# A Model-based approach to realize privacy and data protection by design

Gabriel Pedroza, Victor Muntes-Mulero, Yod Martin, Guillaume Mockly

# A Model-based Approach to Realize Privacy and Data Protection by Design

Gabriel Pedroza
*Universit Paris-Saclay*
*CEA, List,*
*F-91120, France*
*gabriel.pedroza@cea.fr*

Victor Muntés-Mulero
*Beawre Digital SL*
*Barcelona, Spain*
*victor.muntes@beawre.com*

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

Guillaume Mockly
*Trialog*
*Paris, France*
*guillaume.mockly@trialog.com*

***Abstract*—Telecommunications and data are pervasive in almost each aspect of our every-day life and new concerns progressively arise as a result of stakes related to privacy and data protection [1]. Indeed, systems development becomes data-centric leading to an ecosystem where a variety of players intervene (citizens, industry, regulators) and where the policies regarding data usage and utilization are far from consensual. The new General Data Protection Regulation (GDPR) enacted by the European Commission in 2018 has introduced new provisions including principles for lawfulness, fairness, transparency, etc. thus endorsing data subjects with new rights in regards to their personal data. In this context, a growing need for approaches that conceptualize and help engineers to integrate GDPR and privacy provisions at design time becomes paramount. This paper presents a comprehensive approach to support different phases of the design process with special attention to the integration of privacy and data protection principles. Among others, it is a generic model-based approach that can be specialized according to the specifics of different application domains.**

***Index Terms*—Privacy by design, GDPR, data protection, model-based, personal data detection, DFD, MDE, MBSE.**

## 1. Introduction

Big data applications follow a growing tendency nowadays and are central for many domains like Health Care, Banking, Energy and more recently Transportation [1]. The ecosystem around is composed by a variety of players pursuing goals and purposes that can conflict each other. Thus for instance, the need for information sharing and diffusion, *e.g.*, in social networks, is in potential conflict with the privacy and interests of individuals. Also, policy makers and authorities can negatively impact businesses and companies which strongly rely upon data gathering and exploitation. The development of emerging domains, like those relying upon AI techniques, are data greedy but have been proven powerful enough to impact society's pillars like ethics, safety, economy, and even democracy. Along with that, malicious agents can always threaten individuals' data thus challenging the fences to protect them. In this context, the European Commission enacted in 2018 the General Data Protection Regulation (GDPR) [2] which introduces new provisions to protect individuals' rights with respect to the stakes for data utilization. Since then, GDPR has stood for an ecosystem evolution by acting as a balance in the landscape of opposing forces. However, the referred evolution implies changes in the development cycle of data-centric systems and software. Since GDPR remains a legal text, an overall need arose for engineers to ensure compliance with the regulation. In fact, the different stakeholders (engineers, architects, developers) require support to interpret and integrate GDPR provisions into the development cycle, in particular, at design phase. Despite that "Privacy by Design" is a relatively known principle, to our knowledge, a lack still exists for approaches to comprehensively fill the gap between GDPR and typical frameworks understandable to engineers.

In this paper, we propose a comprehensive approach that aims to fill referred gap by realizing Privacy and Data Protection by Design (*PDPbD*) along three main axes. First, the approach relies upon a method that provides guidance for systems and software design. The method is model-based and covers essential phases including personal data detection, modeling and privacy assessments for data, processes and architecture involved. Secondly, we have developed modules that leverage existing technology so as to provide a tooled framework that supports the method. Third, the tandem method-plus-tool realizes the Privacy-by-Design principle by combining conceptualization and implementation in a understandable manner for engineers non-savvy in privacy and GDPR matters. Furthermore, since the approach is generic and agnostic of the underlying tooling, it is (1) amenable to prevail across technology obsolescence, (2) implementable with other tool choices and (3) amenable to be specialized according to the particularities of a given application domain.
The rest of the paper is structured as follows. Section 2 introduces the overall approach, named *PDPbD*, which is detailed in the following Sections: Section 3 is dedicated to the Personal Data Detection phase, Section 4 shows the Model-based Design phase, and Section 5 describes the Privacy Assessments phase. The approach is illustrated through a Case Study pertaining to the automotive domain in Section 6. It is then positioned w.r.t. selected works in Section 7. Finally, some conclusions and perspectives for our work appear in Section 8.

## 2. Privacy and Data Protection by Design

Nowadays, it is generally accepted that systems and software design can be oriented either by goals or by risks [3]. The design guided by goals typically targets the fulfilment of requirements which are elicited to achieve functional objectives/missions and also to meet constraints. The design guided by risks typically targets the elicitation

of cost-acceptable and technically-effective countermeasures which reduce impact of risks below acceptable levels. The method proposed in this work aims to harmonize both perspectives by defining a framework that considers and integrates the elements related to both requirements engineering and risks management. To achieve referred integration, the *PDPbD* method relies upon five phases and is based upon a modeling paradigm inspired in particular by Model Driven Engineering (MDE) [4]. Fig. 1 shows an overview of the *PDPbD* method.

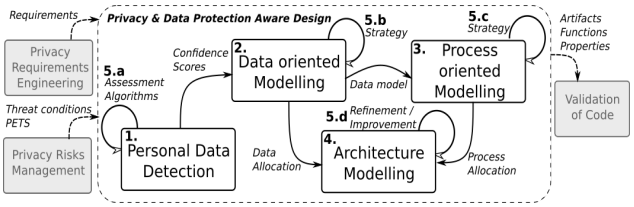


Figure 1. Overall *PDPbD* approach

The first phase is named Personal Data Detection and is meant to help in the identification of personal data. In this paper, we assume personal data are first detected with preexisting tools, and we propose an open data-based mechanism to guide system creators in the identification of linkability risks based on connecting system information to other external data sources representing data subjects. Once personal data are detected, the second phase aims to capture data structures at different levels of abstractions in a so named data-oriented model. This second phase reuses the estimations obtained in the first phase and is meant to assess the protection of data structures. The third phase is dedicated to capture the processes in which data and personal/sensitive data are involved. The so named process-oriented models are typed with the data and scores obtained in previous phases and helps to validate the privacy properties in data flows. In the fourth phase, a functional architecture is proposed to support both the data-oriented and process-oriented elements, *i.e.*, an architecture model. After refinement, the architecture model contains enough details to be taken as a basis prior to deployment. The last phase is an iterative assessment conducted at each of the previous phases (self-pointing arrows in Fig.1). To do so, once a model is obtained, it is allocated to the requirements/properties to be fulfilled. Some privacy-related strategies, techniques or algorithms (*e.g.* Hoepman's strategies [5]) are selected in order to meet requirements. The privacy-related strategy, technique or algorithm is then applied and if, after validation, the requirements are fulfilled, then the design models can be refined or used in the next phase.

## 2.1. Approach context and usage

The proposed approach includes the method previously introduced, and a tool-chain to support it. Both method and tool-chain are intended to be used in two design scenarios: (1) engineering of a brand new system or software, or (2) re-engineering of an existing one. For the sake of clarity, in this paper the approach usage is oriented to the 2nd. scenario, given that it is better suited to address the stakes raised by a system evolution, as is the case since GDPR enactment. Our approach follows a data-centric perspective and, as such, its initial phase focuses on data structures and flows within the system under study, *e.g.*, data warehouses. Thus, existing data structures (like legacy databases) and system specifications/documentation are assumed as method inputs. Once applied, the method aims to produce a new system design compliant with privacy and data protection provisions. The design workflow is detailed in the following Sections and includes both methodological and implementation parts.

# 3. Personal Data Detection

The main objective of the Personal Data Detection (PDD) phase is to identify personal data that are stored by data controllers (or data processors) in their data warehouses. While different methods [6], scores [7] and tools [8] exist to automatically detect personal data within databases, (*e.g.* as a set of scores measuring the probability of a particular data item to represent personal data), many attacks are still possible by linking tables to external data sources given their contents variability across the life cycle. Unfortunately, the literature shows a lack of approaches to support engineers during the identification of such risks. The proposed PDD phase -including method and tool- is indeed a novelty that can help system developers during personal data identification and estimation of related linkability risks. For the sake of brevity, in this paper we strengthen the focus on this latter capability, used as complement to personal data detection.

The PDD phase accepts data which are represented as a relational database and expressed as a set of tables, each of them being a set of attributes. For simplicity, we assume that for each table $T$ represented in the database, a score '$s$' is generated that represents its probability to contain personal data in at least one of its attributes. To estimate the associated linkability risks in order to update the score '$s$', the method consumes Open Linked Data from open data sources like Schema.org, DBPedia, and Wikidata, although the approach could be extended to other data sources. Then data instances are searched representing identifiable individuals which can be related to existing concepts in the database. The method explores ontologies related to persons and it then looks for people related to different classes derived from keywords in the system, using a knowledge graph. The graph connects database tables with concepts representing persons as shown in Fig. 2: blue nodes represent persons, yellow nodes represent entities extracted from the system database under analysis, and orange nodes represent classes, extracted from open data sources, connecting blue and yellow nodes. Two different types of edges are extracted from open data: (i) **isA** - links two entities where one is a subclass of the other or an instance; and (ii) **linkableTo** - links two entities that are semantically related although they represent two different types of concept. A third type of link named **FK** connects yellow nodes whenever a foreign key is used to link tables within the database representing the concepts in nodes.

**Implementation.** Following, we explain the mining algorithm to create the graph shown in Fig. 2 and to compute the Open Data Score to update '$s$'. First, the

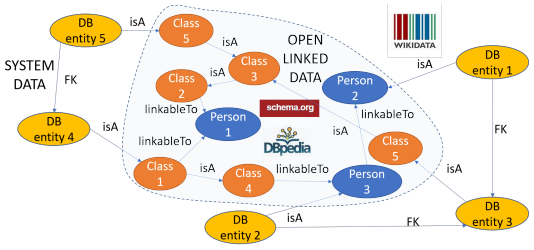


Figure 2. PDD graph linking database entities with data subjects

module extracts information related to classes representing persons from Schema.org and DBPedia to create the core of the graph (blue nodes). In particular, it creates person nodes from instances of "Person" class extracted from properties in Schema.org and subclasses in DBPedia. Then, given any table name in the database, related classes are searched using fuzzy matching provided by the Wikidata search engine, and choosing the most relevant result according to Wikidata. Based on this result, an automated exploratory traversal of information is executed by finding subclasses (property P279 in Wikidata), instances of (P31), and classes that are said to be the same (P460), adding them in the graph. The relationships found are labelled as "isA" in the graph. Domain specific properties are also explored as needed, *e.g.*, if the database is related to the automotive sector, the module explores vehicle-specific properties such as "Designed to carry" (P3349) or "Operator" (P137), to find human beings related to the vehicle. These properties establish relationships that are labelled in the graph as "linkableTo". The algorithm follows a hybrid approach: it stores relevant information in the graph database and only submits extra web queries when required. Once the module has gathered all the information, it then searches for persons related to a particular entity, finding the shortest paths between Tables (yellow nodes in Fig. 2) and classes, initially extracted from the open data sources (blue nodes). Given these shortest paths, further filters can be applied, *e.g.* to limit or minimize the number of "linkableTo" relationships in the path. Once the graph is completed, engineers can then manually validate the results by confirming the concepts selected from the open data sources, and validating the linkability paths, according to the system context. The score '$s$' of each table (originally computed without open data sources) can be updated depending on the path length. More specifically, the linkability of a database entity to personal data is calculated as an Open Data Score $f^d$, where $f$ is the propagation factor ($0 \leq f \leq 1$) and $d$ is the minimum number of hops between the two nodes in the graph. If $f^d > s$ then '$s$' is updated by $s := f^d$. The PDD assessment just described can be run periodically by the data engineer or designer in order to incorporate updates in both system databases and external open data sources.

## 4. Privacy-aware Modeling

Our approach pursues a main objective in this phase which is two fold: first, to represent data, flows and architecture in design models, inheriting outcomes from previous phase (PDD), and secondly, to leverage existing MDE techniques [4], [9] to provide a modeling framework suitable to conduct assessments for improving privacy and data protection, as explained in next Section 5.

### 4.1. Data-oriented model

We propose to rely on a so named data-oriented model to capture and represent the data structures under study in preparation for further analyses. The proposed data-oriented models can contain meta-data, in particular, the outcomes from the personal data detection phase. Following a MDE perspective, data-oriented models are meant to contain high-level representations of data instances still amenable to apply techniques as suggested in the existing data protection strategies: Minimize, Separate, Abstract, Hide [5], [10]. The strategies can be selected by the engineer in order to validate allocated requirements.

**Implementation.** The implementation of the data-oriented models is based upon a UML profile named *PDPbD* and implemented in Papyrus [9]. An excerpt of the profile is shown in Fig. 3 which is indeed an extension of the UML Class language [11].

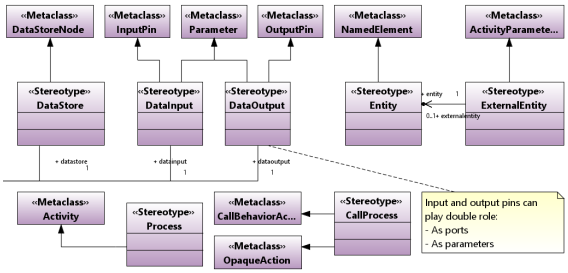


Figure 3. Excerpt of the profile for data and process-oriented models

The profile captures the fundamental notions needed to model data structures and deploy protection strategies. The semantics of the profile elements is described in line.

- `Data.` Represents a generic category of data which can be specialized. Three data elements are defined: *DataLink*, *CompositeData* and *Table*.
- `DataLink.` This element includes attributes to declare a path to an external data source or artefact and to declare its type. A *DataLink* is useful in particular to interface a model with external data sources, *e.g.*, code.
- `CompositeData.` This element plays the role of data container. It can be defined via a composite relationship which allows the aggregation of associated data. This element is useful to break down a data element into its parts.
- `Table.` This element helps to label tables within the model. Tables are typical means to structure data, *e.g.*, in databases.
- `isPersonalData.` The element is useful to label data and their parts when they are identified as personal. It includes an attribute named *likelihood* to be filled with the outcomes from the Personal Data Detection phase.
- `DataType.` This element allows the definition of user-customised data types.
- `OpaqueData.` It is used to represent data not directly interpreted by humans, *e.g.* ciphered data.

### 4.2. Process-oriented model

We propose a so named process-oriented model as a mean to capture data and data flows. The process-oriented models adopt the form of a directed graph where edges represent exchanged data and vertexes represent data sources or consumers, storage or processing elements. A well-known model corresponding to that pattern is the Data Flow Diagram (DFD). A DFD provides a high-level and synthetic view of a process centered on the data exchanged. From a MDE perspective, a DFD can be seen as a specialization of a modeling language like BPMN [12] or UML Activities [11]. To our knowledge, there is no standard nor commonly accepted definition for DFDs: the definitions found in the state of the art are usually based upon a variety of modeling rules. Our definition is essentially inspired from [13], [14] and includes a relevant feature found in [13] to support DFD refinements (not shown in this paper due to lack of space).

**Implementation.** The process-oriented model is also based upon the *PDPbD* profile and implemented in Papyrus [9]. Fig. 3 shows an excerpt of the profile. The profile is indeed an extension of the UML Activity language [11]. This language has been selected as a basis for extension for three reasons. First, it keeps a good level of abstraction which is adequate for future specializations of the *PDPbD* Framework. In addition, the UML Activities have been proven effective to be extended so as to define process-centric languages, like BPMN [12]. Last, flows and data in the DFD can be typed with the elements defined within the data-oriented model thus keeping models consistency. The profile elements are described inline.

- `Process`. The main container of a DFD model diagram representing a process. This element provides a canvas where all the other DFD elements can be modelled and detailed.
- `CallProcess`. This element encapsulates the notion of process. The CallProcess helps as a reference to link the diagram where the process is fully defined and detailed.
- `ExternalEntity`. The external entity is able to interact and call process functions or receive their outcomes. The ExternalEntity can only be located at the border of the Process canvas and thus also play the role of interface, *i.e.* as data source for the process or as data consumer.
- `DataStore`. A node representing a data storage (or data sink) within the process.
- `DataInput/DataOutput`. Unlike most typical DFD definitions, the *CallProcess* should include input/output pins for incoming/outgoing data. The pins can be assigned with a data type according to the conveyed flow.
- `DataFlow`. Directed edges that represent data flows between *CallProcesses* and *Processes*. *DataFlow* edges are connectors between *DataInput* and *DataOutput* pins.

### 4.3. Architecture model

The architecture model proposed herein is meant to support data flows in the DFD. To achieve it, a mapping is defined between elements defining the DFD over the architecture as follows. First, since the DFD introduced in Section 4.1 allows refinements, each *Process* can be decomposed and detailed as needed. Then, a candidate architecture needs to be proposed by the engineer. Last, the refined *Processes* are manually allocated among components. The referred allocation is part of the so called design space exploration problem [15]: the design space is generated by the possible distributions of *Processes* among architectural components (NP-hard problem).

**Implementation.** The architecture modeling language should preserve the consistency w.r.t. previous models. Given that the DFD already incorporates data defined within the data-oriented model, the definition of an architecture modeling language is reduced to transform the DFD profile into a compatible language. The selected candidate is indeed SysML BDD and IBD [16] given that it is standardized and has been successfully used to model system architectures. The DFD↦architecture mapping is implemented in Papyrus [9] and is defined as follows:

- *Process, DataStore* $\longrightarrow$ *Block (BDD)*
- *CallProcess* $\longrightarrow$ *Interal Block (IBD)*
- *InputData, OutputData* $\longrightarrow$ *Port (IBD)*
- *DataFlow* $\longrightarrow$ *Connector (IBD)*

The resulting architecture model is agnostic of deployment choices and consequently no separation between HW and SW is supported. *InputData* and *OutputData* are both mapped into respective Ports where the communicating interfaces can be defined. The definition of interfaces and their signatures relies upon *InputData* and *OutputData* types. Finally, the *DataFlow* corresponds to a single Connector that represents the channel conveying the data.

## 5. Privacy Assessment and Validation

An assessment is proposed in order to validate model properties so as to ensure privacy and data protection by design. The assessment is strongly inspired in the design strategies already proposed in different standards and regulations like ENISA [17] and ISO-27550 [10], and research papers [5]. Table 1 shows an excerpt of the design strategies which are deployed as built-in features of the *PDPbD* framework.

| Design strategy | | Description |
|---|---|---|
| Data oriented strategies | Minimize | Limit as much as possible the processing of PII |
| | Separate | Distribute or isolate personal data as much as possible to prevent correlation |
| | Abstract | Limit as much as possible the detail in which personal data is processed while still being useful |
| | Hide | Prevent PII from becoming public or known |
| Process oriented strategies | Inform | Inform PII principals about the processing of PII |
| | Control | Provide PII principals control over the processing of their PII |
| | Enforce | Commit to PII processing in a privacy friendly way and enforce this |
| | Demonstrate | Demonstrate that PII is processed in a privacy friendly way |

TABLE 1. DESIGN STRATEGIES FOR PDP ASSESSMENT

For the sake of brevity, we only describe four strategies via some techniques as examples selected to show

both conceptual definition and implementation in the tool-chain: in Section 5.1, *Abstract* and *Minimize*, and in Section 5.2, *Control* and *Inform*. Other privacy strategies can also by incorporated and selected by the engineer according to the analysis needs.

## 5.1. Data-oriented strategies

The data-oriented strategies are *Minimize*, *Separate*, *Abstract* and *Hide* [5]. A brief description of these strategies is given in Table 1. They are applied to data-oriented models to validate allocated privacy requirements.

**5.1.1. Abstract - K-anonymity.** An algorithm is implemented based upon the K-anominity definition found in [18]. K-anonymity relies on the notion of Quasi Identifiers (*QI*) which are column attributes of a table from which personal information can be inferred via correlation with other less significant information, *i.e.*, considered as non-personal: *If T is a table and QI is a set of quasi-identifiers associated with it, T is said to satisfy k-anonymity if each row-sequence of values of T restricted to the columns QI (T/QI) appears in at least k occurrences in T/QI*. The algorithm implemented allows to fix a *K-objective* and identify whether a given table satisfies it, given a set of quasi identifiers *QI* selected by the user.

**5.1.2. Minimize - Alpha-Anonymity.** A new definition and algorithm named $\alpha$-Anonymity are introduced. This new metric helps to measure mutual information between tables relying upon selected quasi-identifiers. More concretely, $\alpha$-Anonymity is based upon the notion of entropy $H(X)$ [19] which measures information of a random variable $X$. $H(X)$ allows to measure uniqueness of terms $w_i$ within a table $T$ ($H(T) = 0$) or indistinguishability ($H(T) = 1$). Given two tables $T_1$ and $T_2$ the mutual information [19] is given by:

$$I(T_1;T_2) := H(T_1,T_2) - H(T_1|T_2) - H(T_2|T_1) \quad (1)$$

where $H(T_1,T_2)$ is the joint entropy and $H(T_1|T_2)$ is the conditional entropy. In our definition, $\alpha$-Anonymity$(T_1,T_2,QI)$ of two tables $T_1$, $T_2$ given the quasi-identifiers $QI$ is given by:

$$I(T_1 \cup T_2; \alpha * QI) = H((T_1 \cup T_2) \cap \alpha * QI) \quad (2)$$

where $\alpha = \{\alpha_1 \dots \alpha_n\}$, $\sum_j \alpha_j = 1$, is used to weight the entropy of columns $C_j$ in $QI$ according to its relevance for disclosing personal data in $T_1$ and $T_2$ (*e.g.*, IDs, addresses):

$$H(\alpha * QI) = \sum_{t \in QI} \sum_{t \in C_j} \alpha_i P(X=t) log_b(P(X=t)) \quad (3)$$

The algorithm for $\alpha$-Anonymity gives an order of magnitude for the information shared between $T_1$, $T_2$ and $QI$ to be used for inferring personal identifiers.

## 5.2. Process-oriented strategies

The process-oriented strategies are *Inform*, *Control*, *Enforce* and *Demonstrate* [5]. A brief definition of these strategies is given in Table 1. These strategies can be applied to process-oriented models to validate the allocated privacy requirements.

**5.2.1. Control - Consent.** A provider-receiver pattern is defined to implement techniques within the strategy named Control, in particular, the technique for Consent, *i.e., the Data Subject shall freely provide explicit consent for his/her personal data when they are processed by a given Controller and Processor and for a specific purpose* (GDPR, Art.6, 1.a) [2]. To instantiate the pattern, the user selects a process-oriented model and a specific *Process* within the DFD model. The interface guides the user through several dialogue boxes where options need to be selected. The options are automatically gathered from the model and accordingly listed per category. The *Consent* pattern is instantiated via the following information found in the model:

- The *Controller* responsible for *PersonalData* processing,
- The *DataSubject* to request for *Consent*,
- The *Process* (or *CallProcess*) for which the *Consent* is demanded,
- The *Purpose* of the processing,
- The *PersonalData* involved in the processing.

**5.2.2. Inform - Notifications.** The provider-receiver pattern is also used to implement the technique Notifications within the strategy Inform. Notifications are prescribed by GDPR, Art. 34 [2], in particular, *When the personal data breach is likely to result in a high risk to the rights and freedoms of natural persons, the controller shall communicate the personal data breach to the data subject without undue delay.* To instantiate the Notification pattern, the tool guides the engineer through different dialogue boxes where options can be selected according to the underlying process-oriented model. The notification pattern requires the following information to be incorporated in the model:

- The details about the *PersonalDataBreach*, the *PersonalData* and *DataSubject* involved.
- The *Process* concerned by the *DataBreach*,
- The *Processor/Controller* managing the *Process*,
- The *Data Subject* to be notified,
- The notification can include details about *DataBreachImpact* and *MitigationMeasures*.

# 6. Automotive case study

We consider the case of a Cooperative Intelligent Transport Systems (C-ITS) based upon the Cooperative Awareness Message (CAM) basic service [20] which allows vehicles and the road infrastructure to exchange about their positions, dynamics and also dangerous events. The exchanges allow vehicles and drivers to safely react. The authentication of surrounding vehicles is conducted via digital signatures [21], and public key certificates which ensure message authenticity and integrity. The resulting architecture can be divided into two main parts: The Public Key Infrastructure (PKI), providing vehicle certificates (including revocation certificates list - RCL), and the C-ITS network, linking vehicles and road side and service units. In the following Sections, we apply our proposed method and tool-chain to this system in order to identify potential privacy issues and propose corrective privacy-by-design measures.

## 6.1. Identification of personal data

In this Section, we consider the application of the personal data detection method and module, discussed in Section 3, on a relational database representing the dataset present in the use case (see Fig. 4). In practice, the data should be spread across different participants: the tables *Frame* and *Vehicle* are typically stored within vehicles, while *Owner*, *Registration*, *Authorized Vehicle* are stored at PKI servers. *Certificate*, *Authority*, and *Key* are partially shared across vehicles and the PKI.

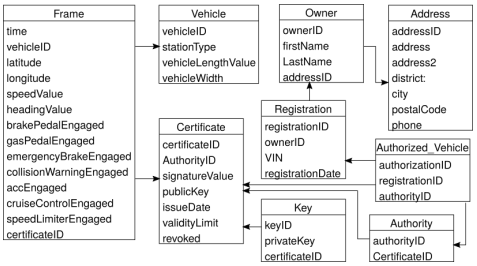


Figure 4. Schema used for personal data detection

Following the method, the first step is to link to specific concepts from the results of a search in Wikidata. For example the *Vehicle* table is indeed a vehicle and the *Address* is more specifically a postal address. The PDD module determines the paths connecting to persons through the knowledge graph using the **isA** and **linkableTo** edges: (i) postal address **isA** personal data, **isA** person property **linkableTo** person; (ii) vehicle **linkableTo** vehicle operator **isA** person. Table 2 compares the score computed by the PDD module for each table before an after the application of open data knowledge. While the *Address* table is identified as related to personal data by the database analysis, the *Vehicle* table is not. However, with the link between vehicle and driver established, the linkability is now considered to be high. Overall, the results in Table 2 show that, by considering open data knowledge, the possibility of having personal data conveyed in the network itself is better assessed, since otherwise, the database analysis would only consider the data handled by the PKI, *i.e.* an improved PDD accuracy.

| Location | Table | Score ($s$) | Open Data Score |
|---|---|---|---|
| PKI | *Owner* | High | High |
| | *Address* | High | High |
| | *Registration* | High | High |
| | *Authorized Vehicle* | Medium | High |
| Shared | *Certificate* | Medium | High |
| | *Authority* | Low | Medium |
| | *Key* | Low | Medium |
| Network | *Vehicle* | Low | High |
| | *Frame* | Low | Medium |

TABLE 2. PERSONAL DATA IDENTIFICATION SCORES

## 6.2. Privacy-aware C-ITS modeling

This Section is dedicated to show the application of the method and tool introduced in Section 4 for building a Privacy-aware design.

**6.2.1. Data-oriented C-ITS model.** A data-oriented model is made according to the schema shown Fig. 4. Since the PDD and the model-driven modules are interoperable, the schema can be directly imported as a data-oriented model including its scores ('$s$' and Open Data Scores). The schema elements are afterwards typed as *PersonalData* according to the scores obtained. A threshold can be settled by the engineer to stereotype personal and non-personal data, *e.g.*, $s := high \Rightarrow PersonalData$. Thus, he data typed as personal is inherited from Table 2. This step is crucial, since the DFD modeling and the privacy assessments strongly depend upon the data classification.

**6.2.2. Process-oriented C-ITS model.** The process-oriented model of the C-ITS includes three main parts: the Public Key Infrastructure (PKI), the Head Unit processing the CAM packets and the C-ITS network. Those parts are refined according to the representative elements involved within the CAM service. A scenario that illustrates the data flows of the system is as follows (see Fig. 5). The *C-ITS application* in the Communications Unit of the vehicle receives a CAM from a surrounding vehicle. The certificate is first validated and the public key used to decipher the payload. The information is then used to execute on-board actions, *e.g.* alert the driver, perform emergency braking. An overview of the respective DFD diagram and involved elements is shown in Fig. 5. Notice that the data ports and flows are typed with elements in the data-oriented model.

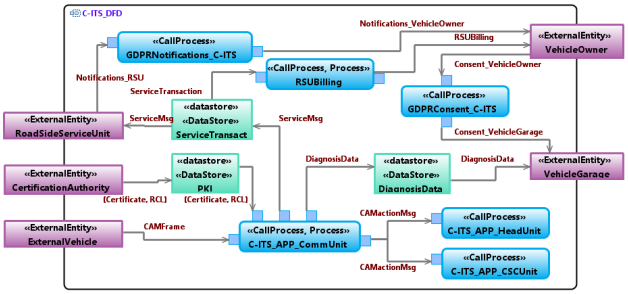


Figure 5. Process-oriented C-ITS model

## 6.3. Privacy assessment and validation

The privacy assessment of the case study is conducted on both data- and process-oriented models.

**6.3.1. Data-oriented privacy assessment.** We mainly focus on non-personal data within the C-ITS data-oriented model, since they can be used as quasi-identifiers (*QI*) and may provide sufficient information to unveil Owner's identity. Particular attention is paid to the *Frame* (see Fig. 4) since other than *vehicleID* and *certificateID*, the rest of attributes are considered non-personal. It is considered that column attributes storing unique data values can become risky unless *QI* is K-anonymous with $K \geq 2$ (minimal criteria for indistinguishability). Thus, candidate attributes to be K-anonymized are *QI*:={*latitude*, *longitude*, *time*}. After applying the K-anonymity algorithm in Section 5.1.1, with $K = 2$, the property is not satisfied. Thus, *Frame/QI* can be used to infer Owner's identity, in

particular, if the quasi-identifiers are used in an external table $T_2$ including, for instance, the vehicle's identifier. Such is the case in highways where the *vehicleID*, *time*, and location are stored by tolls for billing purposes what also requires credit card details [22], [23]. To limit this risk of identification, we consider a new schema in which the *Frame* is split into two tables $T_1 = Frame - QI$, $T_2 = Frame/QI$ and relating both with a common primary key *pk*. The $\alpha$-anonymity algorithm in Section 5.1.2 can be applied to evaluate this new configuration: $\alpha\text{-}Anonymity(T_1, T_2, pk) = 0.08$. This result shows an entropy close to 0 given that $pk$ shall be the only mutual information shared.

**6.3.2. Process-oriented privacy assessment.** The evaluation is conducted on the C-ITS process-oriented model. We particularly focus on the compliance of the design w.r.t. to GDPR provisions. As it is specified, the C-ITS does not include any processing element to cover both Consent and Notifications. These shortcomings are relevant since GDPR compliance is mandatory and information misuse and leaks can impact data subjects' rights. To improve the C-ITS DFD model, we first instantiate the Consent pattern explained in Section 5.2.1. As a result, a new process named *GDPRConsent_C-ITS* is introduced (see Fig. 5) linking the *Owner* to external entities like *CertificationAuthority* and the *Garage* since they respectively access and process *Certificates* (and other identifiers) and also *Frames*, *e.g.*, during vehicle diagnosis. Secondly, the Notification pattern explained in Section 5.2.2 is instantiated. As a result, a new process named *GDPRNotifications_C-ITS* is introduced (see Fig. 5) linking the *Vehicle* to external entities like *RSU* (Road Side and Service Units) and *Owner* meaning that the *RSU* informs the latter in case of personal data breach. The improved model is in compliance with GDPR provisions and is amenable to further refinement prior to implementation. Parts of the two introduced processes can indeed be distributed over the C-ITS architecture.

## 7. Positioning Regarding the State of the Art

The work herein described builds on existing practices from the state of the art of privacy engineering. It integrates them into general-purpose methods and tools already in use by systems and software engineers (viz. MBSE and Papyrus [9]), providing an operational method and tool-chain that align with mainstream engineering practice. Hence, its novelty w.r.t. the state of the art should not be assessed on the basis of its constituent parts, but rather on that of their technology readiness, their integration with one another, and on the privacy-by-design method support to the general engineering practice.

In this respect, Model-Based Systems Engineering (MBSE), as a methodological paradigm for systems and software engineering, may address all the different disciplines, activities, and concerns involved during the development life-cycle —and so can it be applied to privacy: Some remarkable works have focused on creating models of GDPR, *e.g.* [24]–[26] have all defined GDPR conceptual models supplemented with compliance rules; while others [27], [28] have defined meta-models for privacy derived from access control paradigms. Yet others ( [29], [30]) have addressed the introduction of privacy concerns into process models by defining extensions to modeling languages (BPMN [12] and Data Flow Diagrams [13]), and their transformations based on privacy patterns. Indeed, Data Flow Diagrams have themselves been repeatedly employed to model data processing activities in the context of security and privacy engineering [31], [32]. Although referred solutions are useful for their purposes, they all target individual facets of privacy models, whereas our approach addresses several viewpoints (data, process, architecture) in an integrated fashion. This is in fact aligned to the tenets of MBSE, which sustain the definition of complementary views to model different aspects of a system (*e.g.* functional, logical, behavioural, physical).

With respect to our assessment and validation features (5), they heavily draw from Hoepman's Privacy Design Strategies [5], later added to the ISO-27550 standard [10], and developed into privacy tactics [33]. Some of them have been refined into a structured language of privacy patterns [34], [35] elicited from a collaborative pattern catalogue [36]. Some of those patterns (k-Anonymity, $\alpha$-Anonymity, Consent, and Notifications) have been implemented in our tool-chain and illustrated in this paper. Among these, we include a traditional metric for privacy (K-Anonymity) and another one expressly defined herein named $\alpha$-anonymity. We have also defined a risk-oriented score for the personal data detected (3). Many other privacy metrics have been proposed ( [7], [37], [38]) and even if all serve their own purposes, they do not preclude further definitions: the metrics introduced in our work serve for their purpose and show distinctive particularities, *i.e.*, detecting linkability risks and reidentifiability risks associated to personal data. From a practical perspective, our tool-chain can also be compared with off-the-shelf tools, in particular, considering the classification from the IAPP [8]. A variety of tool categories is found: Data discovery; to help organizations identifying personal data in processes, Data mapping; to determine data flows, and Assessment management; to automate privacy program functions. Despite the huge number of vendors per category, they mostly target legal or IT departments and, to our knowledge, none of them addresses engineers needs in their daily practice.

## 8. Conclusions

In this paper, we have presented *PDPbD*, a model-based approach encompassing a method and a tool-chain to support engineers in integrating GDPR provisions during design-time activities (viz. detection of personal data, modeling, and conformity assessment). Regarding personal data detection, *PDPbD* introduces distinctive features that take into account the risk of re-identification from linkability to other sources. This fundamental phase determines the accuracy of the rest of the process. The modeling phase relies upon data, process and architecture models which build upon the outcomes of detection, and provide a basis to structure a privacy- and data-protection-aware design. The assessment phase leverages and implements strategies, techniques and algorithms found in the state of the art to confront design models w.r.t. GDPR obligations. The applicability of the approach was illustrated through an automotive case study. Overall, our approach

suitably combines conceptualization and operationalization, thus supporting engineers who are non-savvy in privacy and GDPR matters. It strives for leveraging existing wisdom and know-how on privacy and data protection whereas still remaining generic and agnostic of technology choices. Notwithstanding, several potential drawbacks can still be addressed. From a construct validity perspective, we heavily feature model-based concepts and practices, which might dispel those developer communities who are not familiar or reluctant to those approaches. Likewise, the interplay with risks and requirements remains open. Plus, improvements can be made regarding *e.g.* metrics comparisons, effectiveness of strategies, or toolchain optimization. All in all, we believe that the *PDPbD* approach already provides a consolidated basis to address and incorporate previous aspects as prospective work.

## Acknowledgments

This work was conducted in the scope of the project PDP4E. This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 787034.

## References

[1] R. C. King *et al.*, "Application of data fusion techniques and technologies for wearable health monitoring," *Medical Engineering & Physics*, vol. 42, pp. 1 – 12, 2017.

[2] C. of European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council," 2016, http://data.europa.eu/eli/reg/2016/679/oj.

[3] A. Cailliau and A. van Lamsweerde, "A probabilistic framework for goal-oriented risk analysis," in *2012 20th IEEE International Requirements Engineering Conference*, 2012, pp. 201–210.

[4] The Object Management Group, 2021, in https://www.omg.org/.

[5] J.-H. Hoepman, "Privacy Design Strategies," in *ICT Systems Security and Privacy Protection*, N. Cuppens-Boulahia *et al.*, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 446–459.

[6] C. Mouza *et al.*, "Towards an automatic detection of sensitive information in a database," in *2010 2nd International Conference on Advances in Databases, Knowledge, and Data Applications*. IEEE, 2010, pp. 247–252.

[7] I. Wagner and D. Eckhoff, "Technical privacy metrics: A systematic survey," Apr 2018.

[8] IAPP, "2021 Privacy Tech Vendor Report," International Association of Privacy Professionals, Tech. Rep., 2021. [Online]. Available: https://iapp.org/media/pdf/resource_center/2021TechVendorReport.pdf

[9] CEA, "Eclipse Papyrus Modeling Environment," 2021, https://www.eclipse.org/papyrus/.

[10] ISO, "Information technology – Security techniques – Privacy engineering for system life cycle processes," ISO, Tech. Rep. ISO/IEC TR 27550:2019, 2019.

[11] The Object Management Group, "UML Specification 2.5.1," 2021, in https://www.omg.org/spec/UML/About-UML/.

[12] ——, "BPMN Specification 2.0," 2021, in https://www.omg.org/spec/BPMN/2.0/About-BPMN/.

[13] E. Yourdon, *Modern Structured Analysis*, ser. Yourdon Press computing series. Yourdon Press, 1989. [Online]. Available: https://books.google.es/books?id=prMeB-4zdpgC

[14] K. Wuyts and W. Joosen, "LINDDUN privacy threat modeling: a tutorial," Distrinet, KU-Leuven, Tech. Rep., 2021, in https://linddun.org/downloads/LINDDUN_tutorial.pdf.

[15] S. Pandey *et al.*, "Architecture level design space exploration and mapping of hardware," in *International Symposium on Signals, Circuits and Systems. ISSCS 2005.*, vol. 2, 2005, pp. 553–556.

[16] The Object Management Group, "SysML Specification 1.6," 2021, in https://www.omg.org/spec/SysML/About-SysML/.

[17] ENISA, "Privacy and Data Protection by Design - from policy to engineering," EU ENISA, Tech. Rep., 2014, in https://www.enisa.europa.eu/publications/privacy-and-data-protection-by-design.

[18] P. Samarati, "Protecting respondents identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 6, pp. 1010–1027, 2001.

[19] L. Paninski, "Estimation of Entropy and Mutual Information," *Neural Computation*, vol. 15, no. 6, pp. 1191–1253, 2003.

[20] "ETSI EN 302 637-2 V1.3.1-Intelligent Transport Systems (ITS)," 2014. [Online]. Available: https://www.etsi.org/deliver/etsi_en/302600_302699/30263702/01.03.01_30/en_30263702v010301v.pdf

[21] "ETSI TR 103 197 V1.3.1-Intelligent Transport Systems (ITS)," 2017. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/103000_103099/103097/01.03.01_60/ts_103097v010301p.pdf

[22] K. Ogden, "Privacy issues in electronic toll collection," *Transportation Research Part C: Emerging Technologies*, vol. 9, no. 2, pp. 123–134, 2001.

[23] J.-J. Hwang *et al.*, "Securing on-line credit card payments without disclosing privacy information," *Computer Standards & Interfaces*, vol. 25, no. 2, pp. 119–129, 2003.

[24] J. Tom *et al.*, "Conceptual representation of the GDPR: Model and application directions," in *Lecture Notes in Business Information Processing*, vol. 330. Springer Verlag, Sep 2018, pp. 18–28.

[25] D. Torre *et al.*, "Model Driven Engineering for Data Protection and Privacy: Application and Experience with GDPR," *arXiv*, Jul 2020.

[26] M. Palmirani *et al.*, "Legal ontology for modelling GDPR concepts and norms," in *Frontiers in Artificial Intelligence and Applications*, vol. 313. IOS Press, 2018, pp. 91–100.

[27] P. Colombo and E. Ferrari, "Towards a modeling and analysis framework for privacy-aware systems," in *Proceedings - 2012 ASE/IEEE International Conference on Privacy, Security, Risk* . IEEE, 2012, pp. 81–90.

[28] J. E. Mokhtari *et al.*, "PrivUML: A privacy metamodel," in *Procedia Computer Science*, vol. 151. Elsevier, jan 2019, pp. 53–60.

[29] P. Pullonen *et al.*, "Privacy-enhanced BPMN: enabling data privacy analysis in business processes models," *Software and Systems Modeling*, vol. 18, no. 6, pp. 3235–3264, Dec 2019.

[30] T. Antignac *et al.*, "Privacy Compliance Via Model Transformations," in *3rd IEEE European Symposium on Security and Privacy Workshops, EURO S and PW 2018*. IEEE, Jul 2018, pp. 120–126.

[31] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, "A privacy threat analysis framework: Supporting the elicitation and fulfillment of privacy requirements," *Requirements Engineering*, vol. 16, no. 1, pp. 3–32, 2011.

[32] D. I. Oliver, *Privacy Engineering: A Dataflow and Ontological Approach*, 1st ed. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 2014.

[33] M. Colesky *et al.*, "A Critical Analysis of Privacy Design Strategies," in *Proceedings - 2016 IEEE Symposium on Security and Privacy Workshops, SPW 2016*. IEEE, aug 2016, pp. 33–40.

[34] ——, "A system of privacy patterns for user control," in *Proceedings of the ACM Symposium on Applied Computing*. New York, NY, USA: ACM, Apr 2018, pp. 1150–1156.

[35] M. Colesky and J. C. Caiza, "A system of privacy patterns for informing users: Creating a pattern system," in *ACM International Conference Proceeding Series*. New York, NY, USA: ACM, Jul 2018, pp. 1–11.

[36] "Privacy Patterns." [Online]. Available: https://privacypatterns.org/

[37] M. Bezzi, "An information theoretic approach for privacy metrics," *Transactions on Data Privacy*, vol. 3, pp. 199–215, 2010.

[38] D. J. Kelly *et al.*, "A Survey of State-of-the-Art in Anonymity Metrics," in *Proceedings of the 1st ACM workshop on Network data anonymization - NDA '08*. New York, New York, USA: ACM Press, 2008.