RSLingo4Privacy: Tools to Improve Consistency Between Privacy Policies and Current Practices

João Caramujo, Shaghayegh Monfared, Alberto Rodrigues da Silva, André Ribeiro, Pável Calado
INESC-ID, Instituto Superior Técnico, Universidade de Lisboa
Lisbon, Portugal

***Abstract*— RSLingo4Privacy is a multi-language approach that intends to improve the specification and analysis of such policies, and which includes several processes with respective tools, namely: (P1) automatic classification and extraction of statements and text snippets from original policies into equivalent and logically consistent specifications (based on a privacy-aware specific language); (P2) visualization and authoring of these statements in a complete and rigorous way based on that privacy-aware specific language; (P3) automatic analysis and validation of the quality of these specifications; and finally (P4) policies (re)publishing. This dissertation addresses the first two processes (P1 and P2), through the definition of RSL-IL4Privacy as a privacy requirements specification language that eases the gap between natural and formal languages, as well as the development of the necessary solutions for the automatic text classification and extraction approach.**

***Index Terms*— Privacy policies, privacy requirements, RSLingo4Privacy, RSL-IL4Privacy, Eddy.**

I. Introduction

Web and mobile applications collect data from users without ensuring traceability between privacy policies and application design decisions. A particular challenge for policy authors and application developers is the need to find a common language that supports translating important privacy policy statements into actionable requirements. For example, the European Union (E.U.) and the United States (U.S.) employ privacy policies as "notices" to end users and, in the U.S., these policies are often the sole means to enforce accountability. Previously, Google developed new services that ultimately re-purposed user data in ways that violated earlier versions of their privacy policy; and third-party "social-plugins" at Facebook were found to transfer Facebook user data to advertisers in violation of Facebook's platform policies. Given the pressure to post privacy policies and the pressure to keep policies honest, companies must do more to align their policies and practices. In this respect, we believe more can be accomplished by enabling developers with new tools to better specify their data needs while policy authors, who are typically legal professionals, can work with those specifications to create more accurate policies or to enforce those policies in the context of developer data needs.

In general, a privacy policy is a technical document that states the multiple privacy-related requirements that a system should satisfy. These requirements are usually defined as ad-hoc natural language (NL) statements. Natural language is an ideal medium to express privacy policies, because it is flexible, universal, and humans are proficient at using NL to communicate. Moreover, NL has minimal adoption resistance as a requirements documentation technique [1]. Although it is the most common and preferred form of requirements representation [2], NL has intrinsic characteristics that become the root cause of quality problems, such as incorrectness, inconsistency or incompleteness [1]. Bearing these properties in mind, it is important to define a domain-specific language (DSL) for specifying the privacy requirements of current and existing privacy policies.

RSLingo4Privacy is a complete approach for specifying and analyzing privacy policies that adopts an *intermediate language* for documenting privacy requirements, hence contributing to the improvement of the specification and analysis of such policies [4]. This intermediate language – called RSL-IL4Privacy – is a DSL that was defined originally as a privacy-aware profile [3] which identified the main concepts of current privacy policies. RSL-IL4Privacy allows policy authors to specify privacy policies by providing several constructs, such as the data type, data recipient and enforcement mechanism, which are necessary to specify and document privacy-related requirements. Through the use of RSL-IL4Privacy, the RSLingo4Privacy approach integrates the following processes [4]: (1) automatic classification, extraction and translation of statements from privacy policies written in NL into RSL-IL4Privacy specifications; (2) specification visualization and authoring; (3) analysis and validation; and (4) specification publishing in a structured and machine-processable format. The goal of the RSLingo4Privacy approach is to use the RSL-IL4Privacy formalization as the necessary mechanism for the specification of privacy policies while providing features for better analyzing and validating the corresponding privacy requirements.

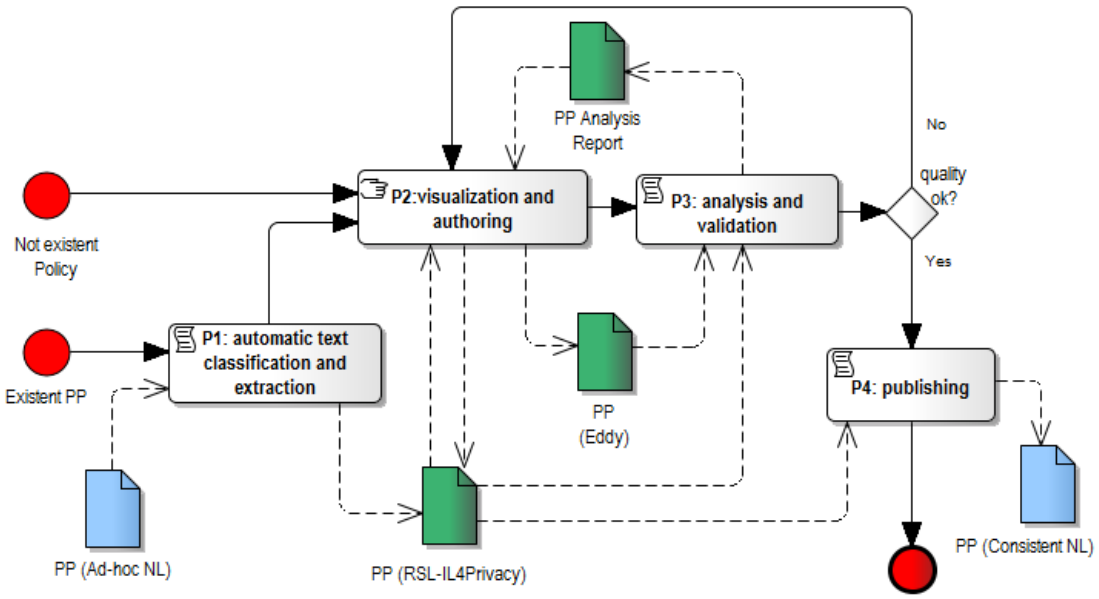

Fig. 1. The RSLingo4Privacy Approach (adapted from [4])

In this paper, we extend the RSL-IL4Privacy language and RSLingo4Privacy approach with the following novel contributions. First, we present a refined version of the RSL-IL4Privacy language, including improvements to its preliminary definition that are crucial to a better specification (hence, validation) of the privacy requirements. Second, we illustrate the multiple representations of this language with a complete case study based on Twitter's privacy policy; Third, we discuss an alignment between RSL-IL4Privacy and the Eddy language [10] and subsequent results from analyzing RSL-IL4Privacy specifications using the Eddy toolset based on five distinct privacy policies (Facebook, LinkedIn, Twitter, Dropbox and IMDb). We also discuss and compare the various policies, detailing the differences emphasized by the RSL-IL4Privacy specifications. Finally, we present the features and transformations included in RSLingo4Privacy-Studio, the tool that supports the proposed RSLingo4Privacy approach.

The paper is organized as follows: Section II overviews the RSLingo4Privacy approach; Section III introduces the RSL-IL4Privacy language and provides a minimal description of its abstract syntax as a UML metamodel and a grammar defined with Xtext framework (for the formalization of such metamodel in terms of a concrete textual syntax); Section IV provides the models and experiments carried out to develop the solutions that deal with process P1 of RSLingo4Privacy; Section V demonstrates the flexibility of this proposal through the development of a method that can issue the necessary interoperability features between RSL-IL4Privacy and other formal languages in order to perform an in-depth analysis of a given policy and analyzes five privacy policies and discusses the conflicts that were obtained; Section VI presents and discusses the related work by comparing RSL-IL4Privacy with other similar languages. Section VII concludes the paper by detailing how RSL-IL4Privacy can improve the analysis of privacy policies and also some threats to validity-

II. The RSLingo4Privacy Approach

RSLingo4Privacy approach supports the specification of privacy policies giving concrete guidance to improve their quality. RSLingo4Privacy includes several processes (supported by respective tools), namely:

1. P1: automatic text classification and extraction;
2. P2: visualization and authoring;
3. P3: analysis and quality validation; and
4. P4: (re)publishing.

RSLingo4Privacy is a multi-language approach that uses the following privacy-aware languages: RSL-IL4Privacy (introduced and discussed in Section III) and Eddy (introduced in Section V). Figure 1 overviews the RSLingo4Privacy approach as a top-level diagram.

A. Workflow

If a given ad-hoc natural language policy exists, the process P1 applies complex text classification and text extraction techniques to automatically produce the equivalent specification in RSL-IL4Privacy. In addition or otherwise, if that policy does not exist, the RSLingo4Privacy approach starts directly with process P2. P2 allows policy authors to visualize and author the new privacy policy in a rigorous and consistent way based on the RSL-IL4Privacy language. Additionally, P2 takes the RSL-IL4Privacy specification and converts it to an equivalent Eddy specification to perform further analysis and validation.

Process P3 takes as input both RSL-IL4Privacy and Eddy specifications, and provides analysis and validation features, producing, for example an analysis report with errors and warnings that can be taken into consideration during these authoring and validation processes.

Finally, when the quality of the policy specified in RSL-IL4Privacy is appropriated, the process P4 is responsible for producing an improved version of the policy, specified again in natural language but in a more consistent and high-quality manner.

B. RSLingo4Privacy-Studio

RSLingo4Privacy-Studio is a tool that supports the RSLingo4Privacy approach, providing the technological support for specifying and analyzing privacy policies. RSLingo4Privacy-Studio is built on top of the Eclipse IDE, more specifically leveraging the Eclipse Modeling Framework (EMF). It uses the RSL-IL4Privacy language as an intermediate language to represent the privacy policies and from it to generate more readable representations of such policies (e.g. MS-Excel or MS-Word files), or to check their quality using its own validators or an Eddy engine (Eddy is explained in detail in Section VI.A). As can be seen in Figure 5, RSLingo4Privacy-Studio relies on several transformations to support the multiple privacy policies representations and the mapping between them, namely Text-to-Model (T2M), Model-to-Model (M2M) and Model-to-Text (M2T). RSLingo4Privacy-Studio provides three main features: Import, Export and Check Quality.

III. THE RSL-IL4PRIVACY LANGUAGE

An important activity when developing and integrating complex software systems, such as social networks, is the enforcement of their privacy policies. One can argue that an incomplete and imprecise specification of privacy requirements may compromise the process of analysis and reasoning of such privacy policies.

RSL-IL4Privacy is a language that enables a more rigorous representation and specification of privacy requirements than writing NL policies alone. RSL-IL4Privacy is defined as a DSL because it allows [6]: (i) the possibility of expressing solutions in the idiom and with the level of abstraction of the problem domain; (ii) increasing the solution's portability, maintainability, reliability and productivity; and (iii) the prospect of reusing domain knowledge in future solutions. In addition, the adoption of a language such as RSL-IL4Privacy allows that its specifications become simpler to read and understand which facilitates the communication between system developers and domain experts [7].

A. Overview of the Language Metamodel

Through the definition of a complete privacy-aware profile [3] it is possible to consider common elements of privacy policies. These elements, and multiple relationships between them, serve as the basis for the specification of the constructs within RSL-IL4Privacy. We overview the privacy-aware profile by summarizing its core elements: *Statement*, *Recipient*, *PrivateData*, *Service* and *Enforcement*. We also highlight the minor refinements to the previous version of the profile. Figure 2

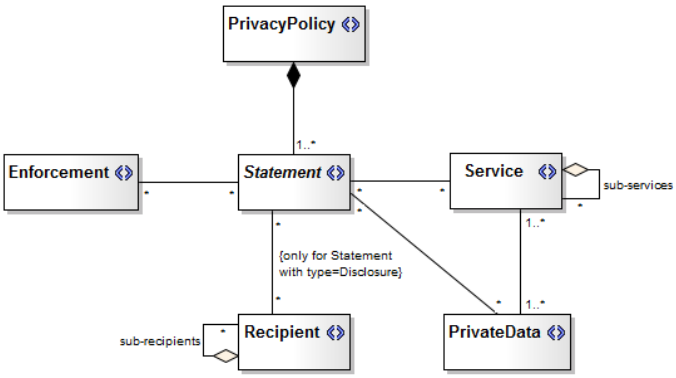

Fig. 2. RSL-IL4Privacy metamodel (partial view)

depicts a partial view of the RSL-IL4Privacy metamodel, a visual schema of the key constructs and associations.

1) Statement

A privacy-aware profile consists of statements. A Statement describes what rules or actions are specified in a policy, thus it can be seen as a *privacy requirement*. A policy comprises a set of statements and each statement may include more than one sentence (e.g., if the meaning of a sentence is dependent on the previous sentence, both sentences encompass a single statement). Depending on the data practices a statement details, it is possible to define five different categories: Collection, defines which data is collected; Disclosure, defines which data is disclosed and eventually to what entities; Retention, defines for how long data will be stored; Usage, defines the purpose for having the data; and Informative, statements with just generic information. It is noteworthy that, according to the information it conveys, a statement can be labeled with more than one type (e.g., "Collection, Disclosure"). Additionally, and for the purpose of reasoning the policies, namely in what concerns deontic logic reasoning, each statement is also defined in terms of its deontic modality (i.e., "Permitted", "Obligation" or "Forbidden") depending on the actions they describe. Lastly, one statement may refer to multiple services and act on different private data (*Service* and *PrivateData* elements, respectively).

2) Recipient

The Recipient element describes the set of entities to whom is disclosed all or part of the users' personal information, i.e., a Recipient element is the set of entities that may receive the information shared by the service provider. The characterization of a given recipient in terms of type and scope is carried out while taking into account the description of its activity that is present in each one of the statements.

3) PrivateData

The PrivateData element represents the users' data that is collected and managed by the social network (or other service provider). A PrivateData element can be defined as personal or usage depending on the source of the information: personal, if such information clearly identifies a given user (e.g., name, email); usage, if the data is gathered based on the user's activity on the system (e.g., device specifications).

4) Service

The Service element describes the multiple high-level services that are provided through the users' point of view. A service can aggregate multiple sub-services that cover more specific purposes regarding the different data flows. While it might be useful to look for major services within policies, smaller services ensure a proper conflict detection. For instance, statements like "We do not share your data in order to send promotional communications" and "We share your data to track the effectiveness of our ads" are not directly conflicting. Even though they refer to the same high-level service – Advertising Service –, the *purpose* for the sharing is clearly different. It is important to point out the association between *Service* and *PrivateData* elements because it makes possible to track what personal information is being employed in each service.

5) Enforcement

Lastly, the Enforcement element is particularly useful because through the description of the mechanisms and tools that are documented in a policy, one can have some insight about how it can be possible to enforce privacy requirements of such privacy policies. On the other hand, it also encompasses rules and specific actions with regard to the use of the system that are important for the enforcement of a given privacy policy.

B. Multiple Representations

The RSL-IL4Privacy language is implemented with different technologies that provide multiple representations (concrete syntaxes), namely: tabular, graphical, and textual representation. The textual representation represents the concrete syntax of RSL-IL4Privacy.

A partial definition of the RSL-IL4Privacy grammar is shown in Fig. 3. The rules of the grammar defined the key entities and their relationships. Each entity has a name and some properties. The type of a property can be defined as final data type (e.g., String, Enumeration) or as a reference to other entities. From this grammar definition the Xtext framework generates an authoring environment that includes various language components and features common to powerful text editors such as syntax coloring, code completion, or error checking.

IV. Automatic Classification and Extraction

The automatic classification and extraction of relevant information (P1 - the first process of the RSLingo4Privacy approach) aims at accomplishing the mapping between original, existing privacy policies and RSL-IL4Privacy specifications. In other words, it encompasses the task of getting the necessary information from existing policies and populating equivalent RSL-IL4Privacy specifications with their mandatory elements that describe the same information. To fulfill this process we rely on two separate tasks: classification of statements and extraction of information for RSL-IL4Privacy constructs.

A. RSL-PL4Privacy

```
PrivacyPolicy:
    'package' name=QualifiedName '{'
        privateData+=PrivateData*
        recipients+=Recipient*
        services+=Service*
        enforcements+=Enforcement*
        statements+=Statement*
    '}';

PrivateData:
    'privateData' name=ID ':' type=PrivateDataType '{'
        'name' aliasName=STRING
        ('partOf' partOf=[PrivateData])?
        ('description' description=STRING)?
        attributes+=Attribute*
    '}';

    Attribute:
        'attribute' name=ID ':' type=AttributeType '{'
              'name' aliasName=STRING
              ('description' description=STRING)?
        '}';

Recipient:
    'recipient' name=ID ':' type=RecipientType '{'
        'name' aliasName=STRING
        'scope' scope=RecipientScope
        ('partOf' partOf=[Recipient])?
        ('description' description=STRING)?
    '}';

Service:
    'service' name=ID ':' type=ServiceType '{'
        'name' serviceName=STRING
        ('refersPrivateData' (refPrivateData=RefPrivateData |
        refPDAll='All'))?
        ('partOf' partOf=[Service])?
        ('description' description=STRING)?
    '}';

Enforcement:
    'enforcement' name=ID ':' type=EnforcementType '{'
        'name' enforcementName=STRING
        ('description' description=STRING)?
    '}';

Statement:
    'statement' name=ID ':' type+=StatementType (','
        type+=StatementType)* '{'
        'name' aliasName=STRING
        'modality' modality=ModalityType
        ('description' description=STRING)?
        ('period' period=STRING)?  // only for Retention Statements

        ('recipient' (recipients=RefRecipient | refRecipi-
        entAll='All'))?  //only for Disclosure Statements
        ('privateData' (privateData=RefPrivateData  | refPri-
        vateDataAll='All'))?
        ('services' (services=RefService  | refServiceAll='All'))?
        ('enforcement' (enforcements=RefEnforcement  | refEnforce-
        mentAll='All'))?
    '}';

    // The values for the types of the various elements (defined as
        enumerations) are not shown due to space constraints. They
        can be seen in the complete metamodel (Appendix A)
```

Fig. 3. Partial Grammar of RSL-IL4Privacy

```
// Collection
W("We|we") W(MOD) W("collect|receive|obtain")
-> "We may collect"
W("You|you") W(MOD) W("provide|give|send") W("us")
-> "you can provide us"

//Disclosure
W("We|we") W(MOD) W(ADV) W(share|disclose|transfer)
-> "We may also share"
W(N|NP) W(MOD) W(v) W("to") W(share|disclose|transfer)
-> "Facebook may need to disclose"
```

Fig. 4. Examples of RSL-PL4Privacy patterns

RSL-PL4Privacy is extended from RSL-PL [5], a pattern-language defined to extract information from Software Requirements Specifications (SRSs). However, we argue that a similar approach in the case of statements of a privacy policy is hardly feasible both for the classification as well as the extraction steps. The statements of a privacy policy can be extremely long, with no "fixed" structure (e.g., lists and/or tables) and with a wide range of vocabulary. These aspects combined (the length and poor syntactic structure of a statement) prove the complexity of defining a set of rules complete enough for recognizing a) the type of the statements; and b) other relevant elements within statements. Regardless of the aforementioned limitations, we still wanted to assess the potential of using patterns to perform the classification of statements.

To come up with a set of patterns that are able to recognize the type of a statement, we make no attempt at defining rules that can "consume"' the entire statement: we only look for "fragments" that are powerful enough to recognize its type. This means that we search for simple patterns based on the syntactic structure of a statement (e.g., Part-of-Speech (POS) tags) along with some specific action verbs [10] for each type. Action verbs designate verbs that are associated with specific actions regarding data.

To provide some sort of consistency with RSL-PL, we used a similar "notation" for the patterns. For our patterns we adapted RSL-PL for the following necessary elements: W(…) represents a word (a generic string), W("example") represents a specific word and W(POS) represents a POS tag. The POS tags that were needed are: N - noun, NP - proper noun V - verb, PRP - personal pronoun, PRP$ - possessive pronoun, MOD - modal verb, ADV - adverb, ADJ - adjective and DET - determiner. Fig. 4 provides two examples of patterns for Collection and Disclosure statements.

B. Classification of Statements

Each statement has a type attribute that can hold more than one value (Collection, Disclosure, Retention, Usage and Informative) as described in the previous chapter. Thus, the task of classifying the statements of a given policy is a multi-label classification problem. Each statement will be represented as a vector of statistical features *Vs.* The challenge of the classification task relies upon using an algorithm that takes as argument the vector of features *Vs* and is then able to predict the type(s) *{T}* of each statement.

TABLE I. The best performance of each solution

	Sol1	**Sol2**	**Sol3**
Perf.	**LP**	**RAkEL**	**LP**
Acc.	0.778	0.845	0.86
HLoss	0.098	0.066	0.06
EMR	0.75	0.817	0.85
F1	0.561	0.701	0.704

Bearing this in mind, we evaluated our various solutions against the most [14] common problem transformation methods: Binary Relevance (BR), Classifier Chains (CC), Ensemble of Classifier Chains (ECC), Label Powerset (LP), Random k-Labelsets (RAkEL) and Fourclass Pairwise (FW). Each method was tested using its default configurations. Additionally, and for each one of the aforementioned methods, we also tested the solutions with different algorithms, namely Naive Bayes (NB), Decision Trees (DTs), k-Nearest Neighbor (kNN) and Support Vector Machines (SVM). All experiments were carried out using MEKA [15] software and 10-fold cross validation. Our data set is comprised of 1759 statements with 81 of those being multi-label. We performed three experiments for our classification solution:

1. RSL-PL4Privacy only (Sol1): This classifier only uses five features, one for each type. If a statement matches one of the patterns, it gets assigned a value of "1.0", "0.0" otherwise. Since we cannot derive patterns for the Informative type, its feature triggers "1.0" if none of the others patterns match.
2. RSL-PL4Privacy + Words (Sol2): In addition to the syntactic patterns, we also pass the words as features. Each feature represents the TF-IDF weights [16] of its constituent words. This weight factor demonstrates the importance of a word for a given statement with regard to the entire set of statements. For this particular problem, we did not consider words that occur in less than three statements. Additionally, we did not include a particular set of English stop words and words with less than two characters (single letters). Finally, we generated tri-grams for each word (a set of three consecutive words).
3. Words only (Sol3): As in the most common text classification work, we use only the words as features (with the same pre-preprocessing as before). This adds up to a total number of 7203 features.

TABLE II. Features used by the CRF

Feature	Description
Word	The word
prevWord, nextWord	The adjacent words
Tag	The POS tag
prevTag, nextTag	The adjacent POS tags
WordPairs, WordTagPairs	Pairs of words and tags (between the word, the previous and the next word/tag)

The effectiveness of the solution was evaluated according to standard performance measures [20] such as Accuracy, HammingLoss (HLoss), Exact Match Ratio (EMR) and F1 (macro averaged). All the solutions deliver better results when using the SVMs as algorithm- Table I shows the best results from each of the solutions. Feature selection was also performed using standard selection functions [21] like Chi-Squared (CHI-SQ), Information Gain (IG) and Pearson's Correlation (CORR). Sol2, with the ECC method, increased the overall accuracy from 83% to 85.6% using the top 5000 features selected with CHI-SQ or IG; Sol3, with the LP method, showed an accuracy of 86.2%.

C. Extraction of Information

The goal of the second step of the P1 process of RSLingo4Privacy deals with the extraction of other relevant information from the statements that are, at this point, already classified. We consider as relevant information the other constructs of RSL-IL4Privacy: Recipients, PrivateData and the Service (its "description"). To achieve this objective, we build the extraction solution on top of the classification task, which means that we only perform extraction on the statements of the most important types: Collection, Disclosure, Retention and Usage. The content of Informative-only statements (although providing some valuable information through the Enforcement construct) is usually very descriptive and has very few mentions to any of these four data practices. Thus, including it in the process would compromise even more the results of a really challenging task. However, the multi-label statements that happened to be also of the Informative type were also included.

For the experiments carried out in this section we used the Stanford NER software [22] that provides an implementation of linear-chain CRF [23] as the algorithm for building the probability models used in the extraction process. CRF are considered the best algorithm for handling sequence labeling tasks. Additionally, 5-fold cross validation was used as simulation technique.

The corpus used in the process of developing of our extractor is a subset of the data set used for the classification task as we only train our solution with the ``Non-Informative" statements (that may include Informative statements if the statements are multi-label), hence removing the Informative-only statements. This accounted for a data set of 521 statements that were then tokenized into 956 sentences and finally into words. A final corpus of 27441 words is annotated with the necessary tags. From these 27441 words, 613 are annotated to define 323 named Recipients ("REC" tag), 1718 to define 755 PrivateData attributes ("PDATT"), 415 to define 208 PrivateData of type "PersonalInformation" ("PDP"), 211 to define 80 PrivateData of type "UsageInformation" ("PDU"), and finally, 3588 words to define 675 Services ("SER").

TABLE III. The performance for the extractor

Entity/Tag	Precision	Recall	F1-score
PDATT	0.813	0.594	0.686
PDP	0.891	0748	0.813
PDU	0.865	0.526	0.654
REC	0.774	0.446	0.566
SER	0.590	0.434	0.500
TOTAL	0.740	0.531	0.618

A NER solution is only as good as the features it is trained with for recognizing the entities in a document. Thus, the definition of a set of features is an important part of the process. There are three "categories" of features [24]: word-level features, list look up features and document-level features. We use a combination of these features as it is provided by the Stanford NER software. Table II lists some of the features used by the CRF to train the extractor.

The effectiveness of our solution, as in any NER task, was evaluated against the standard performance metrics that are also used in single-label classification: precision, recall and F1-score (harmonic mean of same weighted precision and recall). Table III provides the results for our proposed solution.

1) Solution with Gazetter

Due to the fairly low performance of our extractor, one of the possible improvements could be using dictionary look up features [24]. These features are rather self-explanatory. We provide a list (also known as dictionary or gazetter) of words that are previously defined with the tag of a certain entity and the extractor can use these additional features to identify entities in a document. To the best of our knowledge, there is not a gazetter specifically for privacy policies. Thus, in order to assess the impact of using a gazetter in our solution, we created a very basic gazetter that encompassed two sets of entities: "PDATT" and "REC"". Our gazetter has a total of 745 entities, with 504 Recipients and 241 PrivateData attrbutes. These numbers account for various ortographic combinations (e.g., singular/plural, lower-case/upper-case words) of such entities. Table IV details the performance of the extractor using this new feature.

Even though the list of entities provided is really simple and incomplete, the extractor shows an improvement of its performance when using the gazetter as feature. In particular the

TABLE IV. The performance for the extractor using a basic gazetter

Entity/Tag	Precision	Recall	F1-score
PDATT	0.803	0.598	0.688
PDP	0.891	0.752	0.815
PDU	0.880	0.544	0.673
REC	0.782	0.533	0.634
SER	0.586	0.435	0.500
TOTAL	0.740	0.540	0.624

recall of the Recipient entity increases substantially with the gazetter, as well as all the recall values for the other entities. Overall, the extractor delivers a F1-score of 0.624.

V. Specification Analysis with Eddy

One of common claims of *formal languages* to be used for the specification of privacy policies is their possibility of performing some complex analysis and reasoning about such requirements and policies. On the other hand, having a specification written in a language with a *controlled syntax* (such as RSL-IL4Privacy) that still preserves natural language to the maximum, provides a better readability and understanding of such requirements but, at the same time, makes it more difficult to automatically analyze these specifications.

One solution for this dichotomy requires combine both types of languages. Using this solution it is possible to present an understandable specification with complementary representations (in comparison to the original policy), while abstracting the reasoning process which is complex and a forerunner for the creation of more formal languages.

A. Eddy Language and Detection of Conflicts using Eddy

Eddy [10] is one of such formal languages defined for specifying privacy and related data flow requirements, therefore allowing one to trace such data flows and check for potential conflicts. The specification of privacy requirements takes into account three main concepts that are related to each other and that can be structured hierarchically: datum, actor and purpose. Eddy approach consists in mapping requirements (that include the aforementioned concepts) written in natural language text to actions (e.g., collection of data) and roles (i.e., relationships between datum, actor and purpose) in Description Logic (DL) [25]. The use of DL makes it possible to detect conflicts: within a policy, a conflict occurs when there is an intersection between permitted and forbidden actions (e.g., sharing data) carried out by an actor for some purpose. Even though an action can be a "Permission", "Prohibition" (i.e., a forbidden action) or an "Obligation" (in terms of its modality), the detection of the abovementioned type of conflicts does not require allowed actions to be classified as "Permitted" or "Obligation". Since our goal is just to determine if such conflicts exist, labeling an action as "Permitted" or "Obligation" has no particular relevance to the process. However, "permitted-obligation" conflicts tend to occur within privacy policies but their detection is more complex and requires an extra amount of work. Even though we do classify the statements (i.e., the actions they describe) as "Permitted" and "Obligation", the eventual conflicts of this kind are out of the scope of the approach detailed in this paper, thus, their conflict detection process is not discussed.

B. Transform RSL-IL4Privacy into Eddy

A RSL-IL4Privacy to Eddy transformation was defined in the context of the Xtext framework. With this feature it is possible to generate Eddy specifications from equivalent RSL-IL4Privacy specifications. To define this generator we had to find all the common matching concepts between both RSL-IL4Privacy and Eddy grammars.

As referred above a policy specified in RSL-IL4Privacy encompasses a set of privacy elements: *Statement*, *Service*, *Recipient*, *PrivateData* and *Enforcement*. The single definition of a statement (i.e., its description, modality, etc.) encloses the various associations with the remaining elements that are, in turn, defined on the bottom of the policy in RSL-IL4Privacy. On the other hand, a policy in Eddy is represented with a specification header ("SPEC HEADER") and the following specification body ("SPEC POLICY"). The header aggregates the prior definitions of three elements: "P" for Purpose, "A" for Actor and "D" for Datum. The statements are then described on the body. Each statement has a modality, wherein "P" indicates permission, "O" indicates obligation and "R" indicates prohibition; the action verb; the Datum; the source (following "FROM"), the target (following "TO") and the Purpose (following "FOR"). Based on the description of the different elements and keywords from both languages, it is possible to map the following concepts: the *PrivateData* can be considered as Datum, the *Service* as Purpose and the *Recipient* as Actor (target). Since the source ("FROM") refers to the service provider, there is not a direct match between concepts in the two languages. The key mappings between both grammars are defined in Table V.

TABLE V. Matching keywords for RSL-IL4Privacy and Eddy grammars

Language	Modality	Action (type)	Datum	Source	Target	Purpose
Eddy	P, O, R	COLLECT, TRANSFER, USE, RETAIN	D	FROM	TO	P
RSL-IL4Privacy	Permitted, Obligation, Forbidden	Collection, Disclosure, Usage, Retention	privateData	-	recipient	service (description)

TABLE VI. Results using Eddy reasoner engine for conflict detection with converted RSL-IL4Privacy specifications

Privacy Policy	Number of Deontic Conflicts Detected
Dropbox	0
Facebook	2
IMDb	6
LinkedIn	1
Twitter	12

C. Conflicting Statements in Detail

The results obtained using Eddy's deontic conflict detection mechanism with equivalent RSL-IL4Privacy specifications for the policies of Dropbox, Facebook, IMDb, LinkedIn and Twitter are summarized in Table IV. These results account for the number of conflicts between permitted and prohibited actions. If one statement describes a prohibition regarding the handling of some information (e.g., "We will not use your e-mail address for advertising purposes") and then other statement discusses permitted actions for the same data, a conflict occurs.

Twitter is the system that shows the highest number of deontic conflicts (12). Half of the conflicts occur due to the disclosure of personal information, namely Payment Information (e.g., credit card details) or the user's address. Figure 6 depicts some of the conflicting statements in RSL-IL4Privacy using its textual representation. By stating "We consider your payment information and user shipping address private and do not make such information public" (see the statement S22 in Fig. 6) one can safely argue that Twitter is concerned with the improper disclosure of information. Therefore, "not making information public" can mean not sharing any types of personal data with anyone under any circumstances. However, several other statements – in this case, S28 and S34 - describe the disclosure of the user's personal information (including the discussed types) to multiple recipients and for a variety of reasons (e.g., security, processing of transactions). On the other hand, Twitter's privacy policy also shows slightly different conflicts. Even though this social network clearly states that its "Services are not directed to persons under 13" (a business rule defined as an *Enforcement* element in RSL-IL4Privacy), the collection and retention of those users' private information is still carried out until the provider becomes aware of such fact. Conflicts occur when this information – that should not be gathered in the first place and is still undetected – is used and processed by Twitter afterwards. More than pointing out eventual conflicts that may be then used to produce revised versions of the policy, these warnings should trigger Twitter to develop mechanisms for preventing children from using their Services or, on the upper hand, to provide a better assessment before processing the users' information.

```
statement S22 : Disclosure {
    name "S22"
    modality Forbidden
    description "We consider your Payment Information
    and shipping address private and do not make such
    information public."

    recipient R1-…-R8,
    privateData PD7-PD8,
    service SV1-…-SV20
};
statement S28 : Disclosure {
    name "S28"
    modality Permitted
    description "We share your Payment Information with
    payment services providers to process payments.",

    recipient R5,
    privateData PD7,
    service SV6
};
Statement S34 : Disclosure {
    name "S34"
    modality Permitted
    description "If you buy goods or services through
    our Services, we may provide the seller, commerce
    provider or marketplace with your name, email ad-
    dress, shipping address, Payment Information and
    Transaction Data to help prevent, detect and inves-
    tigate fraud or other prohibited activities.",

    recipient R7,
    privateData PD7-PD8-PD10-PD18-PD19
    service SV10
};
```

Fig. 6. Textual representation (concrete syntax) for Twitter's conflicting statements S22 with S28 and S22 with S34 (partial view)

Facebook policy's conflicts are also due to the inappropriate disclosure of user information. We are told that Facebook does not share personal information (i.e., information "that can by itself be used to contact or identify users") with advertising, measurement or analytics partners, unless the user gives permission. However, its policy states that Facebook shares information (due to a vague definition of this concept, we need to assume that it includes also the users' personal information) with a variety of recipients, such as "partners that support their business". It is safe to say that advertising and performance measurement partners also support the business of Facebook, therefore being eligible for receiving a wide range of information – and with no explicit user consent. At the same time, Facebook also declares that it shares information internally (as one might expect) but also with external third parties for the purposes described in its policy. The un-clear definition of the purposes for the disclosure of information leads to inconsistencies. These two examples show conflicts in terms of who is allowed or not to receive information (possibly including users' private information) but also why is necessary for the company to share such information. The conflicts that occur in LinkedIn and IMDb policies are similar to the aforementioned conflicts, therefore being point-less to discuss them.

Lastly, Dropbox is the only policy that showed no conflicting statements during our analysis. Its policy is relatively small

TABLE VII. Comparison of privacy-aware specification languages

Language	**Domain**	**Abstract Syntax,** defined as a…	**Concrete Syntax,** represented by…	**Semantics**
P3P	Web Privacy	XML schema	Textual	Declarative
KAoS	Generic	DAML (XML schema)	Textual	OWL
Rei	Generic	Prolog* constructs	Textual	OWL
XACML	Access Control	XML schema	Textual	Declarative
Eddy	Data Privacy	Grammar	Textual	OWL-DL
RSL-IL4Privacy	Data Privacy	UML Profile + Grammar	Graphic + Textual	Declarative

when compared to the policies of the other service providers and none of the statements explicitly states any kind of prohibition in terms of data practices (i.e., no statements with "forbidden" modality). Thus, as we pointed out, if the focus of our analysis in this paper is the detection of "permitted-forbidden" conflicts, it makes sense that no inconsistencies were found.

VI. Related Work

The analysis of privacy requirements within privacy policies of popular websites, such as social networks, has been the focus of several work and research [11][12]. To provide a common understanding of the concepts that comprise a policy, while giving support for the definition of more formal languages, several works suggest that the creation of different privacy-aware profiles [9][13] - which may be important regarding the development of other third-party applications - can help to easily reveal the existing gap between privacy and system requirements [8]. These works emphasize the importance of having a clear profile, i.e., a metamodel that outlines all necessary and relevant policy elements, as well as the associations between such concepts. On the other hand, the definition of formal languages that could specify privacy requirements has been a prominent part of recent research in Requirements Engineering (RE). Table VII gives a general comparison of the most important approaches on the formalization of privacy requirements specification languages.

Whether deliberately or not, the majority of the privacy-aware specification languages (including P3P, KAoS, Rei or XACML) are not restricted to the data privacy domain, therefore lacking a complete set of the most relevant elements and subsequent relationships between them which we consider necessary to specify with a high-level of detail a whole policy. Their broad domain makes them insufficient for the purpose of our research. On the other hand, an effective communication between requirements engineers and development teams is only maintained if requirements – including privacy requirements – are documented in a clear and unambiguous manner. Being one of the intrinsic features of natural language, ambiguity is a harmful attribute of requirements specification [17], because it may undermine both the goals of achieving a shared vision of the system being built and enabling any sort of further computation upon requirements specifications [18]. The mitigation of this problem requires a great amount of human effort and relies on complementing the documentation of requirements in natural language with several conceptual models, hence improving the accuracy of the requirements [19]. Taking into account the importance of complementing natural language requirements documentation with conceptual models, a formal specification language that is able to provide a thorough specification of privacy-related requirements in a simple representation format (textual or tabular, for different formality) but, at the same time, gives the possibility of having the same requirements specification represented by models is of great value. The design and definition of current formal languages do not suit them to this property, as opposed to what we propose with RSL-IL4Privacy.

VII. Conclusion

In RSL-IL4Privacy, each privacy requirement also maintains the necessary associations between the different elements defined under the scope of this DSL, an extra-layer of information which is directly gathered from the statement at hand. Such information about the remaining constructs can be used directly to quickly infer new knowledge about a specific privacy-related aspect inside a policy, but it also plays an important role on performing a deeper analysis of a policy as it removes the ambiguity that characterizes natural language.

Due to the objective of having a language that is able to specify privacy requirements within policy – as well as the positive results using its reasoner for the detection of conflicts in a policy -, we decided to focus on the development of the necessary interoperability features between Eddy and RSL-IL4Privacy. Even though that this language aims at maintaining natural language to its maximum, RSL-IL4Privacy is powerful enough and has the necessary elements which can be used afterwards in the analysis of privacy policies using reasoning processes of more formal languages (e.g., Eddy). With regard to the syntax of both languages, RSL-IL4Privacy provides a more comprehensible and familiar syntax, despite the difference in size for the specification of a single statement. Despite representing equivalent information, having the original statement included in the requirement specification is one significant difference between both languages. This aspect is particularly valuable in the way that it provides developers

with the opportunity of checking the conflicting statement(s) and take measures to correct and improve it; on the other hand, this also gives end-users the possibility of having a better understanding of the statements that comprise a given privacy policy. Moreover, we believe generic statements (labeled as "informative" in our approach) may still contain useful information in a slightly different context (e.g., explicit business rules, tools or other mechanisms, etc.). Using RSL-IL4Privacy to specify a policy previously written in NL ensures that such information is still kept and even becomes documented in a more organized way. On the other hand, due to a more narrow scope, privacy policies specified in Eddy or in any other standard language tend to restrict the original policies by discarding fragments that do not directly relate to data privacy concerns. In the end, this may cause the loss of valuable non-specific information as it ends up not being included in such regular policy specifications.

REFERENCES

[1] K. Pohl, Requirements Engineering: Fundamentals, Principles and Techniques, Springer 2010.

[2] B. Kovitz, Practical Software Requirements: Manual of Content and Style, Manning 1998.

[3] J. Caramujo and A. R. Silva, "Analyzing privacy policies based on a privacy-aware profile: the Facebook and LinkedIn case studies", IEEE 17th Conference on Business Informatics (CBI), vol. 1, pp. 77—84, July 2015.

[4] A. R. Silva et al., "Improving the Specification and Analysis of Privacy Policies: The RSLingo4Privacy Approach", 18th International Conference on Enterprise Information Systems (ICEIS), SCITEPRESS, April 2016.

[5] Ferreira, D., and A. R. Silva. "RSL-PL: A linguistic pattern language for documenting software requirements." Proceedings of RePa 13 (2013).

[6] A. Van Deursen, P. Klint and J. Visser, "Domain-specific languages: an annotated bibliography", ACM SIGPLAN Notices, vol. 35 (6), pp. 26–36, March 2000.

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[8] L. Bettini, Implementing Domain-Specific Languages with Xtext and Xtend, Packt Publishing Ltd, 2013.

[9] D. Savic et al., "Preliminary experience using JetBrains MPS to implement a requirements specification language", 9th International Conference on the Quality of Information and Communications Technology (QUATIC), pp. 134—137, September 2014.

[10] T. D. Breaux, H. Hibshi and A. Rao, "Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements", Requirements Engineering, vol. 19 (3), pp. 281—307, September 2014.

[11] A. Antón, D. Bolchini and Q. He, "The use of goals to extract privacy and security requirements from policy statements", 26th IEEE International Conference on Software Engineering, 2003.

[12] C. Kalloniatis, E. Kavakli and S. Gritzalis, "Addressing privacy requirements in system design: the PriS method", Requirements Engineering, vol. 13 (3), pp. 241—255, September 2008.

[13] G. Kapitsaki and I. Venieris, "PCP: privacy-aware context profile towards context-aware application development", 10th International Conference on Information Integration and Web-based Applications & Services, pp. 104—110, November 2008.

[14] Tsoumakas, Grigorios, and Ioannis Katakis. "Multi-label classification: An overview." Dept. of Informatics, Aristotle University of Thessaloniki, Greece (2006).

[15] Read, J. et al. "Meka: A Multi-label/Multi-target Extension to Weka", Journal of Machine Learning Research, pp. 1-5, 2016.

[16] Ramos, Juan. "Using tf-idf to determine word relevance in document queries." Proceedings of the first instructional conference on machine learning. 2003.

[17] S. Boyd, D. Zowghi and A. Farroukh, "Measuring the expressiveness of a constrained natural language: an empirical study", 13th IEEE International Conference on Requirements Engineering, pp. 339–349, 2005.

[18] D. Ferreira and A. R. Silva, "RSLingo: an information extraction approach toward formal requirements specifications", IEEE Model-Driven Requirements Engineering Workshop (MoDRE), pp. 39—48, September 2012.

[19] K. Pohl and C. Rupp, Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam – Foundation Level – IREB compliant, Rocky Nook 2011

[20] Zhang, Min-Ling, and Zhi-Hua Zhou. "A review on multi-label learning algorithms." IEEE transactions on knowledge and data engineering 26.8 (2014): 1819-1837.

[21] Forman, George. "An extensive empirical study of feature selection metrics for text classification." Journal of machine learning research 3.Mar (2003): 1289-1305.

[22] Finkel, Jenny Rose, Trond Grenager, and Christopher Manning. "Incorporating non-local information into information extraction systems by gibbs sampling." Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2005..

[23] Lafferty, John, Andrew McCallum, and Fernando Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data." Proceedings of the eighteenth international conference on machine learning, ICML. Vol. 1. 2001.

[24] Nadeau, David, and Satoshi Sekine. "A survey of named entity recognition and classification." Lingvisticae Investigationes 30.1 (2007): 3-26.

[25] F. Baader, The Description Logic Handbook: Theory, Implementantion and Applications, Cambridge University Press 2003.