, IT_j\}
\]

For describing which implementation techniques realise which patterns, PriS assigns seven values to every technique following the same logic as before. Specifically, every implementation technique is assigned by seven values, which represent which process patterns it realises according to Figure 4. Thus, for the Tor architecture, for example, the seven values that will describe it will be \((0,0,0,1,1,1)\) meaning that the specific architecture can implement the anonymity and pseudonymity process pattern, the unlinkability and the unobservability process pattern. For every implementation technique, the following definition occurs:
**Definition 10.** Every implementation technique $IT_i$ is assigned by seven values which represent which privacy process patterns are realised by it and which are not.

$$IT_i = \{\text{auth, author, ident, dat_prot, anon_pseud, unlinkab, unobserv}\}$$

PriS checks the privacy-process patterns that are applied on every process and for every pattern, it suggests a number of implementation techniques according to their respective values. PriS can either suggest separately for every process pattern a number of implementation techniques or can suggest a number of techniques for all the identified process patterns. In the case where the combination of process patterns does not lead to specific implementation techniques, PriS suggests the techniques that realise most of the privacy-process patterns. It should be mentioned that PriS does not choose the best technique out of the suggested ones. This is done by the developer who has to consider other factors like cost, complexity, etc. PriS guides the developer by suggesting a number of implementation techniques that satisfy the realisation of the privacy process patterns identified in the previous step.

5. Applying PriS in e-voting

This section demonstrates the application of the PriS methodology on an electronic voting project. A detailed description of the e-voting project can be found in EU-Information Society DG, IST Programme 2000#29518 (2000).

5.1 The electronic voting system

The main scope of the e-voting system is to provide eligible citizens the right to cast a vote over the internet rather than visiting an election district, aiming to simplify the election processes thus increasing the degree of citizens' participation during elections.

The e-voting system is described by four main principles that form the four primary organisational goals namely:

1. generality;
2. equality;
3. freedom; and
4. directness.

Specifically, generality implies that all citizens above a certain age should have the right to participate in the election process. Equality implies that both political parties – that participate in the election process – and voters have equal rights before, during and after the election process and neither the system nor any other third party is able to alternate this issue. Freedom implies that the entire election process is conducted without any violence, coercion, pressure, manipulative interference or other influences, exercised either by the state or by one or more individuals. Finally, directness implies that no intermediaries chime in the voting procedure and that each and every ballot is directly recorded and counted.

Based on the four primary goals of the e-voting system, the EKD methodology was applied for constructing system’s goal model and for identifying the relevant processes that realise the operationalised sub-goals. A partial view of this model is shown in Figure 5.
5.2 Applying PriS

5.2.1 Identify privacy-related goals. After performing a stakeholder analysis and an identification of the basic privacy concerns for the e-voting system, it is agreed that the privacy goals that need to be considered for the specific system are unlinkability and unobservability.

5.2.2 Analyse the impact of privacy-goals on organisational processes. After the elicitation of privacy goals, the relevant goals-subgoals that unlinkability and unobservability have an impact on based on the application context are identified. These are shown in Table VI.
For every sub-goal, the relevant processes that realise this goal is also identified, shown in Table VII.

It is obvious that one process may realise more than one goal, which may not be directly affected from the introduction of the specific privacy goals. In this case, further impact analysis is performed on each one of the goals linked to the specific process in order to ensure that any alternations of the relevant process will not hinder the realisation of the goals.

For every goal identified, the new privacy goals are realised either by modifying/altering the specific goal or by introducing new ones. For example, for the goal “Ensure the participation of all eligible Voters” and the relevant subgoal “Provide e-access to all eligible voters” two new subgoals for realising the Unlinkability goal are introduced namely “Provide e-access” and “Prevent others to reveal to whom the data are send to”.

In more detail, the e-voting system is responsible to provide eligible voters with a pair of username and password before the election’s starting date. Based on the organizations context, the communication between the system and the user during the dispatch of username and password must be done in an unlinkable way meaning that any malicious third party will not be able to reveal to whom the data are send to even if he/she gets to disclose part of the information being send, thus protecting users personal identified information. The structure of the current goal as well as the suggested modification is shown in Figure 6.

The same process is repeated for all affected goals and their subgoals leading to a new goal structure that incorporates privacy concerns.

5.2.3 Model affected processes using privacy process patterns. The next step is to model the privacy-related processes identified previously. Rather than starting from scratch, PriS suggests the application of the relevant privacy process patterns, described in Section 3.

<table>
<thead>
<tr>
<th>Affected goals</th>
<th>Related processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensuring unlinkability</td>
<td>(G1.1) Ensure the participation of all eligible voters (G1.1.3) Provide e-access to all eligible Voters (G2.1) Equality for political parties (G2.1.3) Ensure integrity of voter’s ballot (G2.2) Equality for voter (G2.2.1) Ensure voter’s eligibility</td>
</tr>
</tbody>
</table>

Table VI. Goals identified during the elicitation process

Table VII. Processes that realise identified goals
Let us consider the privacy-related process “Send Authentication Means to all eligible voters” shown in Figure 6. The privacy requirement affecting this process is unlinkability. Thus, the unlinkability pattern is applied for modelling the impact of the specific privacy requirement on the relevant processes.

The result is shown in Figure 7, which shows that for realising the transmission of the authentication means, an unlinkable communication must first be established.

The same process is followed for all privacy-related processes. It is possible that one process is affected by more than one privacy requirements. In this case the relevant privacy process patterns are successively applied.

5.2.4 Identify the technique(s) that best support/implement privacy-related processes.

For every privacy related-pattern, the developer has to choose the relevant implementation technique indicated by the privacy-related activity in the pattern. The main privacy activities in our example relate to unlinkability, unobservability and authorisation. Authorisation was not mentioned before since it is considered in the unobservability pattern for verifying that every user who tries to see the results is an eligible user.
Specifically, only authorized users are able to verify result’s integrity. Thus, before providing an unobservable connection the system must first verify user’s eligibility. Beside the satisfaction of Unobservability and Unlinkability, PriS identifies that the developer must select also an implementation technique for authorisation which will lead up unobservability. Based on Figure 4, PriS identifies the relevant implementation techniques. The relevant architectures, protocols and tools (marked with an X for each requirement) appear as possible solutions and then the developers can choose which is/are the most appropriate also taking into consideration additional criteria, e.g. implementation cost, architecture complexity, etc.

The developer may choose to use different implementation techniques for realising these activities or to use one that realises all of them (e.g. Tor).

5.3 Formal PriS and e-voting
In this section, the application of formal PriS on the above-mentioned case study is presented. The aim is to formally describe the whole process starting from the goal level until the suggestion of the proper implementation techniques.

5.3.1 Identify privacy-related goals (goal level). The first step is to define the set of goals, the G set. In our case the G set consists of all goals and subgoals shown on Figure 5.

\[ G = \{G_0, G_1, G_{1.1}, G_{1.2}, G_{1.1.1}, G_{1.1.2}, G_{1.1.3}, G_2, G_{2.1}, G_{2.1.1}, G_{2.1.2}, G_{2.1.3}\} \]

Next, the parent and child goals of every goal in the G set are defined as it is stated on Definition 4. The procedure Assign_hier(Gi) is used for that purpose. Thus, the following assignments exist:

\[ \text{Assign}_\text{hier}(G_0) = (0, G_1, G_2) \]

\[ \text{Assign}_\text{hier}(G_1) = (G_0, G_{1.1}, G_{1.2}) \]

\[ \text{Assign}_\text{hier}(G_{1.1}) = (G_1, G_{1.1.1}, G_{1.1.2}, G_{1.1.3}) \]

\[ \text{Assign}_\text{hier}(G_{1.2}) = (G_1, 0) \]

\[ \text{Assign}_\text{hier}(G_{1.1.1}) = (G_{1.1}, 0) \]

\[ \text{Assign}_\text{hier}(G_{1.1.2}) = (G_{1.1}, 0) \]

\[ \text{Assign}_\text{hier}(G_{1.1.3}) = (G_{1.1}, 0) \]

\[ \text{Assign}_\text{hier}(G_2) = (G_0, G_{2.1}) \]

\[ \text{Assign}_\text{hier}(G_{2.1}) = (G_2, G_{2.1.1}, G_{2.1.2}, G_{2.1.3}) \]

\[ \text{Assign}_\text{hier}(G_{2.1.1}) = (G_{2.1}, 0) \]
Based on the identification of the privacy-related goals, unlinkability and unobservability are found to have an impact on the specific system. Formal PriS defines on every privacy-related goal the respective privacy requirements that constrain that goal. Thus, the affected goals are defined with the following privacy variables using the Assign_pv(G_i) procedure.

Assign_pv(G_1.1) = (0, 0, 0, 0, 0, 1, 0)
Assign_pv(G_1.1.3) = (0, 0, 0, 0, 0, 1, 0)
Assign_pv(G_2.1) = (0, 0, 0, 0, 0, 1, 1)
Assign_pv(G_2.1.1) = (0, 0, 0, 0, 0, 0, 1)
Assign_pv(G_2.1.3) = (0, 0, 0, 0, 0, 1, 0)

In Definition 5, it was mentioned that when privacy requirement constraints a goal that is not an end-goal, all child goals, of this goal, are also affected. In the specific case, goal G_2.1 is constraint by both unlinkability and unobservability goals. However, goal G_2.1.1 is constraint only by unobservability goal while goal G_2.1.3 is constrained by the unlinkability goal. In most cases, a constraint in a parent goal affects all child goals. In the specific scenario, this is not the case. Formal PriS allows changes when defining the values of the privacy variables for every subgoal in cases like the one mentioned in this paper.

After the definition of the privacy requirements on the privacy-related goals, and as it was mentioned before and shown in Figure 6, two new goals are introduced (G_1.1.3.1, G_1.1.3.2). In this case, the G set is redefined as follows:

G = \{G_0, G_1, G_1.1, G_1.2, G_1.1.1, G_1.1.2, G_1.1.3, G_1.1.3.1, G_1.1.3.2, G_2, G_2.1, G_2.1.1, G_2.1.2, G_2.1.3\}

In addition, for the newly introduced goals, formal PriS further defines their connection in the goal hierarchy as well as the proper privacy requirements’ variables that constrain them. Also, goal G_1.1.3 is redefined since new sub-goals where introduced:

Assign_hier(G_1.1.3) = (G_1.1, G_1.1.3.1, G_1.1.3.2)
Assign_hier(G_1.1.3.1) = (G_1.1.3, 0)
Assign_hier(G_1.1.3.2) = (G_1.1.3, 0)
Assign_pv(G_1.1.3.2) = (0, 0, 0, 0, 0, 1, 0)
At the end of this step, formal PriS has defined the G set which contains all organisational goals that characterise the system to be, their connection thus forming the goal hierarchy, as well as the privacy requirements that constraint every privacy related goal.

In the next step, we move on to the process level where the processes and the respective process patterns are identified for realising every operationalised goal identified in the current step.

5.3.2 Analyse the impact of privacy goals on organisational processes (process level). Firstly, the P set is defined, which contains all processes identified to realise the operationalised goals (end-goals) of the previous step, based on Figure 5. Thus, the following definition exists:

\[ P = \{P_1, P_2, P_3, P_4, P_5, P_6\} \]

For every process of the P set, it is defined which are the operationalised goals that are realised by it. By using the procedure \( \text{Match}_G_P(G_i) = (P_k) \), the following assignments are introduced:

- \( \text{Match}_G_P(G_1:1) = (P_1, P_2) \)
- \( \text{Match}_G_P(G_1:3) = (P_2) \)
- \( \text{Match}_G_P(G_1:3) = (P_2) \)
- \( \text{Match}_G_P(G_1:2) = (P_4) \)
- \( \text{Match}_G_P(G_2:1) = (P_6) \)
- \( \text{Match}_G_P(G_2:1) = (P_5) \)

After the above assignments, every operationalised goal is connected with the respective process(es) that realise it.

5.3.3 Model affected processes using privacy process patterns (process level). The next step is to locate which process patterns will be assigned on every privacy-related process. Based on Definition 7, every process is assigned by seven values which represent the seven process patterns. These values are applied on the seven process pattern variables that correspond to the respective process patterns (see Table III. The privacy related processes in the specific case study are \( P_3 \) and \( P_6 \) since \( P_3 \) realises the privacy related operationalised goal \( G_{1:3:2} \) and \( P_6 \) realises the privacy related operationalised goal \( G_{2:1:1} \). As it was mentioned in Section 4, the function \( \text{Locate}_pp(G_i) \) is used for detecting which are the process patterns that will be applied on every process that realises the goal \( G_i \). Thus, the specific function will be applied for both goals \( G_{1:3:2} \) and \( G_{2:1:1} \).

\[ \text{Locate}_pp(G_{1:3:2}) \]

\[ \text{Locate}_pp(G_{2:1:1}) \]
The values of the seven privacy variables of goal G1.1.3.2 are (0,0,0,0,0,1,0) and of goal G2.1.1 are (0,0,0,0,0,0,1). Based on formal PriS way of working and on the description of Section 4, the function Locate_pp() for the goals G1.1.3.2 and G2.1.1 will return the following values:

- For goal G1.1.3.2:
  \[ \text{Id} = 0, \text{an} = 0, \text{unlink} = 1, \text{unob} = 0, \text{for the goal } G_{1.1.3.2} \]

- For goal G2.1.1:
  \[ \text{Id} = 0, \text{an} = 0, \text{unlink} = 0, \text{unob} = 1, \text{for the goal } G_{2.1.1} \]

When formal PriS receives the above results from the application of the function Locate_pp(), it applies the procedure Assign_pp() on the respective processes for assigning the proper values on the privacy process pattern variables. Based on Tables IV and V the following values will be applied on the respective processes,

- For process P3: \( P_3 = (0, 0, 0, 0, 0, 1, 0) \)
- For process P6: \( P_6 = (0, 0, 0, 0, 0, 0, 1) \)

meaning that on process P3 the unlikability process pattern will be applied while in process P6 the unobservability pattern will be applied.

5.3.4 Identify the technique(s) that best support/implement privacy-related processes.

In the final step, PriS identifies and suggests a number of implementation techniques based on the privacy-process patterns derived in the previous step. For accomplishing this task, in formal PriS, a set of implementation techniques is defined, the IT set. Based on Definition 9, the IT set is defined as follows:

\[ \text{IT} = \{ \text{IT}_1, \text{IT}_2, \text{IT}_3, \ldots, \text{IT}_{j-1}, \text{IT}_j \} \]

As it is shown in Figure 4, every process pattern can be implemented by a number of implementation techniques. Based on Definition 10, every implementation technique belonging in the IT set, is defined by seven values which represent the process patterns that can be realised. Thus, for example, the Tor architecture is defined to have the following values \( (0, 0, 0, 0, 1, 1, 1) \).

For the specific case study formal PriS searches the IT set and suggests to the developer the implementation techniques that realise both process patterns (unlikability and unobservability) like TOR, GAP, Hordes, etc. As a second choice, PriS suggest all the implementation techniques that realise every process pattern separately (only unlikability technologies for P3 and only unobservability technologies for P6). The developer is the only responsible for choosing which technologies he/she will apply since other factors like cost, complexity, etc. are taken into account.

6. Conclusions

Privacy is widely recognized as a human right. Individuals should be confident that information about them will be handled fairly. This includes personally-identifiable information in the hands of government agencies. In providing services to the public and carrying out various functions, governments collect and use a wide range of personal information about their citizens (i.e. health records, tax returns, law-enforcement records, drivers license data, etc.). With the shift towards electronic
data management and the growth of the information society, governmental gathering, storing and processing of data have grown dramatically. The introduction of e-government and the electronic delivery of services have further expanded government collection of personally identifiable data. A government’s practice in collecting, retaining, and managing personal data about its citizens pose a wide range of privacy concerns (Anderson and Dempsey, 2003).

To this end, this paper presents PriS, a methodology for incorporating privacy user requirements into the system design process. PriS considers privacy requirements as business goals in the organisation domain and provides a methodological framework for analysing the effect of privacy requirements onto the organisational processes using specific privacy process patterns. Using these patterns, PriS accelerates the modelling of privacy-related business processes indicating where the application of privacy implementation techniques is needed, also suggesting a list of specific implementation techniques that can realise each privacy related process. Therefore, PriS provides an integrated way-of-working from high-level organisational needs to the IT systems that realise them.

Unlike existing security-oriented methodologies, which concentrate either to the design or the implementation level, PriS establishes a clear link between technological solutions and the system design process. In addition, PriS, specifically focuses on privacy goals, rather than considering privacy requirements together with other non-functional requirements. This helps stakeholders to identify privacy-related issues in a more “targeted” way since they are guided by specific requirements and not by general issues and objectives. Also, guidance relating to the applicability of specific privacy-enhancing system architectures assists system developers to reach a decision regarding the proper system architecture.

Furthermore, formal PriS, technically expresses PriS way of working. It provides a set of expressions based on which the whole process, from the goal level till the selection of the proper implementation techniques, is accomplished in a more self-directed way. In addition, with formal PriS, it is easier to check and control the accuracy and the precision of the results based on the privacy requirements defined to constraint the whole system.

Future work includes, as a first step, the development of a prototype tool for supporting PriS methodology. In addition, we aim to build an integrated software tool that will automatically identify the impact of privacy goal in the goal-process structure, based on PriS formal definition. The tool will also provide developers with a description of each implementation technique, as well as a guiding procedure for applying the selected technique.

References


Further reading

Corresponding author
Stefanos Gritzalis can be contacted at: sgritz@aegean.gr

To purchase reprints of this article please e-mail: reprints@emeraldinsight.com
Or visit our web site for further details: www.emeraldinsight.com/reprints