What's Your Purpose? An Approach to Incorporating GDPR Purposes into Requirements Analysis

Evangelia Vanezi, Georgia Kapitsaki and Anna Philippou
Department of Computer Science, University of Cyprus, Cyprus

Keywords: GDPR Purpose, Privacy by Design, System Requirements, Use Case Diagrams, Sequence Diagrams.

Abstract: Protecting personal data within software systems is crucial, and as such, several privacy regulations have been enacted, one being the EU's General Data Protection Regulation (GDPR). While GDPR emphasizes "Purpose Limitation" for rightful personal data handling, the concept of purpose lacks clarity in software development practices. Building on our previous work on DiálogoP, which supports the definition of formal processing purposes, this study introduces purpose-aware system requirements. We present AnálisisP, a methodology for integrating *processing* purposes into the software engineering requirements analysis phase and visual representations of these enhanced requirements by extending the Unified Modeling Language (UML) Use Case and Sequence diagrams. We show how our approach enables the integration of AnálisisP with DiálogoP towards formal models whose compliance with processing purposes is rigorously validated. Additionally, we showcase how the proposed extended diagrams assist in addressing further GDPR-related system design queries.

1 INTRODUCTION

The Problem. Protecting personal data within software systems is crucial, and as such, several privacy regulations have been enacted, including the California Consumer Privacy Act (CCPA) (Goldman, 2020) and more recently the European Union (EU) General Data Protection Regulation (GDPR) (European Parliament and Council of the European Union, 2015). The GDPR stands as a pivotal framework, empowering individuals with rights over their data and imposing strict guidelines on organizations regarding its collection, processing, and storage. One very important principle of the GDPR, defined in Article 5, is 'Purpose Limitation', indicating that data collected should be handled only in ways explicitly stated and agreed beforehand between the user and the system. In Software Engineering, *purpose* is meant to characterize the usage of personal data by the system entities processing, and therefore, it should comprise a crucial part of a system's functional requirements. Another important principle is *Privacy by Design* (PbD), which advocates that privacy should be incorporated into systems by default and should be a priority from the beginning of a system's design. Even though the above are significant for privacy, the notion of purpose is still not clearly defined, and software engineers do not explicitly address it during the development process, as some developers indicated in a previous work (Alhazmi and Arachchilage, 2021), let alone in a proactive way, as indicated by PbD.

Our Contribution. In our previous work (Vanezi et al., 2020), we presented DiálogoP, a methodology including a formal language for defining processing purposes of systems as the communication exchanges between a system's entities, and a tool for supporting this definition through a visual interface. In this work, we formalise the underlying layer on which DiálogoP is based: the purpose-aware system requirements. We discuss and present our definition of what *purposes* are in software engineering. We present AnálisisP, a methodology applied during the requirements analysis step of the software engineering process, integrating processing purposes with the requirements. Purpose-aware requirements can be depicted through textual format; however, they are better demonstrated through the use of UML diagrams. We present an extension for visualising such requirements for two of the most popular diagrams: (i) Unified Modeling Language (UML) Use Case Diagrams (Gemino and Parker, 2009) and (ii) UML Sequence Diagrams (Booch et al., 1996). This notation allows the representation, definition and handling of processing purposes in software engineering. We then present the direct mapping and integration of AnálisisP with DiálogoP in a framework towards

Vanezi, E., Kapitsaki, G. and Philippou, A.
What's Your Purpose? An Approach to Incorporating GDPR Purposes into Requirements Analysis.
DOI: 10.5220/0012474400003648

In *Proceedings of the 10th International Conference on Information Systems Security and Privacy (ICISSP 2024)*, pages 907-914
ISBN: 978-989-758-683-5; ISSN: 2184-4356

formally validated purpose-compliant system models. Additionally, we demonstrate how our extended purpose-aware diagrams can assist in addressing further important GDPR-related system design queries. We validate our methodology with a running example. Finally, our conclusions and future work are discussed. Our overall aim is to study purpose and propose a solution for formally integrating it into software engineering following the principle of PbD.

Structure. In Section 2 we present our definition for purpose and then in Section 3 we discuss related work. In Section 4, we present our main contribution, the AnálisisP methodology, in detail, presenting the extended use case diagrams in Subsection 4.2 and the extended UML sequence diagrams in Subsection 4.3. Following, we present the integration with DiálogoP in Section 5, the additional privacy queries in Section 6, concluding with a discussion in Section 7.

2 WHAT IS A PURPOSE?

The notion of *'purpose'* was initially mentioned in the declaration of the *Protection of Personal Data* as one of the fundamental rights of the EU (European Parliament and Council of the European Union, 2012), by declaring that "*data must be processed fairly for specified* purposes". Within the GDPR, purpose is mainly mentioned in Article 5, 'Purpose Limitation' stating that "personal data shall be collected for specified, explicit and legitimate *purposes* and not further processed in a manner that is incompatible with those *purposes*". Additionally, it is mentioned in many more articles of the GDPR.However, the regulation does not explicitly define what a purpose is, despite constituting such an important notion.

Access Control, Roles and Permissions. Although GDPR and purpose is now taken much into account, most existing studies that provide support for purpose control are not appropriate for guaranteeing that data is not going to be reused for purposes other than those intended during the collection of data (Byun et al., 2005; Yang et al., 2007). In the work of (Kouzapas et al., 2016), the authors focus on user roles and formal verification of permissions of access to specific data, capturing the sense of purpose at a certain level, but do not control how these permissions and roles will be used within the system. This is because purpose contrasts with standard access control (Basin et al., 2018), which regulates who may carry out an operation in a system independently of context. For example, if a courier service is allowed to access a client's mobile phone number, then they will be able to do it for any purpose, both intended (delivery notification) and unintended (advertisement). Access control needs to be related to an explicitly stated purpose. However, the question remains, how is a purpose defined?

Formal Semantics. Various works treat purposes using informal or semi-formal descriptions bearing little or no semantics (Barth et al., 2006). This informal treatment of the notion of purpose prohibits a precise analysis to verify whether a system complies with its purpose-aware specification. The research community recognized this shortcoming, and recently, various works have been concerned with providing semantic foundations for the notion of purpose, associating purposes with the actions towards a purpose. In one such approach, Markov Decision Processes were used in (Tschantz et al., 2011) and (Tschantz et al., 2012) and a formalism based on planning was proposed for auditing systems against privacy policies. This approach is also adopted by (Basin et al., 2018), identifying a purpose with a business process, and use formal models of inter-process communication to audit or derive privacy policies. Furthermore in (Jafari et al., 2011), the authors define a semantic model for purpose-based privacy policies, a modal logic and the corresponding model checking algorithm to verify whether a particular system complies with them. Other related works include (Riahi et al., 2017) specifying purposes as workflows modeled by Petri nets and model-checked against actor models, and (De Masellis et al., 2015), proposing semantics of purpose-based privacy policies in temporal logic and defines a run-time monitoring methodology.

Our Definition. Our proposed approach is complementary to other similar works, based on the observation that a purpose can be broken down and expressed as a simple sequence of actions describing the allowed data processing by the system entities during their internal activities (locally), and interactions between them (globally). For example, purpose "Notification for delivery" permits the usage of the client's mobile phone number. To define this purpose, we should explicitly state which actions of which entities preceded this processing (e.g., the shop informed that the order is ready for delivery) and which actions are to follow this processing by which entities, towards fulfilling the purpose (a message is sent from the notifications station to the client regarding the upcoming delivery). With this definition, we aim to fulfil two objectives: (i) to avoid misuse of personal data (as in access control) and (ii) to be able to formally validate the compliance of a system to its processing purposes by checking the actions and interactions of the system entities, in comparison with the defined purposes. We then propose merging purpose directly

into the functional requirements sequence of flow. We deploy a simple example to demonstrate a compliant and a non-compliant system.

Example: Ordering in an e-Shop. Let us assume the following system requirement *"The client can place orders through the e-shop that will be delivered to the provided address"*. This specification does not explicitly restrict the use of the client's personal data for any other purpose beyond delivering the products or only after placing an order. We define the user's address as their personal data. We rephrase the above requirement to include permission-based processing of the personal data as follows: *"The cart entity can disclose the client's provided address to the delivery company."*. This specification allows the user's personal data to be sent to the delivery company anytime. We rephrase the requirement in a purpose-aware manner, as follows: *"The client's provided address can only be sent from the cart to the delivery company after the user confirms an order, and then a delivery needs to follow."* Based on this requirement, we examine a compliant and two non-compliant system executions: the first is non-compliant as the address is disclosed before confirming the order, while the second is non-compliant as the interaction ended without resulting in the order delivery.

Compliant System Execution. The client adds objects to the cart → The client fill-in their address → The client confirms the order → The cart discloses the client address to the delivery company → The delivery company delivers the order → end.

Non-Compliant System Executions. (1) The client adds objects to the cart → The client gives their address → The cart discloses client address to the delivery company ≪violation≫; **(2)** The client adds objects to the cart → The client fill-in their address → The client confirms the order → The cart discloses the client address to the delivery company → end ≪violation≫

3 RELATED WORK

Several works have been discussing and supporting the incorporation of *processing purposes* in the Software Engineering process, in the requirements phase, with the use of popular diagrams.

Data Flow Diagrams (DFDs). An important step towards integrating privacy and purpose into a system technical design, and specifically into DFDs, with the aim of (i) supporting PbD, (ii) validating the system design in comparison to the textual privacy regulation prescriptions, and (iii) reducing the semantic gap between engineers and law, was done in (Antignac et al., 2016). They extend the DFD notation to support specific technical privacy concepts, resulting in privacy-aware DFDs (PA-DFDs). They add an annotation *purpo* in each process, to reflect the way regulations expect all kinds of personal data processing to be associated with a purpose. They distinguish data flows of "personal data" and generic data, and they add the concept of personal data ownership by connecting personal data flows with specific data subject entities. This work deals with processing purposes of "personal data", however, they consider them a single "textual" description in a high level of abstraction. This work is extended in (Antignac et al., 2018) with model transformations, however the handling of processing purposes remains the same as in the initial work. In (Alshareef et al., 2021b) they provide an explicit algorithm and a proof-of-concept implementation to transform DFDs into PA-DFDs. In (Alshareef et al., 2021a) they are concerned with formal refinement for PA-DFDs.

In (Alshareef et al., 2022) the work on PA-DFDs is complemented with a focus on "Purpose Limitation" that was previously handled abstractly. Aiming to model "Purpose", they extend DFDs with purpose labels on data flows to represent the intended purpose for which a piece of data is to be used and privacy signatures on activators to model the impact of processing and storage on these purpose labels. They define a formal mathematical framework for (1) annotating DFDs with purpose labels and privacy signatures, (2) checking the consistency of labels and signatures, and (3) inferring labels from signatures. They also implement their theoretical framework in a proof-of-concept tool. Once again, purposes, even handled in more detail, are still addressed as textual labels.

Business Process Model and Notation (BPMN). Many works are concerned with validating GDPR compliance of Business Process Models like in (Kala, 2019), (Matulevičius et al., 2020), (Sing, 2018), where a GDPR UML Model is adopted from the work in (Tom et al., 2018), in which (among others) a 1-to-many relationship is shown between "Consent" and "Purpose", and a many-to-many relationship is shown between "Purpose" and "Data Processing". Also, the ownership of "Personal data" from a "Data Subject" is modelled. The BPMNs should be designed following the UML Model restrictions, e.g., each data processing should be connected to at least one purpose. Again, purposes are defined on a more abstract and textual level.

In (Basin et al., 2018), the authors suggest that a business process model, by its very nature, explicitly represents one or more purposes and specifies at what points data is collected and used. They show how for-

mal models of inter-process communication can be used to audit or even derive privacy policies. Each purpose is represented by the name of the business process prescribing the actions that use the personal data towards that purpose. In this case, Purposes are defined similarly to how we define them. However, BPMNs are not extended to highlight personal data or their ownerships on the same model. Instead, a process collection is defined to show more abstractly which process collects and uses each personal data. An algorithm is presented to infer privacy policies ("we use d for p") based on data usage.

In (Petković et al., 2011), they propose a purpose representation model, which connects each intended purpose of data (included in the privacy policy) to a business model and detects privacy infringements by determining whether the data have been processed only for the intended purpose, by determining whether the audit trail is a valid execution of the organizational processes representing the purposes for which data are meant to be used. Similar wise to our work, they advocate that it is necessary to extend the current preventive approach by implementing mechanisms for verifying the actual use of data. However, in contrast, they do not perform the validation on system models but in audit trails collected from the systems logs, thus they do not follow the principle of PbD.

4 AnálisisP METHODOLOGY

In this section we present AnálisisP, a methodology for enhancing a system's functional requirements with processing purposes, i.e. how and why personal data are used by each system entity, resulting in the purpose-aware system requirements. The methodology includes the following steps:

- *Step 1.* Define the system functional requirements and define the set of all system entities, e.
- *Step 2.* Define the system purpose (how entities handle personal data and for which reason) in a textual format.
- *Step 3.* Integrate purpose (from *Step 2*) with functional requirements (from *Step 1*) in an allowed sequence of actions, visualising them using the extended use case and sequence diagrams proposed in this work.

Example: a Simple Task Management Application - Requirements. To demonstrate our methodology, we deploy a simple case study. Table 1 lists the requirements for a simple task management application.

Table 1: Example System Requirements.

1	Users should be able to create a new task
2	Users should be able to view a list of all their tasks.
3	Users should be able to edit and update existing tasks.
4	Users should receive notifications for approaching task deadlines.
5	The application should support multiple users with individual task lists.

The set of entities for this system is defined as e = {User, Authorisation, Tasks, Notifications, DB}. A User is an external entity, while Tasks, Notifications, Authorisation and DB are internal entities of the system. This completes *Step 1*.

4.1 Defining the System Purpose

In order to define the system's purpose (*Step 2*), one needs to: (1) List all personal data (2) For each entity of the system, e_x, define **which** personal data they will be providing, collecting, and processing, and **how** exactly they will be processing them (define preceding and succeeding actions), and if needed define *under which circumstances (i.e., conditions)* this will be done.

Example: a Simple Task Management Application - Purpose. For our running example, we (1) define the following personal data: fullname, email address, username, password.

We then proceed with (2) defining the following simple description of the processing purpose, omitting some details for the sake of brevity:

- A User will be providing their username and password during the login process to the Authorisation entity, expecting to either receive an authorisation message and proceed with accessing their tasks, or a denial message and abort the system.
- Authorisation entity receives the personal data sent by the User during the login process, checks them with the DB and either grants or denies access.
- The Tasks entity, receives a request from a User accompanied by the user unique identifier (username), checks with the DB, and returns a list of the tasks that are owned by the specific user. The User can only edit and add tasks on this list.
- The Notifications entity, in case a deadline is approaching, retrieves from the DB the user's email address and then uses the email address to send a message to inform the User for the upcoming deadline.

The above is a strict specification of the processing purpose to respect privacy. For example, the Notifications entity purpose explicitly states that the retrieval and use of the user's email address should only occur when there are upcoming task deadlines. Therefore, if the system retrieves or uses the email

address in other scenarios, or when there are no approaching task deadlines, it would be considered non-compliant. We proceed to present with the proposed visualisations.

4.2 Purpose-Aware Use Case Diagrams

In the first level, we selected to exploit Use Case Diagrams to demonstrate a high-level overview of the purpose-aware requirements (Gemino and Parker, 2009). Use case diagrams are Behavioral Unified Modeling Language (UML) diagrams showing how users and other external entities interact with a system in a simple way. They do not include a high level of detail regarding these interactions or demonstrate interactions between the system's internal entities. They cannot replace the detailed textual description; they can, however, complement it visually.

They consist of the following elements (Figure 1 / left side): (a) *Use Cases*, represented by an oval shape; (b) *Actors*, i.e., users and external entities, represented by figures; (c) *Associations*, lines between the Actors and the Use Cases; (d) *System Boundary Box*, that sets the system scope.

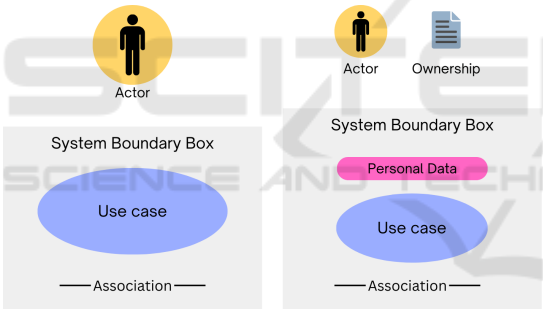

Figure 1: Use Case Diagrams (basic, extended).

We extend Use Case Diagrams with two more elements: (e) "Personal Data" to be shown in the associations between Actors and the System, (f) "Ownership of Personal data" to be shown next to the actor they belong to, as shown in Figure 1 / right side.

We then present a methodology for defining purpose-aware use case diagrams during the requirements capture stage as follows:

1. Define the "Personal Data" given by the user actors and other actors towards the system.
2. Define the "Personal Data" sent from the system to other actors (external systems or users).
3. Define the "Personal Data" used within use cases and on associations.
4. Define the "Ownership" for each "Personal Data".

Example: a Simple Task Management Application - Use Case Diagram. We demonstrate our methodology by creating the use case diagram for the running example, before and after the extension, presented in Figures 2 and 3, respectively.

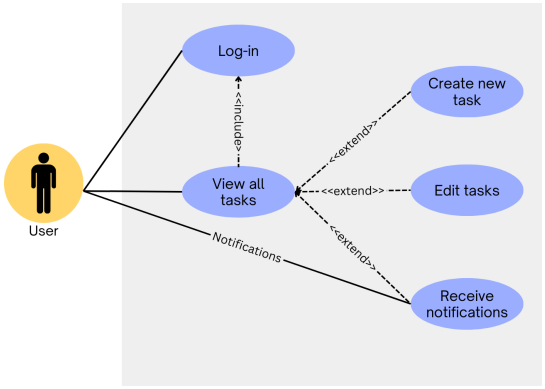

Figure 2: Example Basic Use Case Diagram.

We consider only User to be an external actor and we recognize five use cases: Log-in, View all tasks, Create new task, Edit tasks, and Receive notifications.

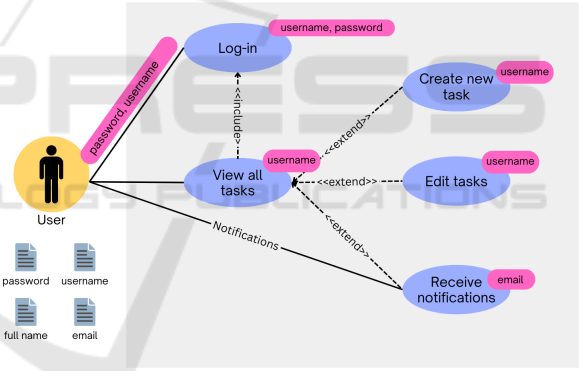

Figure 3: Example Extended Use Case Diagram.

The personal data given by the User towards the system are: full name, username, email address, and password. No personal data is sent from the system to other external entities. The association between the User and the *Log-in* use case carries on personal data (username and password). The *Log-in* use case processes the user's password and username, the *View all Tasks, Create New Task, Edit Task* use cases process the username, while *Receive Notifications* processes the user email address. If the registration operation were also included in our example, we would notice the User sending their full name and email address towards the system. However, for this example, we consider that this data already exists in the DB.

4.3 Purpose-Aware Sequence Diagrams

To accommodate our methodology and definition of purpose, we needed to provide a second level of detail and refinement. To do so, we exploited UML Sequence Diagrams (Micskei and Waeselynck, 2011). Sequence diagrams are Interaction Behavioral UML diagrams, capturing the interaction in a collaboration to realise a use case or an operation. Interactions can be between the user and the system, between subsystems (entities of the system), or between the system and other external entities. They detail how functionalities should be carried out and are also used in requirements engineering. Such diagrams demonstrate the order (in time) of the interactions between system entities and the messages exchanged.

They consist of the following entities (Figure 4 / left side): (a) *Objects*, representing participants involved in the interactions; (b) *Lifeline*, showing the passage of time for a specific participant (order, not duration); (c) *Activation box*, representing the period the participant is active in an interaction; (d) *Actors*, representing external entities interacting with the system, including the user; (e) *Messages*, representing the communications between two participants.

We extend sequence diagrams to highlight the processing of personal data, with two elements: (f) "Personal Data" sent along with messages, with annotations as identifiers, i.e., to demonstrate that it is the same piece of data moving on to the sequence of interactions, e.g. Phone$^{\alpha}$; (g) "Personal Data Ownership", in a similar manner as in extended use case diagrams. We present the extended sequence diagrams in Figure 4 / right side.

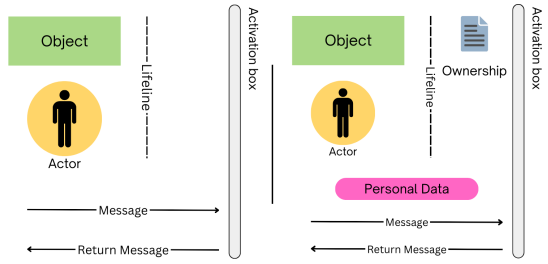

Figure 4: Sequence Diagrams (basic, Extension).

We then present a methodology for defining purpose-aware sequence diagrams as follows:

1. Define the "Personal Data" to be exchanged between objects and actors to fulfil an operation.
2. Define the exact messages on which personal data are sent.
3. Define the "Ownership" for each "Personal Data".

Example: a Simple Task Management Application - Sequence Diagram. We demonstrate our methodology by creating the sequence diagram for the running example before and after the extension, presented in Figures 5 and 6, respectively. The use case and sequence diagrams form the result of *Step 3* of our methodology.

We only present the login functionality diagram for the sake of brevity, which operates as follows: the user requests to login, sending their username and password towards the system authorisation entity, which then in turn sends these personal data to the database of the system to check if log-in can be authorised. The database informs the authorisation entity that the user is either authorised or denied, returning only the userame, so that the authorisation entity can identify the user for which this message goes. Once the user is successfully logged-in the authorisation entity informs the tasks management entity, asking it to display all the tasks of the specific user while accompanying the request with the user username.

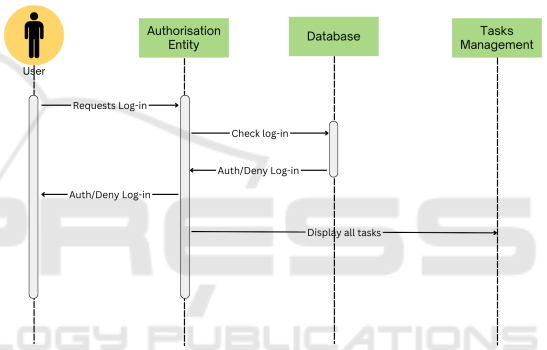

Figure 5: Example Basic Sequence Diagrams.

To model the above-described operation, we have the User as an external actor and the *Authorisation Entity*, *Database*, and *Tasks Management Entity* as objects. The personal data to be exchanged between the object and actors in the specific operation are: username, password. Both will be sent from the User towards the Authorisation entity on the *Request Log-in* message, and from the Authorisation entity towards the DB on the *Check Log-in* message, while the username will be send from the DB to the Authorisation entity on the Auth/Deny Log-in message, and from the Authorisation entity to the *Tasks Management entity* on the *Display all tasks* message. All mentioned personal data are owned by the User.

With the above, we define the global purpose, i.e., prescribing the interactions of a number of system entities, instead of individually. However, a number of different sequence diagrams might be needed to compile the total interactions of the complete system. The prescribed sequence of actions is the only one allowed regarding the personal data involved.

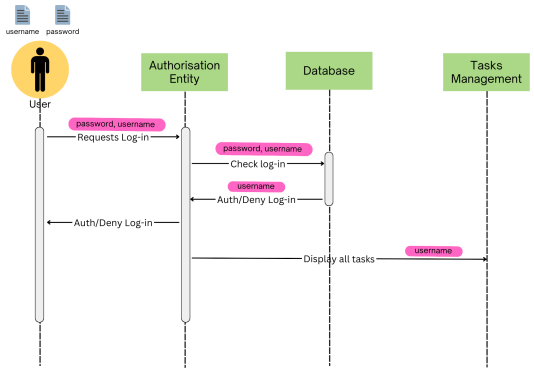

Figure 6: Example Extended Sequence Diagrams.

5 INTEGRATING AnálisisP WITH DiálogoP

The proposed purpose-aware sequence diagrams have a direct one-to-one mapping to our DiálogoP formal purpose language.

On the top level, we have purpose, which is the same case for AnálisisP. Then, in DiálogoP each purpose consists of many *sessions* that correspond directly with sequence diagrams in AnálisisP. Subsequently, each session of DialogoP includes *communicating entities*, corresponding to actors and objects in AnálisisP. The entities in DialogoP exchange *messages* and can be distinguished in *sending entities* and *receiving entities*. In AnálisisP, we do not explicitly distinguish the actors and objects as sending or receiving; we do, however, indicate the arrow direction, therefore implying sending and receiving parties. Messages can carry text or numerical values, but both methodologies also have a special type of *personal data*. Moreover, *personal data stores* define the personal data ownership. A summary of these relationships can be seen in Table 2.

Table 2: Mapping Between AnálisisP and DiálogoP.

	DiálogoP	AnálisisP
1	Purpose	Purpose
2	Session	Sequence Diagram
3	Communicating Entity	Object, Actor
3a	Sending Entity	Object, Actor if sending
3b	Receiving Entity	Object, Actor if receiving
4	Message	Message, Return Message
5	Personal Data Type	Personal Data
6	Personal Data Stores	Ownership

Integration. In AnálisisP, the two types of diagrams are created. Then, sequence diagrams are fed from AnálisisP to DiálogoP, where they are first converted into visual purposes and automatically transformed into formal language purposes. Additionally, the figure illustrates a future addition, ModéloP, aimed to receive the formal purpose and transform it into a formal process calculus model validated for purpose compliance. All three tools, together, comprise the ADMP Framework.

6 PRIVACY QUERIES

The defined purpose-aware use case (UCD) and sequence diagrams (SD) can further assist in responding to a set of additional important privacy queries, as follows:

Q1. Does the system send personal data towards any external entities, and if so, to which entities? (UCD)

Q2. Which personal data does the system collect from users? (UCD)

Q3. Which system entities are processing a specific piece of personal data? (SD)

Q4. Which pieces of personal data are processed by a specific system entity? (SD)

Q5. Are the personal data collected from the users indeed processed by the system entities, i.e. are they used? Correlation with Data Minimisation principle. (combination from both diagrams)

Example: a Simple Task Management Application - Privacy Queries. We return to the previous running example to collect the responses to the above queries.

Q1: we observe that the system does not send any personal data towards external entities.

Q2: we observe that the system collects the password, username, full name, and email of the User.

Q3: we observe that (a) the username is being processed by the Authorisation entity, the DB, and the Tasks entity; (b) the password is processed by the Authorisation entity, and the DB; (c) the email is used by the Notifications entity.

Q4: we observe that (a) the Authorisation entity processes the username and password; (b) the Tasks entity processes the username; (c) the Notifications entity processes the email.

Q5: we observe that all data collected from the User are indeed processed by some system entities, except the user's full name. This raises a flag for the system's privacy, as the system collects personal data that is not somehow used in the offered operations.

7 CONCLUSIONS

This work presents the integration of GDPR processing purposes into system requirements, resulting in purpose-aware system requirements. A visualisation

through extended use case and sequence diagrams is proposed. In our previous work of (Vanezi et al., 2020), we presented DiálogoP, a formal language and tool that allows the transformation of visual purpose-aware requirements into a formal type language purpose specification, which has the potential to be rigorously checked and validate the compliant behaviour of a system model. As an immediate next step, we plan on presenting ModéloP, an algorithm for transforming the formal type purpose into a pi-calculus formal model that is guaranteed to comply with its purpose-aware requirements. This model then has the potential to be used in model-driven engineering to produce a system's code. We also aim, as a future work, to conduct an extensive evaluation of our methodology with software engineers, and a validation through a real-life case study.

REFERENCES

Alhazmi, A. and Arachchilage, N. A. G. (2021). I'm all ears! listening to software developers on putting GDPR principles into software development practice. *Personal and Ubiquitous Computing*, 25(5):879–892.

Alshareef, H., Stucki, S., and Schneider, G. (2021a). Refining privacy-aware data flow diagrams. In *Proceedings of SEFM 2021*, pages 121–140. Springer.

Alshareef, H., Stucki, S., and Schneider, G. (2021b). Transforming data flow diagrams for privacy compliance. *MODELSWARD*, 21:207–215.

Alshareef, H., Tuma, K., Stucki, S., Schneider, G., and Scandariato, R. (2022). Precise analysis of purpose limitation in data flow diagrams. In *Proceedings of ARES 2022*. ACM.

Antignac, T., Scandariato, R., and Schneider, G. (2016). A privacy-aware conceptual model for handling personal data. In *Proceedings of ISoLA (1) 2016*, pages 942–957. Springer.

Antignac, T., Scandariato, R., and Schneider, G. (2018). Privacy compliance via model transformations. In *Proceedings of EuroS&P Workshops 2018*, pages 120–126. IEEE.

Barth, A., Datta, A., Mitchell, J. C., and Nissenbaum, H. (2006). Privacy and contextual integrity: Framework and applications. In *Proceedings of S&P'06*, pages 184–198.

Basin, D., Debois, S., and Hildebrandt, T. (2018). On purpose and by necessity: Compliance under the GDPR. In *Proceedings of FC'18*, pages 20–37. Springer.

Booch, G., Jacobson, I., Rumbaugh, J., et al. (1996). The unified modeling language. *Unix Review*, 14(13):5.

Byun, J., Bertino, E., and Li, N. (2005). Purpose based access control of complex data for privacy protection. In *Proceedings of SACMAT'05*, pages 102–110. ACM.

De Masellis, R., Ghidini, C., and Ranise, S. (2015). A declarative framework for specifying and enforcing purpose-aware policies. In *Proceedings of STM'15*, LNCS 9331, pages 55–71. Springer.

European Parliament and Council of the European Union (2012). Charter of fundamental rights of the european union. *Official Journal of the European Union*.

European Parliament and Council of the European Union (2015). General data protection regulation. Official Journal of the European Union.

Gemino, A. and Parker, D. (2009). Use case diagrams in support of use case modeling: Deriving understanding from the picture. *Journal of Database Management*, 20(1):1–24.

Goldman, E. (2020). An introduction to the California Consumer Privacy Act (CCPA). *Santa Clara Univ. Legal Studies Research Paper*.

Jafari, M., Fong, P. W., Safavi-Naini, R., Barker, K., and Sheppard, N. P. (2011). Towards defining semantic foundations for purpose-based privacy policies. In *Proceedings of CODASPY'11*, pages 213–224. ACM.

Kala, K. (2019). Refinement of the general data protection regulation (GDPR) model: administrative fines perspective. Master's thesis, University of Tartu.

Kouzapas, D., Dardha, O., Perera, R., and Gay, S. J. (2016). Typechecking protocols with Mungo and StMungo. In *Proceedings of PPDP'16*, pages 146–159. ACM.

Matulevičius, R., Tom, J., Kala, K., and Sing, E. (2020). A method for managing GDPR compliance in business processes. In *CAiSE Forum 2020*, pages 100–112. Springer.

Micskei, Z. and Waeselynck, H. (2011). The many meanings of UML 2 sequence diagrams: a survey. *Software & Systems Modeling*, 10(4):489–514.

Petković, M., Prandi, D., and Zannone, N. (2011). Purpose control: Did you process the data for the intended purpose? In *Workshop on Secure Data Management*, pages 145–168. Springer.

Riahi, S., Khosravi, R., and Ghassemi, F. (2017). Purpose-based policy enforcement in actor-based systems. In *Proceedings of FSEN'17*, LNCS 10522, pages 196–211. Springer.

Sing, E. (2018). A meta-model driven method for establishing business process compliance to GDPR. Master's thesis, University of Tartu.

Tom, J., Sing, E., and Matulevičius, R. (2018). Conceptual representation of the GDPR: model and application directions. In *Proceedings of BIR 2018*, pages 18–28. Springer.

Tschantz, M. C., Datta, A., and Wing, J. M. (2011). On the semantics of purpose requirements in privacy policies. *arXiv preprint arXiv:1102.4326*.

Tschantz, M. C., Datta, A., and Wing, J. M. (2012). Formalizing and enforcing purpose restrictions in privacy policies. In *Proceedings of SP'12*, pages 176–190. IEEE Computer Society.

Vanezi, E., Kapitsaki, G. M., Kouzapas, D., Philippou, A., and Papadopoulos, G. A. (2020). Diálogop-a language and a graphical tool for formally defining GDPR purposes. In *Proceedings of RCIS 2020*, pages 569–575. Springer.

Yang, N., Barringer, H., and Zhang, N. (2007). A purpose-based access control model. In *Proceedings of IAS'07*, pages 143–148. IEEE Computer Society.