A System of Privacy Patterns for User Control

Michael Colesky
Radboud University
Nijmegen, The Netherlands
mrc@cs.ru.nl

Julio C. Caiza
Universidad Politécnica de Madrid
Madrid, Spain
Escuela Politécnica Nacional
Quito, Ecuador
julio.caiza@epn.edu.ec

José M. Del Álamo
Universidad Politécnica de Madrid
Madrid, Spain
jm.delalamo@upm.es

Jaap-Henk Hoepman
Radboud University
Nijmegen, The Netherlands
jhh@cs.ru.nl

Yod-Samuel Martín
Universidad Politécnica de Madrid
Madrid, Spain
samuelm@dit.upm.es

ABSTRACT

Privacy by Design is prescribed by the new European General Data Protection Regulation. Getting this privacy preserving design philosophy appropriately adopted is a challenge, however. One natural approach to this challenge would be to leverage design patterns in the privacy domain. However, privacy patterns are scattered, unrelated, inconsistent, and immature. This paper presents a pattern system for user control, which is built upon an existing privacy pattern catalog. By ensuring implementability and uniformity within descriptions, and establishing relationships using consistent terminology, we alleviate some of the aforementioned issues.

CCS CONCEPTS

•Software and its engineering → Patterns; •Security and privacy → *Privacy protections; Privacy-preserving protocols;* Usability in security and privacy;

KEYWORDS

privacy by design; privacy patterns; pattern system; data protection; privacy design strategies

ACM Reference format:
Michael Colesky, Julio C. Caiza, José M. Del Álamo, Jaap-Henk Hoepman, and Yod-Samuel Martín. 2018. A System of Privacy Patterns for User Control. In *Proceedings of SAC 2018: Symposium on Applied Computing , Pau, France, April 9–13, 2018 (SAC 2018),* 7 pages.
DOI: 10.1145/3167132.3167257

1 INTRODUCTION

Privacy by Design (PbD) was jointly introduced by the Dutch Data Protection Authority and the Information and Privacy Commissioner of Ontario, Canada of the time [12]. It is a philosophy that *"ingrains privacy principles into every part of every system"* [7]. Data Protection by Design (and by extension PbD), is a requirement in the European General Data Protection Regulation (GDPR).

This legal framework has an international scope which creates a punitive mandate for privacy protection (Art. 3, 83). It requires services established in the EU, or providing any services to EU residents, to adequately protect privacy (Art. 45). Addressing this need through PbD requires that privacy be considered from the Analysis and Design phases. These phases are often approached through *design patterns.*

Software design patterns were pioneered by Gamma, Helm, Johnson, and Vlisside [22]. These are common solutions to recurring software engineering problems, often used as blueprints for design, and continually improved over time.

An issue frequently faced is how to present patterns in a manner which promotes appropriate use. Privacy pattern *collections* in particular are lacking in coherence and interconnection. Typically patterns themselves are improved by being extended or combined with one another. They may also work together as *compound* patterns which combine multiple pattern approaches into one.

According to POSA (Pattern-Oriented Software Architecture) [5], a *collection* of patterns should support ease of use, comprehension, and extensibility, using a few natural classification properties which provide a roadmap to implementation. The first step towards this is typically a *pattern catalog*, while a *pattern system* adds accessibility, consistency, and illustrates pattern relationships [5].

This work presents a privacy pattern system for user control, accomplished by elevating a subset of an existing community *catalog* [15] to the *pattern system* [5] level. We improve the consistency and implementability of these patterns, as well as the application of the community's classification scheme. As the catalog is extensive, we limit the scope of the system to the patterns within the classification focused on user control.

The paper provides software engineers with the means to facilitate user privacy decisions in a manner consistent with the GDPR. We also provide a basis on which the community can expand to achieve further systemization, or eventually a data protection related pattern language as defined by POSA [5]. This is the ultimate goal of the system.

The following section describes the various efforts within the community towards privacy pattern collections, while Section 3 discusses the process and results of our work. During our presentation of the system, we describe how we adhere to each of the POSA requirements [5]. In Section 4 we elaborate on our findings in applying the schema and the pattern relationship types (summarized as relationships), as well as how we address concerns within our pattern system. Finally we provide an overview of our contributions, conclusions, and suggestions for future work.

2 RELATED WORK

This section discusses different initiatives to gather privacy patterns as a means to help system engineers. We focus on how these efforts fall short of system level as defined by POSA [5] or do not apply to our study domain. According to POSA, there are six requirements for a pattern system. A system of patterns should comprise a sufficient base of patterns, which in turn should be organized, and should support their own evolution as a system. Additionally, all patterns in the system should be described uniformly. Their descriptions should include implementation details suitable for successful application, as well as highlight the relationships between them.

Doty and Gupta [13] had started to work on an online catalog for privacy patterns, which initially included nine draft privacy patterns. It used a variety of categories, including Hoepman's strategies [27]. Little to no emphasis was placed however on establishing relationships with other patterns.

Within the PrimeLife project [21], a pattern catalog for Human Computer Interaction (HCI) was formed to assist in designing accessible user interfaces. This included fifteen patterns for privacy policies, which covered a number of aspects. Some of these include *icons and the display of privacy information, work flows and interaction paradigms*, and *holistic approaches* to the project itself. They include a Related Patterns section, though only as a list of pattern names.

A separate online catalog was set up as part of the PRIPARE project [8]. It describes 26 privacy patterns, classifying them with both attribute tags and categories. To express a few relationship instances they use the following sections: related patterns, supporting patterns, and conflicting patterns.

Drozd [16] additionally proposed an online catalog of privacy patterns which built upon the previous efforts. It presents 38 privacy patterns in total [17]. For classification, a hierarchical schema and application area are used. The hierarchy is based on the ISO/IEC 29100 principles and instructions, while the variety of areas non-exhaustively include: systems' infrastructure, types of applications, and components. Pattern relationships are not explored.

Lenhard et al. [26] further expand upon the previous collections in their mapping study. Beyond these, however, they note various authors which have reported pattern languages in specific domains. Schümmer et al. [33] present such a language with 18 patterns which aim to help control information flow within collaborative environments. Chung et al. [9] report a pattern language for developing applications for ubiquitous computing, presenting 15 patterns organized around 'privacy management'. Hafiz [24], on the other hand, affirms the first pattern language for developing Privacy Enhancing Technologies (PETs). This collection composes of 12 patterns which revolve around anonymity.

As pattern languages, these each show pattern relationships. However, they do not use a formal representation. The ways in which these collections constitute languages are however limited to their specific domains. It is also not suggested that these languages consider the POSA notion of pattern language. The definition given by Lenhard et al. [26] mentions *"a set of patterns and their interrelationships and interactions"*, while this definition sits closer to that of a pattern system [5]. POSA notes that it is not uncommon to mix these notions, reminding that a pattern language should cover *"every aspect of importance in a particular domain"*.

A number of the aforementioned researchers [8, 15, 17] have begun work as a community. They maintain an online repository [15] which hosts various privacy patterns. This community has taken steps to support software engineers, gathering considerable content, and constructing a common schema and classification. However, the patterns within this repository still lack the pattern relationships and implementability necessary to address more holistic and complex problems. We leverage this community catalog in the construction of our pattern system, which aims to address these shortcomings for user control.

3 A SYSTEM OF CONTROL PATTERNS

We have established a system of 20 patterns, shown in Figure 1 along with their identified relationships. They are available at privacypatterns.org. Each of these is a part of the CONTROL classification from Hoepman's strategies [27], and is further classified by a single *tactic* [10] from that strategy. Distribution of these can also be seen in Table 3.

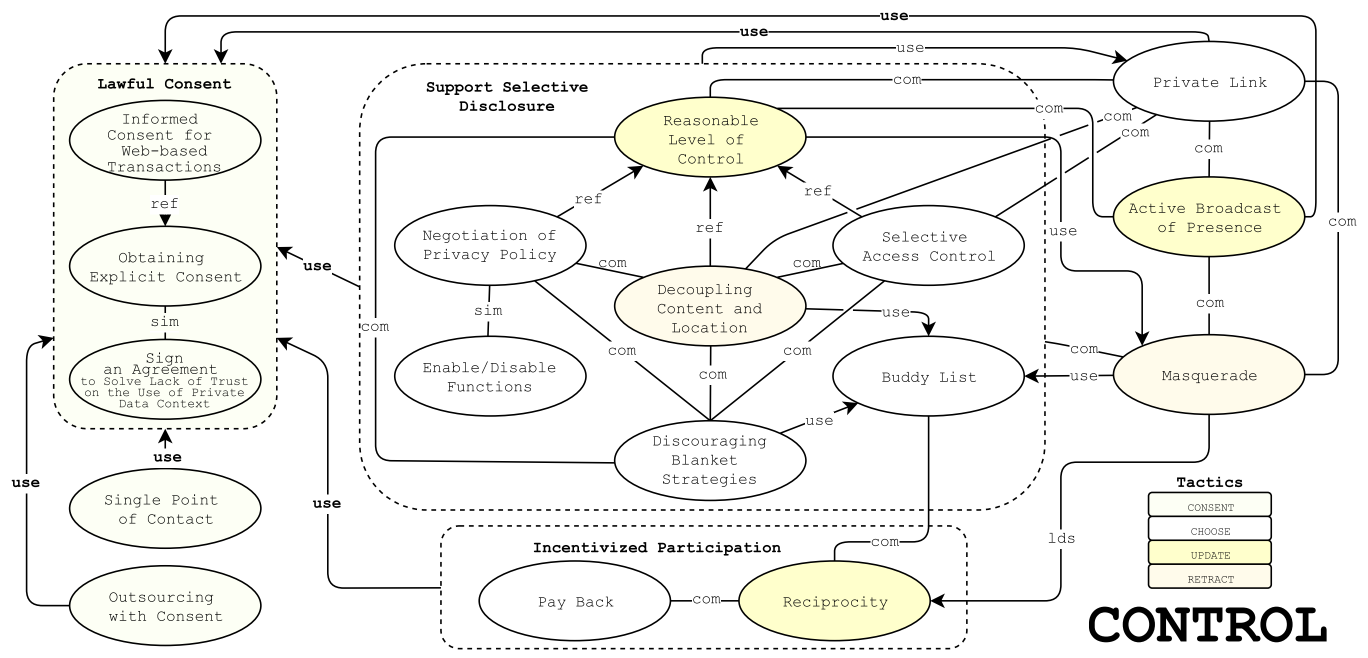
The ways in which our system adheres to POSA [5] are each defined in detail within the following sections. Specifically, they provide six requirements. Our pattern system should support its own *organization* and *evolution*. It should also describe a *sufficient base of patterns* featuring *uniform descriptions*, *relationships*, and *implementability*.

3.1 Sufficient Base of Patterns

At the time of writing there are 32 published privacy patterns and many more being processed in and outside of the community repository. Together, they total 86 potential patterns. However, to maintain a manageable research scope, this work defines a system of privacy patterns within a subset of those. We have selected patterns classified under the CONTROL strategy [27], of which there are 20. This quantity is in line with similar domains such as software architecture [5] (19 patterns) and architectural safety [31] (15 patterns). Additionally, three of these patterns are newly introduced compound patterns, which use a variety of other patterns to provide a more comprehensive solution to a wider problem.

3.2 Uniform Descriptions

Our system conforms to the schema within privacypatterns.org [15]. We have however added an additional useful field for *implementability*, Forces/Concerns, which was not explicit in the schema [15]. According to Harrison [25], *"forces give substance to the problem, and*



Lawful Consent
Informed Consent for Web-based Transactions
Obtaining Explicit Consent
Sign an Agreement to Solve Lack of Trust on the Use of Private Data Context
Single Point of Contact
Outsourcing with Consent
Support Selective Disclosure
Reasonable Level of Control
Negotiation of Privacy Policy
Decoupling Content and Location
Selective Access Control
Enable/Disable Functions
Discouraging Blanket Strategies
Buddy List
Private Link
Active Broadcast of Presence
Masquerade
Incentivized Participation
Pay Back
Reciprocity
use
com
ref
sim
lds
Tactics
CONSENT
CHOOSE
UPDATE
RETRACT
CONTROL

Figure 1: Pattern Organization and Relationships within the System of Patterns

insight into what is behind the symptoms". Table 1 shows the various fields collected by the community in addition to Forces/Concerns.

Table 1: Schema Fields by Importance

Required:	Name, Context, Problem, Forces/Concerns, Solution, *Implementation*, *Related Patterns*
Optional:	Consequences, Constraints, Structure, Examples Known Uses, Summary, See Also, Also Known As

Within Table 1, fields which were previously optional as per the community schema, namely Implementation and Related Patterns, are shown in *italics*. Due to the importance of implementability and pattern relations in our system, these fields were also made mandatory. We sought to ensure that at least the fields we determined to be *Required* were uniform in our system's patterns.

The patterns in this system are described in Table 2. Pattern qualities within are differentiated by *okay*, *decent* (*italics*), and *good* (**bold**). These are followed by their assigned tactic. Since many of these patterns were pre-patterns or otherwise underdeveloped, we endeavored to improve each one so that they may be consistent, comprehensive, and usable. This is further explained in Section 3.5.

3.3 Organization

The system of patterns has been organized according to a classification schema based on *privacy design strategies* and *tactics* proposed by Hoepman [27] and Colesky et al. [10]. These groupings match that followed by the community, and allowed us to delineate our scope to the CONTROL strategy.

Table 3 shows the distribution of potential patterns from the community regarding strategies and tactics. Our patterns are further classified by tactics. These are loosely distinguished by the means for: (CONSENT) *explicit, freely-given, and informed*; (CHOOSE) *optionally selective or exclusive*; (RETRACT) *timely and completely revoked*; and (UPDATE) *accurate and up to date* processing within personal data.

As per Figure 1, patterns sharing the same tactic are shaded according to the legend on the bottom right, while those which are used within a compound pattern are clustered together within its dotted border. Compound patterns are also shaded. Classifications are additionally shown in Table 2

The CHOOSE patterns are most prominent, as they handle the more general rights afforded to users. The GDPR's focus on CONSENT also generates some attention. Conversely, the others, UPDATE (*accuracy*) and RETRACT (*storage limitations*) within Art. 5 clause (d) and (e) of the GDPR [19], are less represented, likely due to specificity. Patterns are positioned to promote relationship visibility.

3.4 Relationships

Explicitly documented relationships are an important feature of a system of patterns. They can aid both in writing understandable patterns and in choosing appropriate patterns for a given problem and context [29].

We have based our types of relationships on those proposed by Caiza [6]. However, we have only found instances of the following relationships in the patterns we examined: *uses, complements, refines, similar to* and *leads to.* The following sections explore the distinctions between these relationships.

Table 2: Patterns Included in the System

Privacy Pattern Name	Summary
Lawful Consent (CONSENT)	A crucial element in privacy protection is ensuring that all sensitive processing is preceded by the acquisition of freely given, informed, specific, and explicit consent.
Informed Consent for Web-based Transactions (CONSENT) [4, 23, 32]	This pattern describes how controllers can inform users whenever they intend to collect or otherwise use a user's personal data.
Obtaining Explicit Consent (CONSENT) [2, 4, 16, 30]	Controllers require consent to be given willingly and specifically when in any way processing the personal data of their users.
Sign an Agreement to Solve Lack of Trust on the Use of Private Data Context (CONSENT) [2]	Services of a controller may require users to sign contracts that stipulate their obligations and processing purposes for which users must consent to use the service. This ensures that users can trust the controller as it is bound to the contract it signs.
Incentivized Participation (CHOOSE)	Users are more willing to contribute valuable input when they can do so without leaking personal data, or perceive an equal or greater exchange in value either monetarily or socially.
Reciprocity (UPDATE) [33]	Let users benefit according to the contributions they make.
Pay Back (CHOOSE) [33]	Give users some benefits in exchange for providing information or content.
Support Selective Disclosure (CHOOSE)	Many services (or products) require the collection of a fixed, often large, amount of personal data before users can use them. Many users, instead, want to freely choose what information they share. This pattern recommends that services Support Selective Disclosure, tailoring functionality to work with the level of data the user feels comfortable sharing.
Discouraging blanket strategies (CHOOSE) [1]	Give users the possibility to define a privacy level from a range of options each time they share content.
Negotiation of Privacy Policy (CHOOSE) [30]	Over time, build user preferences from a privacy-preserving default semi-automatically, through opt-in/opt-out, semantics, and informed solicitations.
Reasonable Level of Control (UPDATE) [3, 9, 28, 32]	Let users share selectively (push) and make available (pull) specific information to predefined groups or individuals.
Buddy List (CHOOSE) [33]	By default, isolate users to a selection of social connections in a user-defined circle of trust. Allow them to expand this circle or create new ones based on the existing members.
Enable/Disable Functions (CHOOSE) [3]	Allow users to decide granularly what functions they consent to before the function is used.
Decoupling [content] and location information visibility (RETRACT) [1]	Allow users to retroactively configure privacy for location information with respect to the content's contextual privacy requirements.
Selective Access Control (CHOOSE) [16]	Allow users to specify who may access the content they generate, both during and after submission.
Single Point of Contact (CONSENT) [4, 20]	The Single Point of Contact is a security authority who protects the privacy and security of sensitive data stored online by validating the authority of requests and ensuring secure communication channels.
Outsourcing [with consent] (CONSENT) [11]	The controller has to obtain additional specific, informed, explicit, and freely given consent before outsourcing data processing to a third party.
Active broadcast of presence (UPDATE) [4, 15]	Users may actively choose to automatically provide updates when they want to share presence information, to increase both the relevance of, and control over, their sharing.
Private link (CHOOSE) [15]	Enable sharing and re-sharing without wide public visibility or cumbersome authenticated access control.
Masquerade (RETRACT) [33]	Let users filter out some or all personal information they would otherwise provide to a service.

Table 3: Strategies and Tactics (Adapted from Colesky, Hoepman, and Hillen [10])

MINIMIZE	HIDE	SEPARATE	ABSTRACT
EXCLUDE 2	RESTRICT 10		
SELECT 4	MIX 6	DISTRIBUTE 2	SUMMARIZE 2
STRIP 1	OBFUSCATE 3	ISOLATE 5	GROUP 2
DESTROY 1	DISSOCIATE 8		
INFORM	**CONTROL**	**ENFORCE**	**DEMONSTRATE**
SUPPLY 9	CONSENT 6	CREATE 3	AUDIT 1
NOTIFY 6	CHOOSE 9	MAINTAIN 3	LOG 1
EXPLAIN 8	RETRACT 2	UPHOLD 3	REPORT 1
	UPDATE 3		

3.4.1 Uses. The first relationship to explore is *uses* (use). As per Figure 1 it comes in two variants. A pattern *may use* (use) another if adapting its solution contributes to the resolution of the problem. It *must use* it (**use**), however, if doing so is integral to the solution. For example, our compound patterns *may use* their constituent patterns in their implementation, as they collectively make up an overall solution, but are not all needed to achieve it. On the other hand, many of our patterns (15 of them) *must use* `Lawful Consent`, as their solutions are not complete without specific, informed, and freely given consent. In order to reduce complexity, compound patterns which *must use* `Lawful Consent` are used to indicate that the same is true for their constituents. This does not apply to other relationships, however.

3.4.2 Complements. A *complements* (com) relationship establishes that two patterns, solving their own problems, may work together to comprise an *anonymous pattern* to solve a new problem. For example, we have found that `Discouraging Blanket Strategies` *complements* the `Negotiation of Privacy Policy` pattern. While the latter allows users to opt-in or opt-out of individual preferences when first using a service, `Discouraging Blanket Strategies` provides granular privacy levels when sharing personal data. These patterns could therefore work together to allow users to revise their initial privacy settings when sharing.

3.4.3 Refines. A *refines* (ref) relationship indicates that a pattern provides a more specific solution to a more specific problem than another more general or abstract pattern. We found that `Negotiation of Privacy Policy` *refines* the `Reasonable Level of Control` pattern. The latter aims to address the provision of sufficient control to users by allowing them to provide information in a selective and granular way. The former meanwhile addresses that same problem in a more concrete scenario, at the first use of a service.

3.4.4 Similar to. A pattern which is *similar to* (sim) another may have a problem and or solution in common. `Sign an Agreement to Solve Lack of Trust` on the `Use` of `Private Data Context` (or rather, `Contractual Consent`) shares many similarities in its solution with `Obtaining Explicit Consent`. The first aims to instill trust in the user through transparency and contractual obligations. The second aims to ensure that user consent is given truly and willingly. Although at first they seem complementary, the solution itself is quite the same. Each point out advantages from their own perspectives.

3.4.5 Leads to. The *leads to* (Lds) relationship describes a situation in which the application of a pattern generates a problem which may be resolved by another pattern. Within our single instance, `Masquerade` *leads to* `Reciprocity`. `Masquerade` allows the user to define a desired identifiability level, which could potentially permit misbehavior. `Reciprocity` could solve this problem by providing different levels of access depending on the identifiability provided.

3.5 Implementability

We have examined and improved the patterns within our system over a three-step process where each author alternates an edit and quality check on each pattern. Patterns were graded from a selection of *ok*, *decent*, or *good* at the end of each step. Distribution was 9:8:3, respectively. The *ok* patterns meet basic implementability requirements, while *decent* patterns have more comprehensive considerations. The *good* patterns fully address a contextual problem. We have assigned this grade to the compound patterns.

In our review, we sought to ensure that each pattern was made to be in line with the suggestions by Meszaros and Doble [29], Harrison [25], as well as Gamma et al. [22]. We examined the available pattern source material in depth before altering the patterns to meet this standard. Despite this, we believe the results of this would benefit from a more thorough evaluation.

As per Harrison [25], we paid special attention to the notion of *Dead Weasels*, that is, the pitfalls of overly simplifying or generalizing the implementation details so that there is only the illusion of procedure. Where otherwise verbose, we opted for more concise variations or, where necessary, *pointers to detail*. In particular, contextually relevant relationships are highlighted within the pattern descriptions.

Despite our efforts towards implementability, it is also worth noting that pattern systems do not need to cover all implementation aspects of an information system [5]. However, as a system should be extensible, implementability may improve as it grows towards that of a pattern language. This, too, is an aspect worth further validating.

Patterns may additionally be implemented through the use of Privacy-Enhancing Technologies (PETs). PETs arise as concrete implementations toward providing privacy to the user. Within the process of this contribution, we have noted that various authors additionally report continuing work on PETs [16, 23, 24]. On the one hand, Hafiz [24] and Graf et al. [23] discuss design patterns for the implementation of PETs, while, Drozd [16] suggests that PETs could provide a means for design pattern implementation. We believe that Drozd's perspective is aligned with our approach. Future advancement on the patterns in our system could result in prospective PETs as concrete implementations.

3.6 Evolution

The system of patterns is additionally available on GitHub [14] and contributes to the community effort. As such it is open to extension. This holds true for both the patterns themselves, as well as the classification, relationships, or schema applied to them. Through the interface that GitHub provides, each pattern's entire content is editable. They are written in the *MarkDown* format which GitHub supports. Changes are however subject to review by those who the community chooses to provide privileges to. The potential changes are submitted as *Pull Requests*.

Outside of the privacy pattern community, the patterns in our system are also capable of evolving independently. Work may be adapted in future collaborations and perhaps later joined to the community project. This was already the case with the patterns delivered by the PRIPARE project on privacypatterns.eu [8], and will soon include the work by Drozd [17] in privacypatterns.wu.ac.at.

4 DISCUSSION

This section describes insight obtained over the course of our examination of the patterns. While we identified and scrutinized potential relationships, various aspects worth further investigation arose. We for example did not find instances of *requires, conflicts, alternative to,* or *extends* relationships. We nonetheless believe these relationships to have concrete examples outside the scope of our system. The following sections highlight additional considerations, beginning with those concerning the use of the system.

4.1 System Usage

As the pattern system is available online, it meets one of the most important aspects for improving its usability: the ability to browse well-classified patterns in a central repository [5]. POSA adds that this single element is more useful than any software *tool* or *method*. While these might aid in guiding engineers towards appropriate patterns for specific contexts, complete automation is not possible [5]. Instead, by ensuring that our patterns are well sorted, connected, and described, engineers may determine their best application.

Typically pattern selection begins with context identification, from which the need for use of the CONTROL strategy is determined. Depending on which *tactic* the engineer focuses on, the relevant patterns within may be identified. The various relationships inside the system aid the engineer in selecting other patterns to be used for appropriate circumstances. This, as stated by previous authors [29], will allow for solving complex privacy design problems.

4.2 Relational Similarities

In our examination of the relationships apparent in the system, we found that there exists a distinction between some instances of *uses*. Some were supportive instances in which a pattern would typically use another, and may find optional or situational improvements on the final result. Others, however, would have largely deficient solutions without the use of some patterns. This behavior, which is also discussed in Section 3.4.1, is distinct from the notion of *requires*.

The *requires* relationship instead stipulates that the pattern in question cannot be utilized without some other pattern (serving a required function) first being in place. The pattern's context would be reliant on the pattern it *requires*, but its solution would be reliant on the pattern it *must use*.

A common candidate relationship for `Lawful Consent` was *requires*. For example, considering whether `Incentivized Participation` *requires* `Lawful Consent`. This was because patterns like `Incentivized Participation` not only *must use* this pattern, but should ideally do so before being utilized. However, effective consent mechanisms can also be considered during utilization. If we were to use *requires*, there are multiple CONSENT patterns, components of `Lawful Consent` in particular, which could be used in sub-optimal or highly specific circumstances.

We note that this introduces the risk of using inappropriate pattern combinations. For example, only `Obtaining Explicit Consent` without ensuring it is informed. To overcome this, each potentially deficient requirement would need additional requirements of its own. Instead, we chose *must use* as a more simple solution.

We encountered similar uncertainty around whether `Reciprocity` and `Pay Back` *leads to* `Lawful Consent`. These patterns are susceptible to obtaining illegitimate forms of consent, such as that acquired to protect emotional investment. For instance, where users benefit without contributing to a service, and risk losing their benefits. Or alternatively, where a desirable feature or reward is blocked with a personal data requirement. `Lawful Consent` addresses this problem. However, this problem should not exist as a consequence of implementation. It should be prevented beforehand. *Leads to* is an after-effect. Therefore, this is also a *must use* relationship.

4.3 Optionality

While analyzing the relationships inside our proposed pattern system, we have seen that in some cases it has been necessary to determine the level of certainty associated with some relationships. A concrete case, mentioned within the previous section, is the *uses* relationship.

This kind of optionality mirrors Caiza's [6] *automatic navigation* (prescriptive pattern usage). Although these notions seem similar, they refer to different perspectives: one to whether patterns are needed, and another to streamline implementation. Further research is required to make more concrete assertions.

4.4 Transitive Relation

We found that some pattern relationships are transitive. That is, if a pattern relates to some other pattern, which relates to another, then the first pattern indirectly inherits the relationship between the others. For instance, a pattern which *uses* another, which *uses* a third, indirectly also *uses* the third. This holds true for many relationships, but we did not find this property consistently for the *complements, conflicts*, and *leads to* relationships. Due to the tight coupling in these, they do not lend themselves to transitivity. Patterns can also be *similar to* or *alternative to* another in different ways, such as whether solutions are distinct or exclusive. This could make transitive assertions in these inaccurate.

In addition to traditional transitivity, in some cases, a pattern may inherit a different relationship than it shares directly. A concrete example of this is `Incentivized Participation`, which *may use* `Reciprocity`. Because `Masquerade` *leads to* `Reciprocity`, it also indirectly *leads to* an implementation of `Incentivized Participation`. This is the case so long as the actual implementation *uses* `Reciprocity`. Therefore, `Masquerade` may lead to `Incentivized Participation`, as this may solve the generated problem.

Further analysis regarding this topic is required, especially considering other systems, before general assertions can be made, however. Where instances of this behavior are useful, they are indicated in the pattern text. Note that Figure 1 does not express relationships as a result of transitivity.

5 CONCLUSIONS

This paper has introduced a preliminary system of patterns to support software engineers in realizing Privacy by Design, specifically providing CONTROL to users. This represents a step forward towards an eventual pattern language for informational self-determination, and then potentially wider privacy issues. The system this paper presents features qualities which address all of the requirements indicated by POSA [5], which we find to be most definitive. These requirements included support for *organization* and *evolution*, and a *sufficient base of patterns* featuring *uniform descriptions*, *relationships*, and *implementability*.

By extending the efforts of a pattern community to classify and make available a large repository, a number of requirements were already in place. With the previous efforts of the community, we were able to build upon the patterns and achieve the requirements set by POSA. The improvements we made aided in ensuring uniformity and implementability in our rewritten patterns. Most notably, the considerable linking achieved in pattern relationships was created from the ground up.

These relationships have been aligned with the patterns as they are at the time of writing. However, this process can be iterative and thus the relationships may evolve along with the patterns. We have found that exploring relationships exposes areas for improvement within the patterns, and thus we expect an iterative process to improve the system overall.

Suggestions for Future Work

The next step in this system is to involve the patterns under the INFORM strategy. This will strengthen the system as a informational self-determination tool. Additionally, it may benefit from an iterative uniformity, implementability, and relational examination. Either the original system or an extended variation would benefit from a validation procedure, potentially through a case study with a real-world example. Additionally, an expert review or panel of experts could be used to assess the system. In this scenario, the experts sought out would likely be experienced software developers. Finally, relationship connectors could be enhanced to better complement the notations software engineers are familiar with.

ACKNOWLEDGMENTS

The authors would like to thank the Netherlands Organization for Scientific Research (NWO) under the 'Patterns for Privacy' project (CYBSEC.14.030), the Escuela Politécnica Nacional, and the Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación of the Ecuadorian Government (SENESCYT) for their sponsorship. This work has also received funding from the EU Horizon 2020 research and innovation program under grant agreement No 731711.

REFERENCES

[1] S. Ahern, D. Eckles, N. Good, S. King, M. Naaman, and R. Nair. Over-Exposed? Privacy Patterns and Considerations in Online and Mobile Photo Sharing. In *CHI '07*, pages 357–366. 2007.

[2] Y. Asnar, V. Bryl, R. Bonato, L. Campagna, D. Donnan, P. E. Khoury, P. Giorgini, S. Holtmanns, M. Martinez-Juste, F. Massacci, J. Porekar, C. Riccucci, A. Saidane, M. Seguran, R. Thomas, and A. Yautsiukhin. Initial Set of Security and Privacy Patterns at Organizational Level. 2007.

[3] H. Baraki, K. Geihs, A. Hoffmann, C. Voigtmann, R. Kniewel, B. Macek, and J. Zirfas. *Towards Interdisciplinary Design Patterns for Ubiquitous Computing Applications.* Kassel, Germany, 2014.

[4] C. Bier and E. Krempel. Common Privacy Patterns in Video Surveillance and Smart Energy. In *ICCCT-2012*, pages 610–615. IEEE, 2012.

[5] F. Buschmann and R. Maunier. *Pattern-Oriented Software Architecture, A System of Patterns.* 1996.

[6] J. C. Caiza, Y. S. Martín, J. M. Del Alamo, and D. S. Guamán. Organizing Design Patterns for Privacy: A Taxonomy of Types of Relationships. In *EuroPLoP '17.*

[7] A. Cavoukian. Privacy by Design. *Information Commissioner's Office*, 2008.

[8] Christoph Boesch, Frank Kargl, Henning Kopp, and Patrick Mosby. privacypatterns.eu - collecting patterns for better privacy, 2017.

[9] E. S. Chung, J. I. Hong, J. Lin, M. K. Prabaker, J. a. Landay, and A. L. Liu. Development and Evaluation of Emerging Design Patterns for Ubiquitous Computing. *DIS '04*, pages 233–242, 2004.

[10] M. Colesky, J.-h. Hoepman, and C. Hillen. A Critical Analysis of Privacy Design Strategies. In *IWPE*, San Jose, CA, 2016. IEEE.

[11] L. Compagna, P. El Khoury, F. Massacci, R. Thomas, and N. Zannone. How to capture, model, and verify the knowledge of legal, security, and privacy experts : a pattern-based approach. *ICAIL '07*, 2007.

[12] Data Protection and Privacy Commissioners. Resolution on Privacy by Design. In *ICDPPC '10*, Jerusalem, 2010.

[13] N. Doty and M. Gupta. Privacy Design Patterns and Anti-Patterns Patterns Misapplied and Unintended Consequences. pages 1–5, 2003.

[14] N. Doty and M. Gupta. github.com/privacypatterns - privacypatterns, 2017.

[15] N. Doty, M. Gupta, and J. Zych. privacypatterns.org - Privacy Patterns, 2017.

[16] O. Drozd. Privacy pattern catalogue: A tool for integrating privacy principles of ISO/IEC 29100 into the software development process. *IFIP AICT*, 2016.

[17] O. Drozd. privacypatterns.wu.ac.at - Privacy Patterns Catalog, 2016.

[18] European Parliament and Council of European Union. Directive 95/46/EC of the European Parliament and of the Council. *Official Journal of the European Communities*, 281(31):31–50, 1995.

[19] European Parliament and Council of the European Union. General Data Protection Regulation. *Official Journal of the European Union*, 2015.

[20] L. Fan, W. J. Buchanan, O. Lo, C. Thuemmler, A. Lawson, O. Uthmani, E. Ekonomou, and A. S. Khedim. SPoC: Protecting Patient Privacy for e-Health Services in the Cloud. 2012.

[21] S. Fischer-Hübner, C. Köffel, J.-S. Pettersson, P. Wolkerstorfer, C. Graf, L. E. Holtz, U. König, H. Hedbom, and B. Kellermann. HCI Pattern Collection - Version 2. 2010.

[22] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: elements of reusable object-oriented software.* Addison-Wesley, 1994.

[23] C. Graf, P. Wolkerstorfer, A. Geven, and M. Tscheligi. A Pattern Collection for Privacy Enhancing Technology. *The Second International Conferences of Pervasive Patterns and Applications*, 2(1):72–77, 2010.

[24] M. Hafiz. A Pattern Language for Developing Privacy Enhancing Technologies. *Software - Practice and Experience*, 43, 2013.

[25] N. Harrison. Advanced Pattern Writing. *Pattern Languages of Program Design 5*, 2006.

[26] J. Lenhard, L. Fritsch, and S. Herold. A Literature Study on Privacy Patterns Research. *SEAA 2017.*

[27] J.-H. Hoepman. Privacy Design Strategies. *IFIP SEC 2014.*

[28] G. Iachello and J. Hong. End-User Privacy in Human-Computer Interaction. *Foundations and Trends in Human-Computer Interaction*, 2007.

[29] G. Meszaros and J. Doble. A pattern language for pattern writing. *Pattern languages of program design*, pages 1–36, 1997.

[30] J. Porekar, A. Jerman-Blažič, and T. Klobučar. Towards organizational privacy patterns. *Proceedings - The 2nd International Conference on the Digital Society, ICDS 2008*, 2008.

[31] C. Preschern, N. Kajtazovic, and C. Kreiner. Building a safety architecture pattern system. *ACM International Conference Proceeding Series*, 2015.

[32] S. Romanosky, A. Acquisti, J. Hong, L. F. Cranor, and B. Friedman. Privacy patterns for online interactions. *PLoP '06*, page 1.

[33] T. Schümmer. The Public Privacy - Patterns for Filtering Personal Information in Collaborative Systems. In *Proceedings of CHI workshop on Human-Computer-Human-Interaction Patterns*, 2004.

[34] US Department of Commerce. EU-U.S. Privacy Shield, 2016.