Extending PriPoCoG: A Privacy Policy Editor for GDPR-Compliant Privacy Policies

Jens Leicht and Maritta Heisel[a]
Paluno - The Ruhr Institute for Software Technology, University of Duisburg-Essen, Germany

Keywords: GDPR, Compliance, Privacy Policies, Tool, Editor.

Abstract: Privacy policies are an important tool for service providers around the world, especially after the enactment of the General Data Protection Regulation (GDPR). Such privacy policies are commonly expressed in long texts written in legalese. In many cases multiple departments of a company are involved in the definition of these policies; however, only the legal department is able to evaluate the level of GDPR-compliance. We propose and evaluate a privacy policy editor that can be operated by a broader audience. Our editor provides policy authors with guidance on what information to include in a policy. Using the Prolog Layered Privacy Language (P-LPL) our editor can also perform GDPR-compliance checks and warn policy authors when compliance issues arise during policy definition. The privacy policies created with our editor are well structured and computer-interpretable as we use an existing policy language (P-LPL). This may also be beneficial for the data subjects, who will be reading the privacy policies, as user interfaces can visualize the policies in structured and better comprehensible ways, compared to the pure legalese texts of today's privacy policies. Data controllers and data processors may also use our editor for defining service level agreements.

1 INTRODUCTION

Although privacy policies are important in the context of compliance with data protection legislation, the development of tools for improving privacy policies has been lacking. Legislations, such as the General Data Protection Regulation (GDPR) of the European Union (European Parliament and Council of the European Union, 2016), require service providers to inform end-users/data subjects about data collection and processing in a transparent manner. A popular solution to achieve this transparency is the use of privacy policies.

Such policies are written in a complex legalese, which the end-users can hardly comprehend. Multiple departments of a company might be involved in the process of defining a privacy policy and only the legal department is able to evaluate the level of GDPR-compliance.

We developed a privacy policy editor that integrates into the Privacy Policy Compliance Guidance framework (PriPoCoG) and provides feedback concerning GDPR-compliance during the policy definition (Leicht et al., 2022). This feedback enables all departments to evaluate the GDPR-compliance of their currently defined privacy policy. Only at the end of the definition process, the legal department should review the resulting privacy policy.

Our editor not only improves the policy definition process, but also provides benefits for the end-users/data subjects. The use of the Prolog Layered Privacy Language (P-LPL) results in structured privacy policies, that can be visualized in different ways, which can improve data subjects' comprehension of the policy. In combination with privacy-policy-based access control (P2BAC) both sides of the privacy policy, service providers as well as data subjects, benefit from privacy policies created with our editor (Leicht and Heisel, 2023). Our privacy policy editor can also be used to express service level agreements (SLAs) between data controllers and data processors. These agreements state how some data should be handled by the data processor.

Since our editor is aimed at professional users, we use the term *user* in this paper when talking about policy authors, who can for example be data controllers or data protection officers. In this paper we address the following two research questions:

RQ1. How can we improve the GDPR-compliance of data controllers and their privacy policies?

[a] https://orcid.org/0000-0002-3275-2819

Leicht, J. and Heisel, M.
Extending PriPoCoG: A Privacy Policy Editor for GDPR-Compliant Privacy Policies.
DOI: 10.5220/0012599600003687
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 19th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2024)*, pages 307-318
ISBN: 978-989-758-696-5; ISSN: 2184-4895
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

RQ2. How can we improve the usability of the PriPoCoG-framework?

The paper is structured as follows: First, we introduce some background knowledge in Section 2. Next, we present our privacy policy editor in Section 3. Afterwards, we discuss the evaluation of our editor in Section 4, followed by related work in Section 5. Finally, we conclude this paper in Section 6.

2 BACKGROUND

In this section we present some background knowledge about the GDPR terminology, the PriPoCoG-framework, and P-LPL policies.

2.1 GDPR Terminology

The General Data Protection Regulation (GDPR) (European Parliament and Council of the European Union, 2016) introduces some terminology, which we use throughout the paper to distinguish different roles that benefit from our policy editor.

Data Subject. Person whose data are collected and processed by the data controller.

Data Controller. Person or legal entity in charge of controlling data handling. The data controller specifies the privacy policy to inform its data subjects.

Data Protection Officer. Person, entitled by the data controller, who ensures GDPR-compliant data handling. Also handles requests from data subjects.

Data Processor (Data Recipient). External entity that processes some data on behalf of the data controller. Called data recipient in the P-LPL privacy policies.

Purpose. Reason for which data are processed. Privacy policies contain purposes explaining to the data subjects why their data are being handled.

Supervisory Authority. The local data protection authority entrusted with the investigation of data breaches and with imposing and collecting fines for such breaches.

2.2 P-LPL-Policies

The Prolog-Layered Privacy Language (P-LPL) implements and extends the Layered Privacy Language (LPL) by Gerl (Gerl, 2020). P-LPL uses Prolog[1] to formalize the language constructs of LPL and requirements from the GDPR, as described in (Leicht et al., 2022). This formalization of the policies as well as GDPR requirements makes it possible to perform compliance checks on privacy policies expressed in P-LPL.

Figure 1 visualizes the structure of P-LPL policies. The structure shown only contains elements that are directly created using our editor. P-LPL contains some elements that are created automatically further down the data usage chain, for example when the data subject provides consent to parts of the policy; these elements are not shown in Figure 1. Square brackets **[]** visualize the fact that this element is a set/list of elements.

Privacy models are used for de-identification and can for example be k-anonymity or l-diversity. They describe which data are considered during the de-identification process. *Automated decision making* is a description of the automated decision making that might take place for a given purpose. *Pseudonymization methods* explain which data elements are replaced by a pseudonym. The purposes of a policy are organized in a hierarchy to provide different levels of detail. How the purposes are arranged is saved in the *purpose hierarchy*. Policies can have an *underlying policy*, which can for example be a service level agreement or a previously accepted privacy policy.

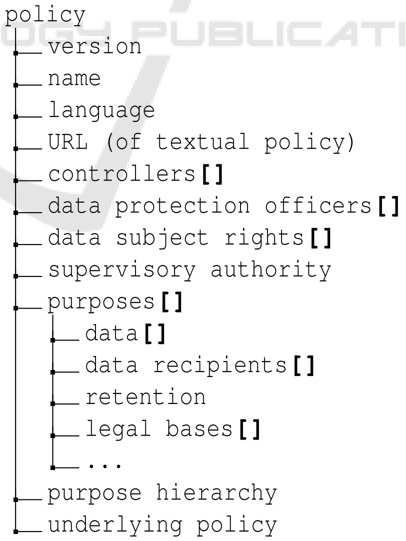


Figure 1: Structure of a P-LPL privacy policy, limited to elements created with our editor (some elements are excluded "...").

[1] https://www.swi-prolog.org/

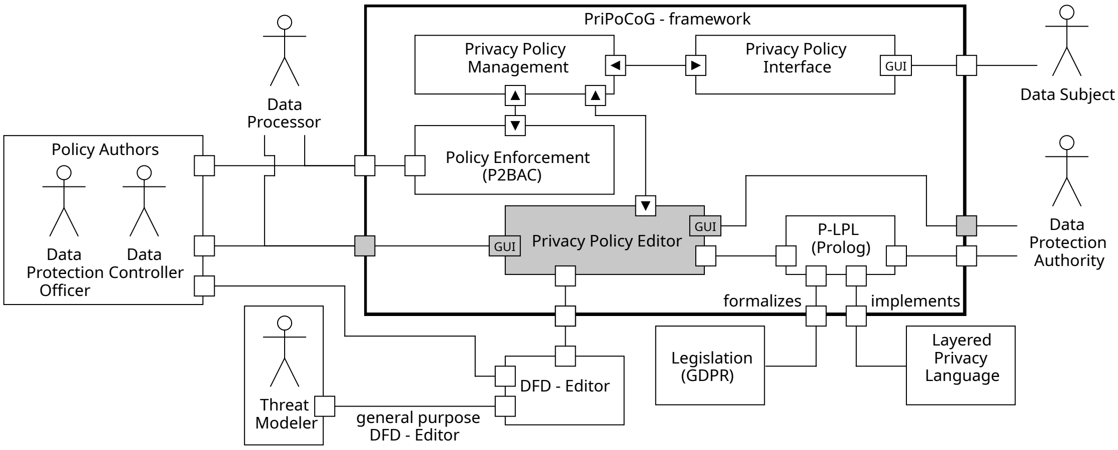


Figure 2: Uml composite structure diagram of the PriPoCoG-framework with the policy editor highlighted in grey, based on (Leicht et al., 2023).

2.3 PriPoCoG-Framework

The Privacy Policy Compliance Guidance (PriPoCoG) framework (Leicht et al., 2022) uses computer-interpretable privacy policies and checks them for compliance with the GDPR. We extend this framework with a privacy policy editor, improving the usability of the framework. Figure 2 shows an overview of PriPoCoG, with our editor highlighted in grey. How our editor integrates into the framework is described in Section 3.1 below.

The PriPoCoG framework uses *P-LPL* to implement *LPL* and formalize parts of the *GDPR*. This formalism is then used to check privacy policies for compliance with legislation (Leicht et al., 2022), either using our policy editor or by directly interacting with the P-LPL command line interface. P-LPL and consequently our editor can be used by *data protection authorities* to investigate GDPR-compliance. PriPoCoG already provides an editor for data-flow diagrams (DFDs), which generates files that can be imported into our policy editor. The *DFD-editor* can be used by *policy authors* as well as *threat modelers*. The framework also provides *policy enforcement* using Privacy Policy Based Access Control (*P2BAC*) (Leicht and Heisel, 2023). This enforcement is used by *data controllers* and *data processors*. *Privacy policy management* and the *privacy policy interface* towards the *data subject* are currently under development.

3 EDITOR

In this section we discuss where our editor is situated within the PriPoCoG-framework. We explain the functionality of the editor, give some details about the implementation, and provide a usage procedure. The editor is open-source and available online[2].

3.1 Framework

Figure 2 shows an overview of where the privacy policy editor is located within the PriPoCoG-framework. The contribution of this paper is highlighted in grey. An overview of the other components of the framework is presented in Section 2.3. In addition to creating privacy policies, the editor we propose can be used by *policy authors* and *data processors* to define service level agreements. *Data protection authorities* can use the editor to check the GDPR-compliance of a given P-LPL policy. Our editor can also be integrated into future *privacy policy management* systems.

Figure 3 gives an overview of the different use cases of our editor. The hierarchy on the left-hand side of the diagram shows that we define the actor *Service Provider* as a more general term for *Data Controller* and *Data Protection Officer*. The *Service Provider* has three use cases: **i)** the general definition of privacy policies, **ii)** GDPR-compliance checks for the policies defined with the editor, and **iii)** definition of Service Level Agreements (SLAs) in cooperation/conjunction with *Data Processors*. Compliance checks can also be used by *Data Protection Authorities*, to assure that privacy policies are GDPR-compliant. The *Data Subject* can indirectly benefit from our editor, as the privacy policies are created and stored in a well-structured manner. This may improve policy comprehension when combined with a suitable UI-representation towards the *Data Subject*.

[2] redacted for blind-review

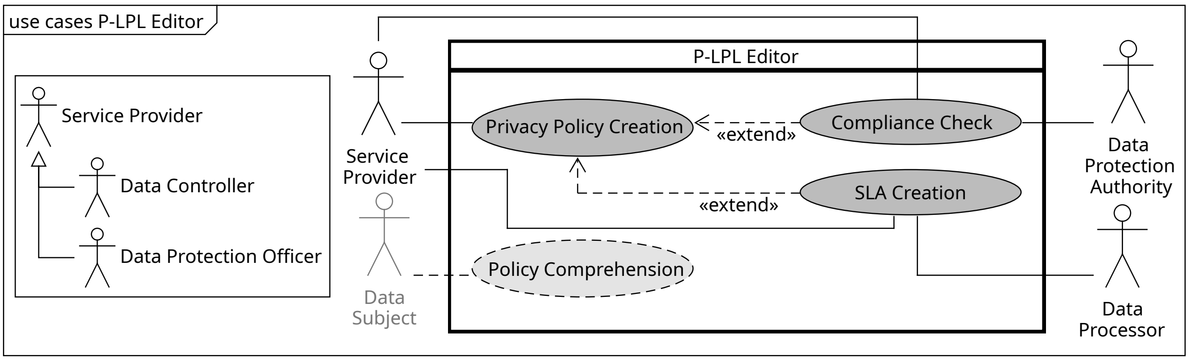


Figure 3: Use cases for our privacy policy editor.

In the following we take a look at the main features of the editor.

3.2 Functionality

We first present the basic features that any privacy policy editor should fulfill, before discussing the unique features of our editor. Figure 5 shows the main view of the graphical user interface (GUI) of our editor.

3.2.1 Basic Features

The editor allows users to create and edit privacy policies. The definition of a privacy policy is performed in the two main tabs of the editor, highlighted in the **b**-region in Figure 5. The *Main Information*-tab is used to specify general information required by a privacy policy. Here we also specify the basic information about the purposes for which data are collected and processed. Further details about the purposes are later entered in the *Purpose Details*-tab.

The privacy policies contain all elements defined in P-LPL, and the editor provides a structured way of entering necessary information. Information can be entered by selecting the corresponding tile in the main area of the editor (**c**-region in Figure 5). A form for entering the relevant information will be shown when editing policy elements. Figure 6 shows an exemplary form, filled with information from the Amazon.de privacy policy.

The editor provides hints for each field, also allowing inexperienced users to define privacy policies. Missing or incorrect information is highlighted in red.

Exporting Privacy Policies. The policy created with the editor can be saved as an XML-file, and loaded back into the editor. For a manual GDPR-compliance check using P-LPL, the policy can also be downloaded as a Prolog-file. Additionally, the editor allows the export of the privacy policy in a plain text-file. These operations are available via the main menu on the left-hand side of the editor (**a**-region in Figure 5).

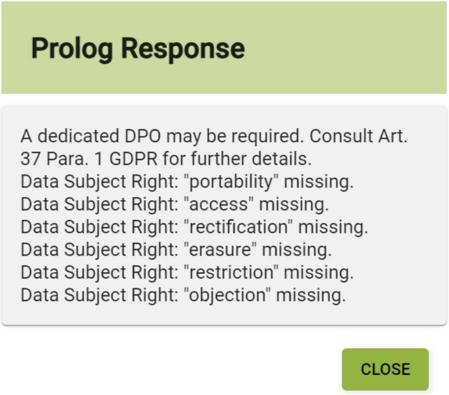
In the following we discuss the unique features of the editor in more detail.

3.2.2 Unique Features

To the best of our knowledge, the following features are unique in the context of privacy policy editors and generators.

GDPR-Compliance. The automatic check of GDPR-compliance can be divided into two types of checks:

1. PLP-Checks. The first type of check is the GDPR-compliance check, achieved by integrating P-LPL into the editor. The information entered in the editor is transformed to P-LPL syntax and input into the P-LPL compliance checker. The results of the compliance check are displayed to the user of the editor (cf. Figure 4). By integrating P-LPL into the editor any future updates of P-LPL where the policy



Prolog Response

A dedicated DPO may be required. Consult Art. 37 Para. 1 GDPR for further details.
Data Subject Right: "portability" missing.
Data Subject Right: "access" missing.
Data Subject Right: "rectification" missing.
Data Subject Right: "erasure" missing.
Data Subject Right: "restriction" missing.
Data Subject Right: "objection" missing.

Figure 4: Exemplary compliance feedback: "a dedicated Data Protection Officer (DPO) may be required", ...

language itself remains unchanged will be directly included in our editor. Hence, updates to the compliance checks are easy. Changes to the P-LPL policy language, however, may require changes in the editor.

Our editor provides an increased usability compared to the bare P-LPL implementation, since the user does not need to understand Prolog. Additionally, no interaction with a command line interface is required. Users can just use the editor to define a privacy policy and get compliance feedback directly in the editor.

2. Completeness Checks. In addition to the compliance-checks performed by P-LPL, the editor warns the user when some information is missing or incomplete. This is not strictly a unique feature, but it is closely related to the compliance checks, hence we mention it here. Required fields are marked with an asterisk * and highlighted in red, when incomplete. A yellow triangle with an exclamation mark (next to the *SAVE Policy*-button in the main menu, **a**-region in Figure 5) warns the user before saving an incomplete policy. Red exclamation marks on the tiles in the main-area of the editor further highlight missing information. Red triangles with exclamation marks in the tabs (**b**-region in Figure 5) highlight in which tab information is missing.

The exclamation marks for missing information, as well as the question marks for helpful hints, provide a guided way of defining a privacy policy. We provide a more detailed description of the guided process in Section 3.5 below. There is no fixed order in which information must be entered, the users can enter the information in any order they like. The completeness checks ensure that the policy author enters all information required for a privacy policy.

No Expert Knowledge Required. The structured nature of P-LPL policies improves the usability of our editor compared to existing privacy policy editors and generators. There is no wall-of-text to be edited and managed. In combination with the guided process the number of domain experts required during the policy definition process is reduced to a single software engineer with knowledge about the system at hand. No legal expert is required during policy definition. Only after finishing the policy, a legal expert should check the quality of the policy and make adjustments where needed.

Importing Information from Software Engineering Artifacts. Compared to other privacy policy editors, our editor does not require the user to enter all information from scratch. Instead, the editor allows the user to import information about data processing from data-flow diagrams, as shown in (Leicht et al., 2023). This makes the policy definition process less tedious and reduces the risk of discrepancies between the policy and actual system behavior.

Structured Policy Representation. The structured nature of the policies also improves clarity, when a suitable user interface is used for presentation of the policy towards the data subject. This enhanced clarity can further improve the transparency of privacy policies, resulting in a more informed consent collection, compared to regular textual privacy policies.

As long as such structured representations of privacy policies are not standardized by the European Union, we also support policy authors in legal compliance by providing an export to textual policies.

3.2.3 Extensibility and Adaptability

In its current form, our editor is optimized for GDPR-compliant privacy policies. However, since the compliance-checks are separate from the user interface, the editor can be easily adapted to other legislations.

On the one hand, the compliance-checks can be replaced with ones of other legislations, e.g., COPPA (Federal Trade Commission, 1998) or HIPAA (Employee Benefits Security Administration, 2004). On the other hand, the editor itself could easily be adapted/extended to accommodate additional information. This may require to adapt the policy language (P-LPL) and the corresponding model of the policies, used by the editor.

A possible extension of the editor could be checking the compliance with company-internal policies. A company or business may have already defined policies that describe how data may be handled in-house. Such policies may be conflicting with the data processing stated in the privacy policy. Checking the compliance of a newly defined privacy policy with such company-internal policies may highlight misleading information. The misleading information could then be removed from the privacy policy, increasing transparency and correctness of the policy. Internal policies to be checked could for example be business processes, security/access control policies, or codes of conduct.

The import of policy relevant information from different sources can also be extended. Currently information can be retrieved from data-flow diagrams. For the future we plan to import information from business process models in the business process

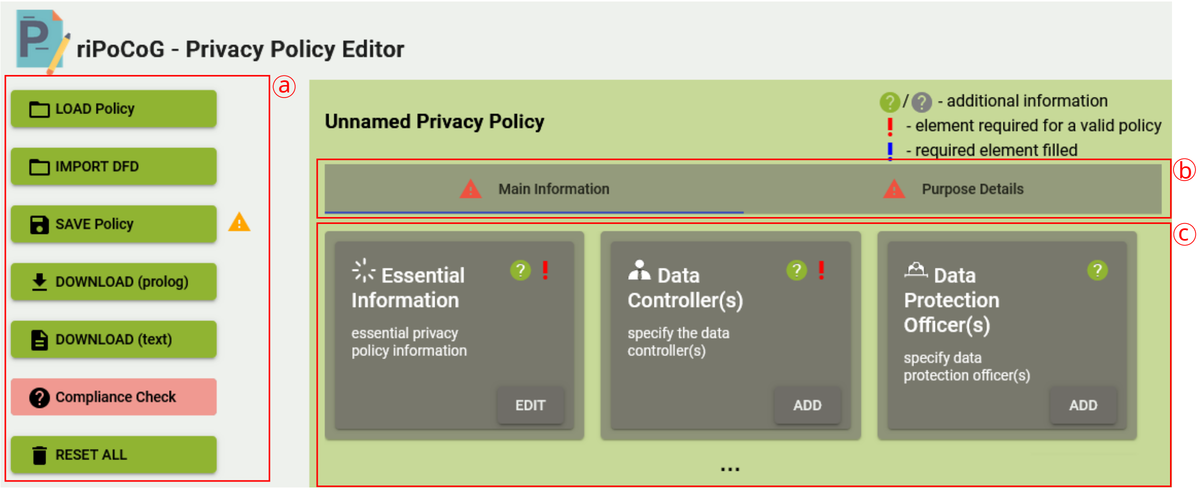

riPoCoG - Privacy Policy Editor
ⓐ
LOAD Policy
IMPORT DFD
SAVE Policy
DOWNLOAD (prolog)
DOWNLOAD (text)
Compliance Check
RESET ALL
Unnamed Privacy Policy
? / ? - additional information
! - element required for a valid policy
! - required element filled
ⓑ
Main Information
Purpose Details
ⓒ
Essential Information
essential privacy policy information
EDIT
Data Controller(s)
specify the data controller(s)
ADD
Data Protection Officer(s)
specify data protection officer(s)
ADD
...

Figure 5: GUI of our privacy policy editor with the three main regions: a) the main menu, b) main information and purpose tabs, and c) the main editor area (excerpt).

model and notation (BPMN[3]), see Section 6.2.

3.3 Development Challenges

In the development of our editor we faced some challenges, which we briefly discuss in the following.

The integration of the P-LPL compliance checker was a complicated task as P-LPL is implemented in Prolog. Available integration layers, which allow other programming languages to access Prolog, required us to adapt the usage of P-LPL.

Another challenge was the user interface (UI) design and the abstraction from the P-LPL language. The first version of the editor provided input fields for all information that could be stored in a P-LPL policy. This, however, made the UI cumbersome to use and required the user to enter some information repeatedly. In the next revision of the UI we abstracted from P-LPL and only ask the user for input regarding important policy information and generate many P-LPL fields automatically from information already entered by the user.

The export of textual privacy policies was inspired by the work of Mohammadi et al. (Mohammadi et al., 2020). Here the challenge lies in the creation of well structured and fitting patterns, to be instantiated using the P-LPL policy from our editor.

3.4 Implementation

Our editor is implemented as a web-based tool, which can also be deployed locally as a stand-alone application. We use Angular[4] for the front-end and Python, which invokes the P-LPL Prolog calls, for the back-end.

The editor implements the P-LPL privacy policy model in an object-oriented manner. This ensures compatibility between our editor and the existing P-LPL framework.

The editor has a main-menu (**a**-region in Figure 5) on the left-hand side of the GUI. This menu contains actions that can be performed on the whole privacy policy, e.g., import and export of policies, as well as the P-LPL compliance check, and a reset-button for starting over. The *save*-button downloads the policy as an XML-file, whereas the download buttons export the policy as Prolog-file or a textual policy. The textual privacy policy is generated using textual patterns. The right-hand side of the GUI contains the main editing tiles (**c**-region in Figure 5), clustering information according to the structure of P-LPL policies. We make use of tabs (**b**-region in Figure 5) to divide the contents of a privacy policy into two types of information. In the *Main Information*-tab we ask the user for general information about the policy itself, data controllers, data protection officers, data subject rights, supervisory authorities, the data collected, possible data recipients, the purposes for which data will be collected and processed, as well as the purpose hierarchy, structuring the purposes. The main editor-area is divided into tiles structuring the content of the privacy policy. Using tiles provides a way of getting an overview of the policy, before each information category can be opened in a separate dialogue.

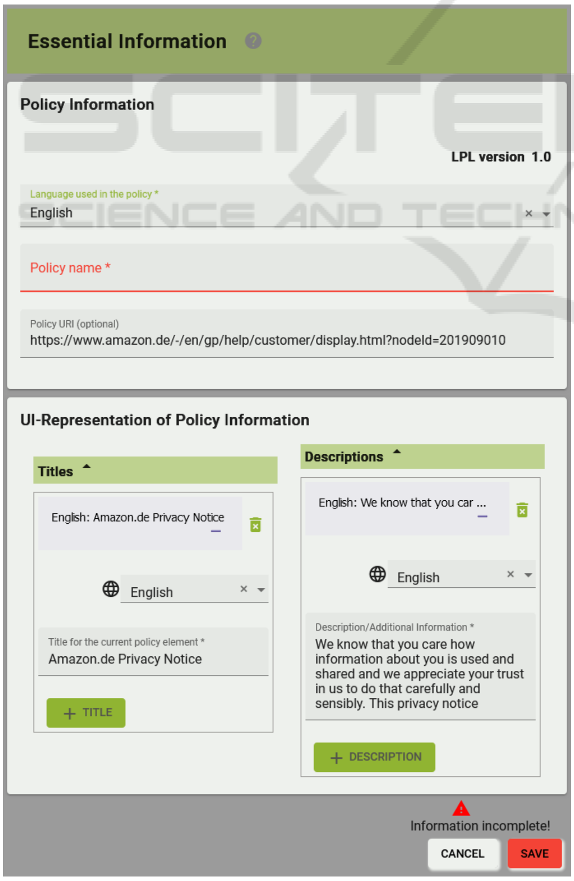
We use a similar tiled overview in the *Purpose Details*-tab. Here we ask for further details regarding

[3] https://www.omg.org/spec/BPMN/ (accessed 2023-12-12)

[4] https://angular.io/ (accessed 2023-12-04)

each of the purposes previously defined in the main information tab. The user should state which data are going to be collected and processed for a given purpose, recipients of the data, the retention period of the data collected for a given purpose, and the legal basis on which the purpose is based. Additionally, the user can supply privacy models, information regarding automated decision-making, and pseudonymization methods used on the data of this purpose.

Each tile of the editor can be used to open a specific input form. At the top of these input forms the main information required by this category is requested. Figure 6 shows the input form for the *Essential Information* category, filled with information from the Amazon.de privacy policy. In this example: the language of the policy, a name for the policy, and optionally a URL pointing to the textual representation of the privacy policy can be entered. Required fields are highlighted with an asterisk (*) and colored red when no information is entered (e.g., *Policy name* in Figure 6). Below the main information many policy elements provide input fields for a title and a description. This information can be provided in diverse languages and will be used by privacy policy interfaces, when presenting the privacy policy to the data subject. Since no P-LPL data subject user interface has been developed yet, this information is considered optional. However, it is considered good practice to fill out the titles and descriptions, so that the information can easily be used by user interface developed at a later time. The descriptions can be used to provide reasoning for different elements, for example why some data need to be collected or processed.



Essential Information

Policy Information

LPL version 1.0

Language used in the policy *
English

Policy name *

Policy URI (optional)
https://www.amazon.de/-/en/gp/help/customer/display.html?nodeId=201909010

UI-Representation of Policy Information

Titles

English: Amazon.de Privacy Notice

English

Title for the current policy element *
Amazon.de Privacy Notice

Descriptions

English: We know that you car ...

English

Description/Additional Information *
We know that you care how information about you is used and shared and we appreciate your trust in us to do that carefully and sensibly. This privacy notice

Information incomplete!

Figure 6: Input form for the essential privacy policy information.

The use of icons and colors further improves usability of our editor, as the user can get a quick understanding of missing/wrong information. Advanced users may also navigate the policy quicker when focusing on icons instead of reading lengthy titles.

3.5 Usage

As already mentioned earlier, there is no fixed step-by-step method of defining a privacy policy using our editor. Policy authors are free to enter information in any order they may prefer.

The general procedure of defining a privacy policy in our editor is as follows:

1. Optionally, import information from data-flow diagrams, created using the DFD-editor presented in (Leicht et al., 2023).
2. Enter some information in the *Main Information*-tab until all required elements have been filled. Alternatively, you can proceed with step three intermittently, once you have defined at least one purpose. You can return to this step at any time.
3. Switch to the *Purpose Details*-tab to enter information about the purposes for which data are collected and processed. If you haven't completed step two, yet, you can return to step two intermittently.
4. When all required policy elements are filled, check the GDPR-compliance of the policy by clicking the *Compliance Check*-button in the main menu.
5. If something is marked as non-compliant, go back to editing this policy element and return to Step 3.
6. Finally, if required, export the policy as a textual privacy policy or save the P-LPL file.

This process gives guidance to the user while at the same time allowing greatest possible flexibility. Next, we take a look at the evaluation of our privacy policy editor.

4 EVALUATION

To evaluate our privacy policy editor, we applied a two-phase approach. In the first phase, we asked computer science students for feedback using the questionnaire described in the next section. Afterwards, we improved our editor by adding features and changing the layout of some of its components. In the second phase, we asked five computer science researchers to evaluate our editor according to the same questionnaire. After we analyzed the results of the second phase, we performed some further improvements, resulting in the privacy policy editor we presented above. Some of the feedback received in phase two will be addressed in future updates of the editor.

We used the Technology Acceptance Model (TAM) by Davis (Davis, 1985) for the evaluation of our privacy policy editor. Using an adapted TAM-questionnaire we evaluated the first three of the following hypotheses:

H1 The privacy policy editor is easy to use.

H2 The privacy policies created with the editor have a high quality.

(H2.1) The resulting policies are well structured.

(H2.2) The resulting policies are easy to comprehend.

H3 The editor is useful for the creation of GDPR-compliant privacy policies.

(H4) Privacy policies created with our editor have an improved accuracy compared to state-of-the-art textual privacy policies.

(H5) Using our privacy policy editor improves the transparency conveyed by the privacy policies.

Although we consider hypotheses (H2.1) and (H2.2) subordinate to H2, they cannot be evaluated using the TAM questionnaire. The quality of the output of the editor is analyzed in the TAM questionnaire, but not in enough detail to support nor reject hypotheses (H2.1) and (H2.2). These hypotheses need to be evaluated separately, preferably by domain experts, using a different evaluation methodology.

Hypotheses (H2.1), (H2.2.), (H4), and (H5) will be evaluated in future work. In this paper we focus on the privacy policy editor itself.

4.1 Questionnaire

The questionnaire we created for the evaluation of our editor is a combination of the TAM3 questionnaire from the original TAM by Davis (Davis, 1985) and the updated questions as described in the later revision of the TAM (Davis, 1989). As suggested by Davis we adapt the questionnaire by leaving out some categories of questions as well as adapting individual questions to our privacy policy editor.

Our questionnaire contains the following four categories of questions: *Overall Evaluation*, *Perceived Characteristics of Output*, *Perceived Ease of Use*, and *Perceived Usefulness*. Each category includes at least two questions.

The original TAM contains two more categories. However, since we evaluated our editor with students and computer science researchers, which both are probably not going to work with the editor in their jobs, we removed the category *Anticipated Use* from our questionnaire. And, since the editor is considered a professional tool, we also excluded the category *Anticipated Enjoyment of Use*.

The original TAM questionnaires used seven-value scales, e.g., from *extremely likely* to *extremely unlikely* including the neutral option *neither*. To reduce the chance of participants answering neutral for all questions, we reduced the number of available options from seven to six, removing the neutral option *neither*.

Depending on the question asked, we used one of the following scales:

- bad → good
- unlikely → likely
- harmful → beneficial
- foolish → wise
- unconfident → confident
- low → high
- negative → positive

Each of the scales takes the levels of: extremely A - quite A - slightly A - slightly B - quite B - extremely B, where A and B are the values of the scales listed above.

We asked the participants 28 questions overall. 22 of these were adapted from the TAM and four questions were of demographic nature, including age, gender, and field of study. Additionally, we asked for information regarding prior professional experience in the field of computer science as well as privacy. We also provided free-text inputs for the participants to give feedback regarding bugs they might encounter during the usage of the editor, as well as comments and general feedback for the improvement of the editor. A complete overview of the questionnaire is attached in the appendix of this paper.

In the following we take a look at the circumstances and results of each of the evaluation phases.

4.2 Phase 1

In the first phase of our evaluation we asked nine computer science students to participate in our survey. The participants have at least a bachelor's degree in computer science. None of the participants stated any prior experience in the field of privacy. However, they received a lecture of one hour on privacy and the GDPR prior to participating in our evaluation survey.

Demographics. The participants were between 20 and 28 years old, resulting in an average age of around 24 years. The group of participants consists of six male and two female Master students. One person did not answer the question regarding their gender.

TAM Results (Average).

- Overall Evaluation: **slightly positive**
- Perceived Characteristics of Output (quality): **quite high**
- Perceived Ease of Use: **slightly likely**
- Perceived Usefulness: **slightly likely**

The participants rated the editor *slightly positive* (4.47/6) in the overall evaluation. The quality of the resulting privacy policies was considered *quite high* (4.83/6). A main concern of the first evaluation phase was usability, as the participants rated ease of use with *slightly likely* (4.11/6). The value *slightly likely* stems from the way questions are formulated in the TAM (cf. questionnaire in the appendix).

Bug Reports. The first phase identified some major bugs, like data loss when accidentally refreshing the page, as well as some performance issues when editing long lists of policy elements. *We fixed all bugs that were identified, before starting the second evaluation phase.*

Feature Requests and General Feedback. The main concern in the general feedback section has to do with automating more of the input. Some information had to be entered manually although it could be deduced from other inputs. *We implemented some automation for these inputs, to reduce the amount of manual input by the policy author.*

In the first phase, the help/descriptions were lacking in information, hence the participants wanted more information contained in the help texts. Main information tiles and purposes were listed on a single page, and participants wanted a better overview. *This is the reason for having two tabs, separating main information from purpose details.*

We also considered further minor feature requests and feedback for the development of the editor.

We concluded this phase of the evaluation with an overall slightly positive result. Taking into consideration a list of feedback comments and bug reports, we improved the editor before conducting the second phase of the evaluation of our editor.

4.3 Phase 2

In the second phase we interviewed five computer science researchers on an improved version of the editor. Two of the researchers were Master students, one had a master's degree in applied computer science, and two researchers had PhDs in computer science. Four male and one female researcher participated in this phase.

This time we focused on the free-text feedback of the participants instead of strictly requiring them to fill out our survey questionnaire. From the four questionnaires returned we calculated the following evaluation results:

TAM Results (Average).

- Overall Evaluation: **quite positive**
- Perceived Characteristics of Output (quality): **quite high**
- Perceived Ease of Use: **slightly likely**
- Perceived Usefulness: **slightly likely**

The overall evaluation was more positive in the second phase. However, the ease of use still was a major concern with 3.57 out of 6 points. This is also reflected in the feature requests for improved usability, which we present below.

Bug Reports. The participants of the second phase also identified some bugs in the editor, e.g., crashes of the back-end. *We fixed all bugs and considered a number of feature requests and general feedback, as we describe below, for the version of the editor we present in this paper.*

Feature Requests and General Feedback. The P-LPL required field *title (HEAD)* and *description (DESC)* are tedious to fill out for every element of a privacy policy. Users of the editor do not see/know the use of this information. *We plan to semi-automatically generate this information, for improved usability.* Some input is repeated throughout the privacy policy, for example, information regarding entities that fill in different positions at the same time. A single person could be considered data controller and data protection officer at the same time. *Making such information reusable is our goal for future updates of the editor.* We provide guidance using help-pages for each policy element that the policy author needs to provide. The participants wanted additional information directly inside the forms, e.g., by providing mouse-over texts or additional information next to the forms. *This idea is also planned for a future update of the editor.* Finally, the form controls *save* and *cancel*, currently positioned at the bottom of each form, could be optimized by also providing them at the top of the forms. When editing long lists of elements, scrolling down to the bottom of the page slows down the user of the editor.

4.4 Evaluation Conclusion

We now summarize the results of our evaluation of our privacy policy editor by taking another look at the hypotheses stated in Section 4.

Hypothesis *H1* regarding the usability of our editor can be evaluated using the questions of the category *Perceived Ease of Use*. The results of the surveys show that the participants have a neutral to slightly positive opinion on the usability of our editor. This neither supports nor rejects *H1*.

Our survey only considers the main hypothesis *H2* in its questionnaire, *(H2.1)* and *(H2.2)* need to be considered in future work. The quality of the privacy policies is evaluated using the category *Perceived Characteristics of Output*. The surveys show that the perceived quality of the resulting privacy policies is quite high, supporting *H2*.

Hypothesis *H3* can be evaluated using the questions of the category *Perceived Usefulness*. Regarding the usefulness of our editor, we received weak results tending towards the positive side of the scale. This neither supports nor rejects *H3*.

Overall, we can conclude the evaluation with a slightly positive result. The results of the surveys tend towards the positive side of the scales for each of the three hypotheses evaluated in this paper. However, only one of the hypotheses is significantly supported by the survey results (*H2*). The other two hypotheses are only supported slightly, hence we consider these hypotheses as neutral and propose further evaluation in the future. A next level of evaluation should include domain experts, to find out whether the editor is useful in its intended environment and use-case, as well as to find out whether *(H2.1)* and *(H2.2)* are supported.

Hypotheses *(H2.1)*, *(H2.2)*, *(H4)*, and (H5) are concerned with the characteristics of the privacy policies that are created with our editor. The characteristics of the policies need to be evaluated in a separate study. This study should also consider state-of-the-art textual privacy policies in comparison to P-LPL privacy policies.

5 RELATED WORK

There exist many different online privacy policy generators that generate privacy policies according to some user input. Termly Inc.'s generator[5] is an example of a free privacy policy generator. The user is interviewed using a comprehensive questionnaire and the information entered is used to generate a complex textual privacy policy. The Termly generator has the benefit of covering a broad range of data protection legislations. However, it only gives hints concerning necessary information. The user can skip entering required information, resulting in a non-compliant privacy policy. This sort of policy generators is also very limited in the export of the policy. Most online generators provide the resulting policy in a textual form, meaning that other representations of the policies are not possible without considerable manual effort. Our editor saves the policy using P-LPL, making it possible to visualize the policy in many different ways, which may be developed in the future.

For other kinds of policies, for example access control policies, commercial products including policy editors are available. An example of such an access control system is the WSO2 Identity Server[6], which also includes an eXtensible Access Control Markup Language XACML editor. Due to the limited access to these commercial systems, we cannot compare these to our editor.

Gerl and Meier developed a policy editor for the original Layered Privacy Language (LPL) in conjunction with extending LPL (Gerl and Meier, 2019). Their editor is limited to LPL and, thus, cannot be combined with P-LPL for compliance checks. Hence, their editor can only check for required fields or for ill-formed input like letters in phone numbers or incomplete e-mail addresses. Compliance checks, as they are performed by P-LPL in our editor, are not part of Gerl and Meier's work.

Dittmann et al. proposed a privacy compliance architecture that ensures the compliance of data transfers with applicable legislation (Dittmann et al., 2022). This architecture checks which legislation is applicable for a given data-flow and ensures that the data-flow is performed in compliance with the said

[5] https://termly.io/ (accessed 2023-12-07)

[6] https://wso2.com/identity-server/(accessed 2023-12-13)

legislation. The presented architecture was developed in the context of connected-vehicle services, which however does not limit its use. The architecture could be adapted to and included in the PriPoCoG framework, as an additional compliance measure for changing legislative contexts.

6 CONCLUSIONS

We presented our privacy policy editor and now discuss our contributions and take a look at future work.

6.1 Contributions

Policy authors are supported during the policy definition process by the GDPR-compliance feedback from our editor. This enables persons from other domains than the legal department to specify privacy policies.

Data subjects profit from well-structured and GDPR-compliant privacy policies. With the improved comprehension of privacy policies data subjects can make a more informed decision when accepting or rejecting some data processing.

The P-LPL policies defined with our editor can also be enforced using P2BAC (Leicht and Heisel, 2023), where both sides, data controllers as well as data subjects benefit from this enforcement.

When policy information is imported from DFDs the accuracy of the privacy policies is increased. Hence, there are less discrepancies between the data handling described in the policy and the actual data handling performed by the system.

Going back to our research questions presented in Section 1 we provide the following answers. Concerning **RQ1**, the GDPR-compliance of data controllers, our editor provides compliance feedback to the policy authors. This will improve the overall GDPR-compliance of data controllers by reminding them that some data processing might not be compliant. Regarding **RQ2**, the usability of the PriPoCoG-framework, we provide a user interface for the definition of P-LPL policies, which enables policy authors without prior knowledge of P-LPL in creating privacy policies. We conclude that our editor provides answers for both of the research questions.

Next, we take a look at future work around the editor and the PriPoCoG-framework.

6.2 Future Work

Future updates of the editor could improve usability and add new features. Similar to the DFD-import functionality described in Section 3.5, an import functionality for business process models could useful, as these models contain a lot of information about business processes that might be relevant for a business' privacy policy.

To further improve the comprehension of the resulting privacy policies, future work should focus on the presentation of policies towards data subjects. Using P-LPL as a policy exchange format, different kinds of policy interfaces could be developed and evaluated. A service provider could provide multiple views for its privacy policy, so that data subjects could select the best view for the best comprehension.

As already mentioned in Section 4, further evaluation of the privacy policies created using our editor is required in the future. This evaluation should include the comprehensibility (H2.2), as well as the accuracy of the policies (H4), compared to commonly used textual privacy policies.

The PriPoCoG framework and especially P-LPL could also be extended and adapted to other legislations.

REFERENCES

Davis, F. D. (1985). *A technology acceptance model for empirically testing new end-user information systems: Theory and results*. Thesis, Massachusetts Institute of Technology.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, pages 319–340.

Dittmann, G., Giblin, C., Osborne, M., and Rahul, R. (2022). Automating privacy compliance in the decentralized enterprise. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 2218–2223. IEEE.

Employee Benefits Security Administration (2004). The Health Insurance Portability and Accountability Act (HIPAA).

European Parliament and Council of the European Union (2016). Regulation 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal of the European Union*, L119:1–88.

Federal Trade Commission (1998). Children's Online Privacy Protection Rule ("COPPA").

Gerl, A. (2020). *Modelling of a privacy language and efficient policy-based de-identification*. Thesis, Universität Passau.

Gerl, A. and Meier, B. (2019). The layered privacy language art. 12–14 GDPR extension–privacy enhancing user interfaces. *Datenschutz und Datensicherheit-DuD*, 43(12):747–752.

Leicht, J. and Heisel, M. (2023). P2BAC: Privacy policy based access control using P-LPL. In Mori, P., Lenzini, G., and Furnell, S., editors, *9th International Conference on Information Systems Security and Privacy*, pages 686–697. SciTePress.

Leicht, J., Heisel, M., and Gerl, A. (2022). PriPoCoG: Guiding policy authors to define GDPR-compliant privacy policies. In *International Conference on Trust and Privacy in Digital Business*, pages 1–16. Springer.

Leicht, J., Wagner, M., and Heisel, M. (2023). Creating privacy policies from data-flow diagrams. In *Computer Security. ESORICS 2023 International Workshops*.

Mohammadi, N. G., Leicht, J., Goeke, L., and Heisel, M. (2020). Assisted generation of privacy policies using textual patterns. In *ENASE*, pages 347–358.

APPENDIX

The questionnaire contained the following questions. Although we list them here by TAM category, they were presented to the participants in a mixed order. The confidence question was repeated for each page of the questionnaire (three times), except for the free text feedback part. Participants were asked to answer these questions from the perspective of being a Data Controller, who uses PriPoCoG to define a GDPR-compliant privacy policy for their online service. The answer scales were inverted for some of the questions, so that participants had to actively think about placing a cross at the intended position on the scale.

Overall Evaluation.

1. Using PriPoCoG in my job would be ...
   extremely bad → extremely good
2. Using PriPoCoG in my job would be ...
   extremely harmful → extremely beneficial
3. Using PriPoCoG in my job would be ...
   extremely foolish → extremely wise
4. Using PriPoCoG in my job would be ...
   extremely negative → extremely positive

Perceived Characteristics of Output.

5. Assuming I were to use PriPoCoG, the quality of the privacy policy I would get would be high. —
   extremely unlikely → extremely likely
6. Using PriPoCoG, the effectiveness of the finished privacy policy would be:
   extremely low → extremely high

Perceived Ease of Use.

7. Learning to operate PriPoCoG would be easy for me. —
   extremely unlikely → extremely likely
8. I would find it easy to get PriPoCoG to do what I want it to do. —
   extremely unlikely → extremely likely
9. My interaction with PriPoCoG would be clear and understandable. —
   extremely unlikely → extremely likely
10. I would find PriPoCoG to be flexible to interact with. —
    extremely unlikely → extremely likely
11. It would be easy for me to become skillful at using PriPoCoG. —
    extremely unlikely → extremely likely
12. I would find PriPoCoG easy to use. —
    extremely unlikely → extremely likely

Perceived Usefulness.

13. Using PriPoCoG in my job would enable me to accomplish tasks more quickly. —
    extremely unlikely → extremely likely
14. Using PriPoCoG would improve my job performance. —
    extremely unlikely → extremely likely
15. Using PriPoCoG in my job would increase my productivity. —
    extremely unlikely → extremely likely
16. Using PriPoCoG would enhance my effectiveness on the job. —
    extremely unlikely → extremely likely
17. Using PriPoCoG would make it easier to do my job. —
    extremely unlikely → extremely likely
18. I would find PriPoCoG useful in my job. —
    extremely unlikely → extremely likely

Confidence.

How confident are you in the ratings that you have made on this page? —
extremely unconfident → extremely confident