Towards the Design of Usable Privacy by Design Methodologies

Argyri Pattakou, Aikaterini – Georgia Mavroeidi, Christos Kalloniatis
Privacy Engineering and Social Informatics Laboratory,
Dept. of Cultural Technology and Communication
University of the Aegean
Lesvos, Greece
{a.pattakou, kmav, chkallon}@aegean.gr

Vasiliki Diamantopoulou, Stefanos Gritzalis
Information and Communication Systems Security Laboratory
Dept. of Information and Communication Systems Engineering
University of the Aegean
Samos, Greece
{vdiamant,sgritz}@aegean.gr

***Abstract*— As privacy engineering gains much attention, recently literature records a number of methodologies that support software designers to model privacy – aware systems starting from the early stages of the software lifecycle until the late design stages prior to implementation. However, in order for these methodologies to be used and applied successfully from system engineers, it is important to be developed following a number of existing usability criteria for increasing designers' acceptance and performance. In this paper, we, initially, identify the set of usability criteria presented in the respective literature and examine how the existing privacy requirement engineering methodologies conform with these usability criteria. The results show that most methodologies conform with a number of criteria but still there are opportunities for further improvements.**

***Index Terms*— Usability, Comparison, Privacy, Privacy Requirements Methodologies, Privacy Engineering, Privacy by Design Introduction**

I. INTRODUCTION

The importance of considering privacy as a part of a system's development process is widely accepted as an important aspect towards the development of privacy-aware systems. As a result, a number of privacy requirements methodologies have been introduced in order to assist system designers and developers to analyse and elicit privacy requirements for different software systems and in different architectures. Due to the increased usage of such methodologies, it has become important to ensure not only that they can support the elicitation and analysis of privacy requirements, but also to be developed with the appropriate usability aspects in mind, in order to ensure that it will be applied in the intended and proper way.

Usability is defined by the ISO 9241-11 [1] as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use". A more IT oriented definition, that is provided by [2], states that "Usability refers to the quality of a user's experience when interacting with products or systems, including websites, software, devices, or applications. Usability is about effectiveness, efficiency and the overall satisfaction of the user".

Requirement Engineering methods use various concepts for eliciting and modelling functional and non-functional requirements. Lately, due to the vast amount of the recorded privacy incidents, the increased user awareness about privacy protection, the common understanding that privacy is a multifaceted concept that requires attention from the early stages of system design (privacy by design initiatives) etc., many software engineering methods were introduced, and existing methods adopted their way of working in order to be able to support the elicitation and modelling of privacy requirements. A number of these methods stay on a very abstract level while others include tool support for assisting engineering in accomplishing a holistic approach during system design. To the best of our knowledge, usability criteria have never been considered as a way for assessing the usability of every method. Thus, this paper moves into this direction, by presenting well-known requirement engineering methods that deal specifically or additionally with privacy and examines their usability against various usability criteria, identified in the respective literature.

Specifically, the remainder of the paper is organised as follows. The identified usability criteria and their definitions are presented in section 2. In section 3 the requirement engineering methodologies that deal with privacy are described. Section 4 presents the results of this study, by tailoring every identified usability criterion to the context of requirement engineering methodologies. The comparison results are shown in the same section. Finally, section 5 concludes the paper.

II. DEFINING USABILITY CRITERIA

As the information technology is rapidly grown, engineers develop systems that use different technologies and infrastructures in order to serve potential users. It is important to notice that a system is usable when users with or without technical background are able to use the system and complete their tasks without requiring the knowledge of the background system's processes [3]. Thus, usability is a critical sector in the Information Technology, as any system has to be usable in order to support users to complete their tasks successfully. Usability is examined over the user interface (UI) and the system itself [4-8]. However, at this point, it should be reported that, as our

work concerns the usability in privacy requirements methodologies, we identified the connection of usability with such methods. This analysis is presented in section 4.

Literature records a number of usability criteria in a wide range of IT sectors. The aim of this section is to summarise the usability criteria that are reported in the literature of several IT sectors. Due to sectors' wide range, we had to restrict the selection of them. In that case, we selected these sectors that report many usability criteria. The selected sectors are: cloud computing, software engineering, mobile devices, web services, and health care systems. Additionally, during analysing these criteria, we match them with the specific IT sectors in order to highlight the importance of these usability criteria in specific systems or architectures.

Usability criteria can be considered as a base for software engineers that design systems in order to examine on time the usability of the system into consideration. The most reported guidelines are based on ISO 9241-11 [1] contents and Nielsen criteria for usable user interfaces or systems [9]. Relating this ISO, effectiveness, efficiency and satisfaction are three requirements that a system should fulfil [10]. Nielsen proposed ten criteria related to aesthetic of interface, help, prevention of errors, the flexibility of use, consistency of system's actions, visibility of system's status, etc. [9].

In order to review those criteria that can describe a system as usable, we consolidate ISO 9241-11, Nielsen criteria, and the usability criteria that have been recorded in the literature. The most common identified criteria are: accessibility, adaptability, aesthetics, completeness, consistency, control customisation, error prevention, help, learnability, memorability, predictability, reliability, responsiveness, simplicity, transparency, understandability, usefulness, valuableness, and visibility. A more comprehensive analysis of these criteria follows in order to explain how each criterion is applied in several approaches as well as the potential connections among them.

Accessibility: Although this criterion is important for all sectors, it is examined mostly in cloud computing and web services. Cloud computing provides shared computing resources which should be accessible to consumers with a variety of needs and characteristics. Similarly, web services should be accessible to their consumers to achieve goals in specific contexts of use. Services of these two sectors should not be accessible only by expert users, but also by users with specific skills. For example, a system should be usable by hearing or visually impaired users [3][8].

Adaptability: The system should be able to be adapted according to changes in its environment or in parts of the system itself. This criterion is mostly examined in cloud computing and mobile services [6] [11].

Aesthetics: This criterion is examined mostly on mobile devices and is related to the interface of the system. The interface should be designed in such a way that the product will be appealing, friendly and attractive to the user. [3] [10-11].

Completeness: Systems should be designed in a way that all user requirements can be satisfied, so users will complete their tasks effectively. This criterion is reported mostly to cloud computing and web services [3] [11].

Consistency: This criterion indicates that systems and their user interfaces should be consistent to help users complete their tasks. If functions, notifications, symbols, etc. have consistency, services offered by the respective systems will be less confusing to users. It is mostly used in cloud computing, and mobile devices [3] [8] [10-11].

Control: Users should have the control of the system. They should have the choice of agreeing to or refusing some actions. This encourages users to continue using systems because they feel more comfortable while using them. This criterion is used on cloud computing, mobile devices and web services [3] [8] [10].

Customisation: Systems are used by users with different skills, knowledge and needs. A system should be customisable in order to be used by many users. For example, language selection could be provided as an option. This criterion is examined mostly on cloud computing and web services [3] [8].

Error prevention: To avoid errors, all systems should have this criterion. Alerts, notifications or other ways to warn users for potentially wrong choices or actions are helpful. This criterion is examined in all discussed sectors [3-6] [10-11].

Help: All products should have some short of documentation, which helps users to understand the way that specific actions should be carried out to complete a task. This criterion is examined mostly on mobile devices and web services [3] [10].

Learnability: When a user utilises a system's functions and learn how to use it, then, the aim of this criterion is accomplished. This criterion is reported mostly on cloud computing, software and mobile services [3-6] [11].

Memorability: The scope of this criterion is accomplished when a user remembers some sequences of actions after having used a product or service on a previous occasion. A user is, in this case, more prone to using it. Especially, this criterion is examined in cloud computing, mobile devices and software engineering [5-6] [11-12].

Predictability: Users should anticipate the next state of a system, given the current one. Predictability is a criterion that is given high attention, especially by developers of cloud computing applications [2].

Reliability: Having this criterion, a system should perform its required functions under stated conditions for a specified period. It is mostly examined in cloud computing, web services, and health care sector [3] [8] [11].

Responsiveness: This criterion is mainly examined in cloud computing because this sector requires a higher degree of performance. If a system reacts quickly, it keeps users engaged and does not inconvenience them [8].

Simplicity: Simple systems do not mean low-level designed systems (e.g., less buttons, categories, etc.). A system should be well designed and well structured. This criterion is examined mostly on cloud computing and web services [3] [11].

Transparency: One of the most important principles for human-computer interaction is the transparency of a system. Users should use systems without knowing the systems' functions. This criterion is used by all sectors [3] [8].

Understandability: In every sector all functions should be understandable by their users. This criterion is examined in all sectors. Understandability facilitates the efficient interaction of the users with the product, in order for the former to complete their tasks [3] [5].

Usefulness: All systems should be useful and offer information to their users. For example, health care systems are useful for hospital staff and their patients because they keep records of patients' history, appointments, etc. Hospitals need systems to improve their processes, save time, etc. Before designing a product or a service, developers should know who will use it, why they need it and other information to create a more useful system for the specified audience. This criterion is examined in all sectors [3].

Valuableness: A system is valuable when it provides functions that are less likely possible to find in other systems. This criterion is mainly examined on cloud computing environment [8].

Visibility: During the interaction between users and systems, the state of these systems should be given to users. Users should know the current state of systems to manage their actions. Visibility is mostly reported on cloud computing, mobile devices and web services [3] [8] [10].

III. Privacy Requirements Methodologies

As the software industry presents a rapid growth during the last decades, many methodologies and tools have been introduced in order to support system analysis and design process. However, the software engineering community realised that any possible security and privacy attacks might cause many costly and time-consuming processes. Under these circumstances, many researchers considered the need for examining security and privacy not as an ad-hoc process during implementation stage but as an integrated process during the system analysis and design level. Thus, a number of requirements engineering methods that deal specifically or additionally with privacy issues, have been introduced in order to support the elicitation of i) privacy requirements or ii) privacy in parallel with security requirements from the early stage of system design level. In this section, first, we review a number of privacy requirements engineering methods. This analysis serves in our analysis presented in section 4.

A. Privacy Requirements Engineering Methodologies Review

1) *LINDUUN:* LINDDUN [13] is aiming to support the elicitation and fulfilment of privacy requirements in software-based systems through a privacy threat analysis framework. The first step of LINDDUN methodology comprises the design of a data flow diagram (DFD) of the system, as well as the identification of the privacy threats of the system. Then, the identified privacy threats are related to the listed elements of the DFD. According to the LINDDUN methodology, threats are categorised in seven types: Linkability, Identifiability, Non-repudiation, Detectability, Information Disclosure, Content Unawareness, Policy and consent Non-compliance. Privacy threats trees and misuse cases are used in order the threat scenarios of the system to be collected. LINDDUN methodology supports system designers to select the appropriate techniques that satisfy the selected privacy requirements through matching privacy-enhancing technologies (PET's) to each identified privacy requirement. LINDDUN is not supported by a formal tool.

2) *SQUARE for Privacy:* SQUARE methodology [14] was first introduced as a risk-driven method that supports the elicitation, categorisation, prioritisation and inspection of the security requirements through a number of specific steps. As privacy plays an important role in software engineering and SQUARE does not support the elicitation of privacy requirements, the authors of SQUARE extended their methodology in order to support the elicitation of privacy requirements at the early stages of software life cycle. SQUARE for privacy [15] follows the same steps as the original SQUARE method in conjunction with the Privacy Requirements Elicitation Technique (PRET) [16]. PRET tool includes a database of privacy requirements based on privacy laws and regulations with the prospect of supporting the elicitation and prioritisation of privacy requirements.

3) *PriS:* PriS [17] is referred as a goal-oriented approach that considers privacy requirements as organisational goals that have to be achieved by the system into consideration. Additionally, PriS uses privacy-process patterns in order to model the privacy-related organisational processes, to describe the affected organisational processes by the privacy goals, and to support system developers to select the most appropriate implementation techniques and architectures that best satisfy privacy goals. PriS method includes a formal case tool [18] in order to support the operation of the method.

4) *RBAC:* The RBAC [19] framework has been introduced as an agent-oriented framework that is aiming to model privacy requirements and to map user's roles and permissions with a structured way. The main idea of the RBAC method is that it considers privacy requirements as constraints on permissions and user's roles in order to define access control policies. The RBAC framework is not supported by software tool.

5) *STRAP:* The STRAP [20] method has been introduced in order to promote the design of privacy-aware systems and to support system designers to identify privacy requirements from the early stage of systems' design level. STRAP is referred as a goal-oriented approach and it is based on a structured analysis of privacy vulnerabilities, as well as on an iterative process of four steps (Analysis, Refinement, Evaluation and Iteration) for the integration of preferences. The main idea of this method is that the derived privacy vulnerabilities are identified as privacy requirements and these vulnerabilities are considered as obstacles in the satisfaction of a system's functional requirements. The STRAP method is not supported by formal tool.

6) Secure Tropos with PriS: Islam et al. [21] presented a model-based process that takes into consideration security and privacy concepts in parallel at the early stages of system analysis and design. This process integrates two different

methods. Secure Tropos is the main methodology that is used in order security requirements to be identified. However, as this method does not consider privacy concepts, Secure Tropos extended by integrating the PriS method. In this way, it is possible the parallel identification of security and privacy requirements of a system. Thus, security and privacy requirements of a system into consideration can be identified in parallel [22].

7) *The i* method:* The i* method [23] was introduced as an agent-oriented method that focuses on system agents and their interdependencies and was aiming to analyse, model and design the organisation's processes at the early stages of system design. However, recently, the i* method is used in order to support the identification of security and privacy requirements of a system into consideration and to model these requirements through the system processes. The method focuses on the notion of soft goals, as it does not examine the overall system goals but the individual goals of system's actors. As it is referred above, the primary target of the i* method is to construct a domain model that will capture all the involved actors and their dependencies in order to perform an accurate security analysis. The i* method is formally supported by a case tool namely Organisation Modelling Environment (OME) [24].

B. Structural analysis of the Privacy Requirements Engineering methodologies

Based on the respective literature [25-26], a RE methodology mainly consists of its modelling language, processes, and tool. The modelling language can be graphical or textual and both cases use specific notation. The visual notation of a graphical modelling language consists of a set of graphical symbols, a set of compositional rules and definitions of the meaning of each symbol [27]. In particular, each concept can be described by using 1D graphic elements (lines), 2D graphic elements (areas), 3D graphic elements (volumes), textual elements (labels), and spatial relationships. In the examined methodologies we have identified the use of figures, such as rectangles, triangles, polygons, ellipses, circles, polylines, text objects, etc. A textual modelling language includes several variables with data types, such as Boolean and Integer, connected with the use of various symbols (i.e. <, >, =, $\cap$, $\in$, and !=) and/or keywords, like Boolean logical operators (i.e. Boolean AND, OR, and NOT). The privacy requirements methodology is applied through its several processes, i.e. distinct phases/steps. Some methodologies offer specific tools (software programs) to support their use.

Table I summarises the privacy requirements engineering methods that presented in the previous subsection and records the type of their modelling language, processes, and the formal tools that some of these methods are supported from.

As it is reported previously, the concept of LINDUUN methodology is a privacy threat analysis. The method consists of a graphical modelling language and its concept is described using figures (e.g., a circle is used for the "root threat"). Connecting all the defined figures, diagrams are created (e.g., threat tree for linkability of a data flow), which are used for the application of the methodology. The design of privacy threat trees supports the processes of LINDDUN and the threat scenario of the system is collected.

The same parts are included in the method Secure Tropos with PriS. The method uses a graphical modelling language, where figures are defined and used (e.g., a circle is used for the "actors"). Through the activities of this method (e.g., identification of requirements, etc.), security and privacy concepts are considered in parallel at the early stage of system analysis.

Equally, the method RBAC consists of modelling language and processes. In this method, the modelling language is textual, i.e. it uses logical operators (e.g., BOOLEAN AND, OR, etc.) and variables. RBAC supports the modelling of privacy requirements and user privacy preferences through processes, e.g., the Role-Permission Analysis (RPA) phase.

The SQUARE method for privacy is supported by a tool, namely Privacy Requirements Elicitation Technique (PRET), which helps the elicitation and prioritisation of privacy requirements. Applying the processes of SQUARE for privacy method (e.g., selection of elicitation techniques, inspection of requirements, etc.), privacy requirements are elicited at the early stages of software development process.

STRAP is the only method, which consists of one part, i.e. the processes. Following the four steps of STRAP (e.g., analysis, refinements of privacy requirements, etc.), system designers achieve to analyse and elicit privacy requirements at the early stage of system design circle.

In contrast with the other methods, PriS and i* include all the parts presented in Table 2. The modelling language of PriS is textual, as the notation concerns variables (e.g., A⊆B), various symbols (e.g., $\vee$, $\cup$, $\in$), keywords, etc. Through the processes of this method, privacy requirements are integrated into the system design process. PriS method is supported by the tool, which is called PriS case. On the other hand, i* method uses a graphical modelling language, where figures are defined and used (e.g., a circle is used for the "actors"). The aim of this method is to support the modelling of security and privacy requirements of a system into consideration, which is achieved by several processes (e.g., the analysis of the attacker, vulnerabilities, etc.) The method i* is supported by a case tool, namely the Organisation Modelling Environment (OME) tool.

TABLE I. PRIVACY REQUIREMENTS METHODOLOGIES

Method	**Methodology's part**				
	Language		***Processes***	***Tool***	
	Graphical	***Textual***		***Supported by tool***	***Formal Tool***
LINDDUN	√		√		
SQUARE			√	√	PRET
PriS		√	√	√	PriS Case tool
RBAC		√	√		
STRAP			√		
Secure Tropos with PriS	√		√		
i*	√		√	√	OME

IV. Usability Criteria in Privacy Requirements Methodologies

In this section we examine the fulfillment of each usability criterion on every methodology as presented before. Specifically, the analysis is conducted in two sections. Firstly, we provide specific meaning to every usability criterion presented in section 2 for the assessment of the privacy requirements engineering methodologies since the criteria presented before are generic enough and are not defined specifically for assessment of the RE methods addressed in this paper. Next, we match which criteria belong to the assessment part of the respective methods. For instance, aesthetics criterion, concerns exclusively the use of the tools that support the methodologies. This matching can assist system designers or developers that need to evaluate the usability of a methodology, as they will be able to examine usability criteria only in the appropriate parts of the methodology into consideration. The results of this analysis are summarised and presented in Table 2. Afterwards, based on the results presented in Tables I and II, we analysed which usability criteria are connected to which parts of the privacy requirements methods described in Section 3. A such analysis supports a future research regarding the evaluation of these methods considering usability issues. The results of this analysis are presented in Table 3.

Accessibility: This criterion examines if the processes of the methodology and the tools that support the methodology are easily accessible by individuals anytime. For instance, the tool and the processes of a methodology could be considered accessible, when they are available on the web for free downloading and the installation process is easy by anyone interested.

Adaptability: The criterion is referred to the ability of the tools to be adapted in different users' environments, like in different operation systems or in different system architectures (traditional or cloud environments). The criterion of adaptability is linked with the criterion of accessibility, while a tool in order to be accessible should be adaptable to different environments as well.

Aesthetics: This particular criterion concerns the appearance of the tool's interface. Namely, some questions that could be considered are: i) Are the colours attractive? Do they match each other? ii) Are the used shapes understandable? iv) Is the placement of the buttons – actions understandable or confusing to the user?

Completeness: The criterion of completeness is referred to the ability of a methodology to facilitate user's to complete tasks via well-defined steps. In other words, it examines whether the steps of the methodology help users to complete their actions. This usability criterion can be examined at all parts of a methodology, like the formal language, the notation, the processes and the tool of a method. In the case of the tool, it is examined whether the user interface is understandable and if the tool's steps and commands are clear and comprehensible.

Consistency: This usability criterion concerns the consistency of the defined steps that provides a methodology in order users to complete their tasks. The processes of a methodology should provide a continuous flow; the supported tool should be distinguished from the coherence of its functions and the formal language should represent with accuracy the defined steps of the processes.

Control: Control criterion is referred to the use of the tool that supports a methodology, where users must be able to control their actions and their results, so that they are autonomous and feel comfortable during the use of the tool.

Customisation: This criterion concerns exclusively the use of the tool that supports a methodology and is referred to the ability of a user to customise the tools' options depending on their occasional needs.

Error prevention: The error prevention criterion can be examined in cases that a methodology is supported by a tool and it is referred to the ability of the tool to prevent users from mistakes during their actions, by providing them with error messages, confirmation messages, etc. Moreover, the tool should be understandable to users in order to prevent them from possible mistakes.

Help: This usability criterion can be examined at all parts of a methodology, like the formal language, the processes and the tool of a method. The formal language should be described in detail to be understandable. Methodologies' steps should be explained in a document in order to help users to accomplish their tasks. Usually, this document has the form of a guide.

Learnability: Based on its definition, the steps contained in each methodology should be understandable so that the user can learn how to use such methodologies. The same applies to the tool.

Memorability: When a methodology has comprehensible modelling language and steps, it is easier for the user to remember them. In addition, the functions of the tool that users interact with, should be understandable. In this case, modelling language, processes and tool will satisfy the user and they will use them again with less effort, because upon using them, they can remember their steps, etc.

Predictability: While using the tool, the user should be sure regarding the results of their actions each time. For example, if the user wants to delete a file pressing the corresponding button, then the system should delete the file and not insert another one. When there is a consistency between the actions of the user and the functions of the system, then the system is predictable.

Reliability: The tool of a methodology should provide its functions in the same way under various conditions for a specified period of time. In that case, the user will use the tool with consistency without changing their actions' flow.

Responsiveness: The tool should work and respond quickly. The users should use the tool with effectiveness and efficiency without distracting them from their actions and making extra effort.

Simplicity: The language, steps and tool of the methodologies must be understandable and easy to use. If all these parts are simple and well-described, it is easier for the user to use the methodology.

Transparency: This criterion concerns on the background processes of the system. The interaction between user and computer should be transparent, as the user should not know,

understand, be informed about the background processes of the tool.

Understandability: If a methodology includes understandable processes and they are not complicated, misspelled, or sophisticated, then the users will understand and succeed in completing their actions. The same concerns the language and tool.

Usefulness: In this case, the usefulness of a methodology's formal language, processes and tool for fulfilling a goal is important, i.e. whether they are useful for the user's purpose.

Valuableness: Based on this criterion, users will be able to know the value of the processes and tool, either economic or qualitative. It answers questions such as why to use them, what they will offer us, why we should invest on them. A tool or the processes should consider this criterion because it raises the value of their use.

Visibility: By ensuring this criterion, the user is aware of the state and the processing carried out by the tool.

Table 2 summarises the assignment of usability criteria to the parts of methodologies. According to Table 2, it is clear that some usability criteria cannot be examined in all parts of a methodology but only in some of them. Thus, we distinguish three different categories of usability criteria that can be examined, depending on the parts of the methodology that someone needs to examine.

Category I – Modelling Language: Completeness, Consistency, Help, Memorability, Simplicity, Understandability, Usefulness.

Category II – Processes: Accessibility, Completeness, Consistency, Help, Learnability, Memorability, Simplicity, Understandability, Usefulness, Valuableness.

Category III – Tools: Accessibility, Adaptability, Aesthetics, Completeness, Consistency, Control, Customisation, Error prevention, Help, Learnability, Memorability, Predictability, Reliability, Responsiveness, Simplicity, Transparency, Understandability, Usefulness, Valuableness, Visibility.

At this point, it is noted that the whole set of usability criteria can be examined only in the related tools that support the methodologies. Otherwise, for instance, in case that system designers need to examine the usability of a privacy requirements method in its processes or language parts, only the existing usability criteria that belong in the relevant Categories (I or II) can be considered.

Based on the above categorisation and the parts of the privacy requirements methodologies, described in Table 2, we present a second-level analysis, which concerns the correspondence between the usability criteria and the parts of the privacy requirements methodologies.

TABLE II. USABILITY CRITERIA IN THE PARTS OF A METHODOLOGY

Usability Criteria	**Parts of a methodology**			
	Modelling Language		***Processes***	***Tool***
	Graphical	***Textual***		
Accessibility			X	X
Adaptability				X
Aesthetics				X
Completeness	X	X	X	X
Consistency	X	X	X	X
Control				X
Customisation				X
Error prevention				X
Help	X	X	X	X
Learnability			X	X
Memorability	X	X	X	X
Predictability				X
Reliability				X
Responsiveness				X
Simplicity	X	X	X	X
Transparency				X
Understandability	X	X	X	X
Usefulness	X	X	X	X
Valuableness			X	X
Visibility				X

More specifically, as LINDDUN includes only a Graphical Language and the relevant processes, the usability criteria of the corresponding categories I and II can be examined.

On the other hand, SQUARE for privacy methodology does not include a formal language but it is supported by a formal tool and the relevant processes. Thus, all the usability criteria can be considered during examining the usability of its tool but only the criteria of Category II can be examined for the processes part of the method SQUARE for privacy.

As it is described in Table 2, PriS methodology includes all the parts of a methodology, namely textual formal language, processes and formal tool. So, the usability criteria of the Categories I, II and III can be examined for the corresponding parts.

Regarding to RBAC and Secure Tropos with PriS methodologies, both of them include formal language and processes. The difference is that RBAC uses a Textual formal language, while Secure Tropos with PriS uses a Graphical formal Language. Thus, in both of them, usability criteria that belong in Category I, can be considered for the Language part of methodologies and usability criteria that belong in Category II can be examined for the main processes of these methodologies.

STRAP methodology includes only a main process and it is not supported by a formal language or tool. Thus, in case of

evaluation of this method, only the set of usability criteria in Category II can be examined.

Finally, as the i* method includes a Graphical Language, processes and it is supported by a formal tool, the set of usability criteria from the three categories can be considered for evaluating the corresponding parts of the methodology.

Table 3 summarises the usability criteria that can be considered in each privacy requirements methodology depending on the parts that includes each one.

TABLE III. MATCHING USABILITY CRITERIA WITH PRIVACY REQUIREMENTS METHODOLOGIES

Criterion	**Privacy Requirements Methodologies**						
	M1	***M2***	***M3***	***M4***	***M5***	***M6***	***M7***
Accessibility	P	P, T	P, T	P	P	P	P, T
Adaptability		T	T				T
Aesthetics		T	T				T
Completeness	GL, P	P, T	TL, P, T	TL , P	P	GL, P	GL, P, T
Consistency	GL, P	P, T	TL, P, T	TL , P	P	GL, P	GL, P, T
Control		T	T				T
Customisation		T	T				T
Error prevention		T	T				T
Help	GL, P	P, T	TL, P, T	TL , P	P	GL, P	GL, P, T
Learnability	P	P, T	P, T	P	P	P	P, T
Memorability	GL, P	P, T	TL, P, T	P	P	GL, P	GL, P, T
Predictability		T	T				T
Reliability		T	T				T
Responsiveness		T	T				T
Simplicity	GL, P	P, T	TL, P, T	TL , P	P	GL, P	GL, P, T
Transparency		T	T				T
Understandability	GL, P	P, T	TL, P, T	TL , P	P	GL, P	GL, P, T
Usefulness	GL, P	P, T	TL, P, T	TL , P	P	GL, P	GL, P, T
Valuableness	P	P, T	P, T	P	P	P	P, T
Visibility		T	T				T

* M1=LINDDUN, M2=SQUARE for privacy, M3=PriS, M4=RBAC, M5=STRAP, M6=Secure Tropos + PriS, M7=i*, GL= Graphical Language, TL= Textual Language, P= Processes, T= Tool

V. CONCLUSIONS

A number of privacy requirements methodologies have been introduced in order to support the development of privacy-aware systems. However, in order for these methodologies to be applicable during system design stage, it should be ensured that they cover the appropriate usability aspects. The aim of a method is to be used by users with different characteristics and knowledge and this can be achieved by examining usability during the early stages of its design lifecycle. Any methodology must be usable by people with or without technical background in order to be applied successfully and to achieve its goals. Otherwise, the methodology will not be able to achieve the 100% of its goals and the user will not be satisfied by the results.

During our review, we observed that literature does not record any efforts to consider the usability of privacy requirements engineering methods. The aim of this work was to cover partly this gap by examining how usability can be combined with privacy requirements methods. To achieve that, we elicited and redefined the usability criteria in order to understand how they can contribute in such methods. Next, we analysed existing privacy requirements approaches. Afterwards, we determined which usability criteria can be examined in each part of these methodologies. This study can be used by developers in order to determine the criteria that should be considered during the design of such methodologies. Finally, we presented which usability criteria can be examined in the existing privacy requirements methodologies. It is worth noting that each defined usability criterion might be applied in all or in some parts of a privacy requirements methodology. This approach could be a pattern for a future evaluation of these privacy requirements methods regarding usability. However, as security and privacy are two different concepts that should be examined in parallel during designing information systems in order to prevent possible privacy and security incidents, we believe that this work could be a first step not only for assessing usability in privacy requirements engineering methods but security requirements engineering methods as well. Concluding, usability is an important sector and developers should consider such issues for the effective use of the privacy requirements methods.

REFERENCES

[1] ISO 9241-11:1998: Ergonomic requirements for office work with visual display terminals (vdts) – part 11: Guidance on usability. Tech. rep. (1998).

[2] Usability.gov; improving the user experience, https://www.usability.gov/, last accessed 2017/07/01.

[3] Papadomichelaki, X., Magoutas, B., Halaris, C., Apostolou, D., Mentzas, G.: A review of quality dimensions in e-government services. In: International Conference on Electronic Government. pp. 128–138, Springer, Berlin Heidelberg, (2006).

[4] Holzinger, A.: Usability engineering methods for software developers. Communications of the ACM 48(1), 71–74 (2005).

[5] Jackson, M., Crouch, S., Baxter, R.: Software evaluation: criteria-based assessment.Software Sustainability Institute (2011).

[6] Seong, D.S.K.: Usability guidelines for designing mobile learning portals. In: 3rd International Conference Proceedings on Mobile technology, applications & systems. p. 25. ACM, Bangkok, Thailand (2006).

[7] Yen, P.Y., Bakken, S.: Review of health information technology usability study methodologies. Journal of the American Medical Informatics Association 19(3), 413–422 (2011).

[8] Stanton, B., Theofanos, M., Joshi, K.P.: Framework for cloud usability. NIST Special Publication 500, 316 (2015).

[9] Hussain, A., Ferneley, E.: Usability metric for mobile application: a goal questionmetric (gqm) approach. In: Proceedings of the 10th international Conference on information

integration and Web-Based Applications & Services. pp. 567–570. ACM, Linz, Austria (2008).

[10] Bevan, N.: International standards for hci and usability. International journal of human-computer studies 55(4), 533–552 (2001).

[11] Aniobi, D.E., Alu, E.S.: Usability testing and evaluation of a cloud computing based mobile learning app: Students' perspective. International Journal of Computer Science Issues, 13(1), 67 (2016).

[12] Holzinger, A.: Usability engineering methods for software developers. Communications of the ACM 48(1), 71–74 (2005).

[13] Deng M., Wuyts K., Scandariato R., Preneel B., Joosen W.: "A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements", Requirements Engineering, 16(1), 3-32, (2011).

[14] Mead N. R., Stehney T.: "Security quality requirements engineering (SQUARE) methodology", ACM, 30 (4) , 1-7, (2005).

[15] A. Bijwe and N. R. Mead, "Adapting the square process for privacy requirements engineering", 2010

[16] Miyazaki S., Mead N., Zhan J.: "Computer-aided privacy requirements elicitation technique", Asia-Pacific Services Computing Conference, APSCC'08, IEEE, (2008).

[17] Kalloniatis C., Kavakli E., Gritzalis S.: "Addressing privacy requirements in system design: the PriS method", Requirements Engineering, 13(3), 241-255, (2008).

[18] Kalloniatis C., Kavakli E., Kontellis E.: "Pris Tool: A Case Tool For Privacy-Oriented Requirements Engineering", In: Proceedings of the 4th Mediterranean Conference on Information Systems, G. Doukidis et al. (Eds.), Athens, Greece, pp. 71, (2009).

[19] He Q.,Anton A.I.: "A framework for modelling privacy requirements in role engineering", In: Proceedings of REFSQ, 3, pp. 137-146, (2003).

[20] Jensen C., Tullio J., Potts C.,Mynatt E.D.: "STRAP: a structured analysis framework for privacy", Georgia Institute of Technology, (2005).

[21] Islam S., Mouratidis H., Kalloniatis C., Hudic A., Zechner L.: "Model based process to support security and privacy requirements engineering", International Journal of Secure Software Engineering (IJSSE), 3(3), pp. 1-22, (2012).

[22] Mouratidis H., Kalloniatis C., Islam S., Huget M. P., Gritzalis S.: "Aligning Security and Privacy to support the development of Secure Information Systems", Journal of Universal Computer Science,18(12), pp. 1608-1627, (2012).

[23] Liu L., Yu E., Mylopoulos J.: " Security and privacy requirements analysis within a social setting", In: Proceedings of the 11th International Conference on Requirements Engineering, pp. 151-161, IEEE, USA, (2003).

[24] Horkoff J., Yu Y., Eric S. K.: "OpenOME: An Open-source Goal and Agent-Oriented Model Drawing and Analysis Tool", iStar, 154-156, (2011).

[25] He X.,Ma Z., Shao W., Li G.: "A metamodel for the notation of graphical modeling languages", In: Proceedings of Annual International Conference on Computer Software and Applications, IEEE, Beijing, China (2007).

[26] Eichelberger H., Schmid K.: "A systematic analysis of textual variability modeling languages", In: Proceedings of the 17th International Conference on Software Product Line, pp. 12-21, Tokyo, Japan, (2013).

[27] Moody, D.: "The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering", IEEE Transactions on Software Engineering, 35(6), 756-779 (2009)