PDP4E

Methods and tools for GDPR Compliance through

Privacy and Data Protection 4 Engineering

Specification and design of model-driven design tool for privacy and data protection

Project: PDP4E
Project Number: 787034
Deliverable: D5.1
Title: Specification and design of model-driven design tool for privacy and data protection
Version: v1.0
Date: 26/07/2019
Confidentiality: Public
Author(s): Gabriel Pedroza (CEA),
Patrick Tessier (CEA),
Julien Signoles (CEA),
Thibaud Antignac (CEA),
David Sánchez (Trialog),
Elena González (Beawre),
Jacek Dominiak (Beawre),
Victor Muntés-Mulero (Beawre)

Funded by

Table of Contents

Document History

Version	Status	Date
V0.1	Initial Table of Contents	24/04/2019
V0.2	Re-structuring the document: new Table of Contents	29/06/2019
V0.3	Integration of partners' contributions (BeAwre, CEA-LSL).	02/07/2019
V0.4	Elicitation of the PDPbD framework requirements	02/07/2019
V0.5	Specification of the modules for data, process and architecture models.	03/07/2019
V0.6	Introduction, executive summary and summary added.	04/07/2019
V0.7	References included.	04/07/2019
V0.8	Requirements modelling section included. Use cases section included.	11/07/2019
V0.9	Addressing Tecnalia remarks.	18/07/2019
V1.0	Addressing UDE remarks.	24/07/2019

Approval		
	Name	**Date**
Prepared	Gabriel Pedroza (CEA)	24/04/2019
Reviewed	Jabier Martínez (Tecnalia)	15/07/2019
Reviewed	Nicolas E. Diaz Ferreyra (UDE)	22/07/2019
Authorised	Antonio Kung (Trialog)	31/07/2019

Circulation	
Recipient	**Date of submission**
Project partners	26/07/2019
European Commission	31/07/2019

List of Figures

List of Tables

Abbreviations and Definitions

Abbreviation	Definition
AID	Available Information Diagram
AST	Abstract Syntax Tree
BPMN	Business Processing Model and Notation
DFD	Data Flow Diagrams
DPIA	Data Protection Impact Assessment
DSIFD	Detailed Stakeholder Information Flow Diagram
DSL	Domain Specific Language
GDPR	General Data Protection Regulation
ICT	Information and Communication Technologies
IoT	Internet of Things
LGPL	Lesser General Public License
LINDDUN	Linkability, Identifiability, Non-repudiation, Dectectability, information Disclosure, content Unawareness, and policy and consent Non-compliance
MDE	Model Driven Engineering
OEM	Original Equipment Manufacturers
PDP	Privacy and Data Protection
PDPbD	Privacy and Data Protection by Design
PDP4E	Privacy and Data Protection 4 Engineering
PDP4E-Req	Tool resulted from WP4 to management GDPR and privacy requirements
PET	Privacy-enhancing Technologies
ProPAn	Problem-based Privacy Analysis
PID	Personal Information Diagram
PSCS	Precise Semantics of UML Composite Structures
ReqIF	Requirements Interchange Format

SDLC	Systems and Software Development Life Cycle
SIPOC	Suppliers, Inputs, Process, Outputs, Customers
SQL	Structured Query Language
SysML	Systems Modeling Language
TFEU	Treaty on the Functioning of the European Union
UML	Unified Modelling Language
UML4PF	UML 4 Problem Frames
UDEPF	University Duisburg-Essen Problem Frames
V&V	Validation and Verification
WP	Work Package
WP29	Article 29 Data Protection Working Party

Executive Summary

This document is the first specification of the PDP4E framework for Privacy and Data Protection by Design (PDPbD). The architecture is named “PDPbD framework” and is composed by several modules targeting several design goals. Overall, the framework aims to integrate the legal obligations introduced by the EU General Data Protection Regulation (GDPR) into systems and software projects during the design phase. To do so, the designer needs are covered via several modules of the architecture also described in the document. The design flow covers critical phases like the identification of personal data and their linkability, the representation of processes and architectures conveying data at high level, and the validation of privacy-related properties via different strategies and techniques including validation at code level. Indeed, when needed, the architecture should allow model refinements amenable to verify properties on external, implemented artefacts like code. The overall architecture embraces a model driven design approach specially for the data, process and architecture modelling. The architecture also supports the analysis of privacy features on structured/unstructured sources and their outcomes shall be used to generate or enrich MDE models. Referred models are the basis to validate fulfilment of privacy-related properties: whereas some of the properties can be validated at structural level (e.g., in the architecture model), it is foreseen that some of them shall depend on the implementation. Thus, the support for code validation provided by the PDPbD framework will improve the certainty on the property fulfilment and provide evidence of the requirements fulfilment. The initial choices taken to implement the PDPbD architecture mainly pursue three goals (1) leverage existing and mature MDE techniques to ease privacy-aware design, (2) support the method for PDPbD specified in the report D5.4 [6], and (3) keep the architecture flexible enough to interoperate with other PDP4E tools and methods, in particular, the frameworks for risk analysis (WP3), requirements engineering (WP4) and assurance process (WP6). Referred flexibility also means that models can be used in both prescriptive mode (e.g., after application of enhancement techniques) and descriptive mode (e.g., to refine or provide more detailed views). The MDE approach is meant to ease the achievement of referred goals.

Overall, the core contributions of this deliverable are:

- A set of functional requirements the overall architecture should satisfy to facilitate design phases oriented to achieve privacy and data protection
- A first draft of the PDPbD architecture that aims to provide support to non-savvy privacy engineers in order to achieve compliance with privacy regulations and in particular with GDPR

1 Introduction

1.1 Objective of the document

The objective of this document is to specify in a comprehensive manner the tool architecture proposed in PDP4E to achieve Privacy and Data Protection by Design (PDPbD). The specification aims to align a set of global functional requirements with the modules of the architecture covering the requirements.

1.2 Structure of the document

To achieve its goal, the specification is structured as follows. First, the section 2 introduces two platforms as the Model Driven Engineering (MDE) basis to build the architecture. These platforms are the basis upon which Papyrus-based extensions are developed. In section 3, a summary of designer needs is presented. The designer needs intersect with the specific obligations the GDPR impose on system and software designs involving data. In section 4, a first list of global functional requirements is specified. The requirements state the support foreseen for non-savvy privacy engineers regarding design activities. The section 5 gives an overview of the stakeholders' use cases for the architecture. In section 6, the five modules integrating the PDPbD architecture are described. Each module is associated to the requirements covered. Finally, the section 7 gives a summary of the specification.

1.3 Relation with other deliverables

The PDPbD architecture is developed in the scope of WP5 and provides tool support to apply the methodology specified in the report D5.4 [6]. As can be seen in Figure 1, the tools developed in WP5 receive inputs from, and provide outputs to other work packages. Regarding WP3, the impacted assets and privacy countermeasures resulting from data protection risk management may require to be considered during the design phase, and conversely. The requirements elicited in WP4 need to be allocated to different design models and traceability links accordingly settled in order to manage their validation. The assurance process may also demand references to those links as an evidence of compliance. Consequently, an harmonization of the PDPbD tool with the methods and tools developed in those work packages is foreseen (D3.1 [1], D3.4 [2], D4.1 [3], D4.4 [4], D6.1 [7], D6.4 [8]). This specification seeks an alignment with the report D2.4 entitled Overall system requirements [9].

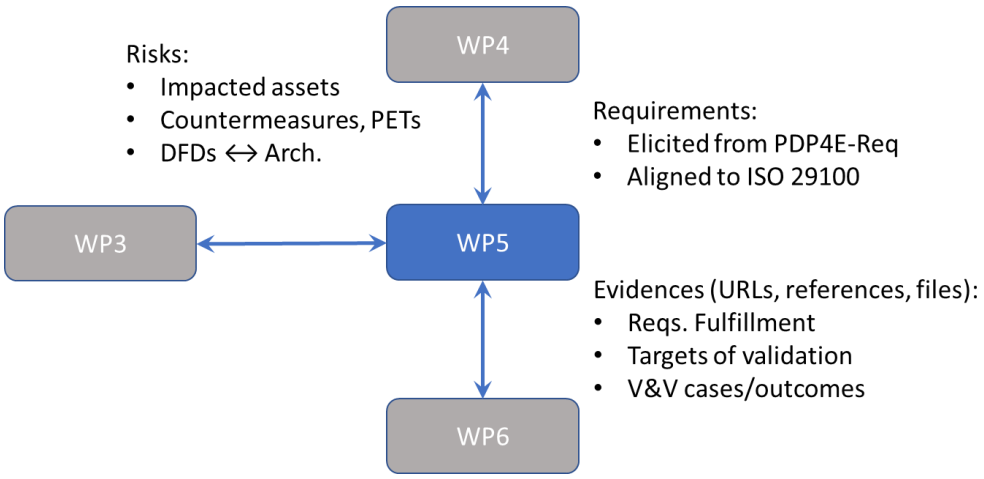

Figure 1. Relationships between the PDPbD framework defined in WP5 and other WPs

2 MDE background of the PDPbD framework

In this Section, we provide a brief description of two MDE tools that are used as a basis to build the PDPbD framework.

2.1 Eclipse

Eclipse[1] is a foundation created to benefit the providers of software development offerings and end-users by allowing a vendor neutral, open, and transparent community to be established around the Eclipse projects. More details can be found in the following report [24].

See the following explanation from Eclipse provides also a technical platform for development. See the following explanation from [25]:

"It defines the set of frameworks and common services that collectively make up infrastructure required to support the use of Eclipse as a component model, as a Rich Client Platform[2] *(RCP) and as a comprehensive tool integration platform. These services and frameworks include a standard workbench user interface model and portable native widget toolkit, a project model for managing resources, automatic resource delta management for incremental compilers and builders, language-independent debug infrastructure, and infrastructure for distributed multi-user versioned resource management."*

The Eclipse Platform is the technical basis upon which the PDPbD framework is developed. The main reason for this is that Papyrus is developed on top of Eclipse and we have selected it as a MDE background tool to be leveraged for the purposes of PDP4E. In addition, the Eclipse technology also offers features that will ease the implementation of interfaces to ensure PDP4E tools interoperability. In particular, the modularity and extensibility of Eclipse are to be exploited for that purpose.

2.2 Papyrus

Papyrus [15] is a modeling environment implementing the standard modeling languages UML [11], SysML [12], BPMN [10] and MARTE [13], among others. Papyrus has been built on top of the Eclipse Platform and implements several OMG standards[3]. The environment is distributed relying upon an Eclipse Public License (EPL). Among others, the implementation allows the definition of operational semantics for the languages. An operational semantics settle rules for the operation of elements within the model thus leading to represent dynamical features like order, passage of time, events and actions occurrence. For instance, the fUML standard [14] defines the semantics of the models based on class and activity diagrams, as well as its extensions defining the semantics of component models (Precise Semantics of UML Composite Structures). A forthcoming standard shall define the semantics of state machine diagrams. Thus, Papyrus allows the execution of models and more generally offers an environment allowing the automatic exploitation of the models (simulation, analysis and synthesis) for engineering purposes for the systems and software design.

[1] The Eclispe foundation. In https://www.eclipse.org/.
[2] https://wiki.eclipse.org/Rich_Client_Platform
[3] The Object Management Group. In https://www.omg.org/index.htm.

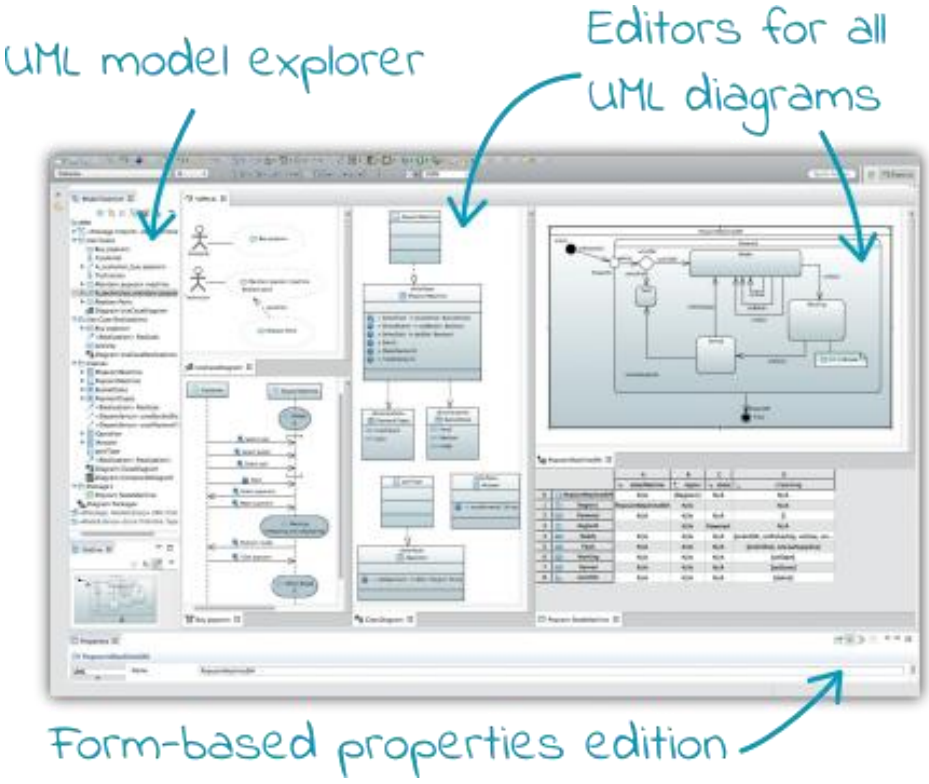

Figure 2: Papyrus framework overview

3 Design engineer needs

The main intended users of the PDPbD framework are data analysts and privacy design engineers. The framework shall support referred users during systems and software design projects, in particular, to ensure fulfilment of requirements related to privacy and data protection. The GDPR imposes new exigencies to any system or service involving data processing activities and in particular when personal or sensitive data are involved. To achieve compliance with GDPR, the requirements need to be characterized in terms of the *nature, scope, context, purpose and risks of the data processing activities* (Art. 24.1, Art. 35.1). Thus, design engineers and data analysts may require determining the requirements to be abided by a system or software development project. Once determined, the users of the PDPbD framework may also require support to:

1) Identify potential personal/sensitive data and explore the linkability between them from structured/unstructured sources
2) Identify/trace requirements to be fulfilled by a specific system design and its context
3) Select candidate design strategies for the requirements to be fulfilled
4) Select data protection techniques to carry out a strategy and fulfil a set of requirements
5) Conduct refinements leading to a detailed architecture specification
6) Validate privacy related properties on code involving data

The need for support on design-related tasks is influenced and oriented by the requirements to be fulfilled (e.g., elicited from risks or requirements management phases). Regarding GDPR, first analyses suggest that some of the following aspects shall be considered during the ascertainment process for finding out requirements:

- the categories of personal data i.e. how sensitive data is (Rec. 51, Art. 9),
- the size of the organization with regards to e.g. record keeping (Rec. 13, Art. 30),
- the application of corporate policies viz. binding corporate rules (Art. 47),
- the application of specific processing types such as profiling (Art. 22),
- the purposes for which the personal data was provided (Rec. 50, Art. 5.1.b, Art. 5.1.e),
- whether the data subject may have objected to some purposes e.g. direct marketing (Art. 21.2, Art. 21.3),
- special purposes such as research (Rec. 53, Rec. 156, Rec. 159, Rec. 162, Art. 89),
- the lawfulness and legitimacy of the bases for processing (Rec. 47, Art. 6), etc.

During the design phase, the requirements need to be instantiated regarding the specific scope and features of a system or software under design (SUD). Once instantiated, the needs 1)-6) listed above emerge and tool support is required, especially for non-savvy privacy engineers. The PDPbD architecture aims to provide such support. The proposal in this report shall evolve according to the evolution of users/stakeholders needs and related requirements as specified in D2.4, *Overall systems requirements* [9]. Since this is the first iteration in the work plan, the engineer design needs are expected to evolve according to the feedback obtained from implementation and consolidation steps.

4 Requirements for the PDPbD framework

A first set of requirements to be covered by the PDPbD framework is presented in subsection 4.2. Some salient aspects of the elicitation process are described in subsection 4.1.

4.1 Elicitation process

For elicitation of requirements, we first identified some specific user needs that arise when seeking conformity of a system design with respect to GDPR (see previous Section 3). The overall user-needs are the foundation of the whole PDP4E tool box which is developed taking into account the exigencies listed in *D2.4 Overall System Requirements* [9]. In addition to those references, the PDPbD architecture shall in particular consider and provide tool support for the method specified in *D5.4 Methods for data protection model-driven design* [6]. The method includes the phases to be applied when seeking privacy and data protection by design. Thus, the requirements capture the functional needs according to its phases. Last but not least, among the activities that have been conducted in the scope of WP5, the following activities have helped as a basis for the elicitation of requirements for the PDPbD framework:

a) Discussions about representative approaches to achieve privacy and data protection by design
b) Presentations to describe technical perspectives from different partners, background and foreground modules
c) Presentations and discussions about expected contributions from partners

Since this is a first version of the design architecture and it is composed by several modules, the requirements are specified at high level in a comprehensive manner.

4.2 PDPbD framework requirements

Table 1. Global requirements for the PDPbD framework

PDPbD-Req01	**IdentifyPersonalData**
	WP5
Description	The PDPbD framework shall provide for the Data Analyst Engineer the ability to estimate whether personal-sensitive data are present within structured and unstructured sources.
Assigned WP	WP5
Relation to other requirements	
Actor	Data Analyst Engineer
Priority	Could have
Type	Functional
Non-functional category	

Rationale	Non-savvy privacy engineers may require identifying personal data on existing data sources prior to any further design or redesign task.

PDPbD-Req02	**EstimateDataLinkability**
	WP5
Description	The PDPbD framework shall provide the Data Analyst Engineer the ability to estimate the potential links between personal-sensitive data and external data sources.
Assigned WP	WP5
Relation to other requirements	PDPbD-Req01; the estimation on linkability relies upon personal data identification.
Actor	Data Analyst Engineer
Priority	Could have
Type	Functional
Non-functional category	
Rationale	Once candidates for personal data are identified, it is relevant to estimate potential links with external data sources[4] to unveil potential privacy issues to be addressed during design or redesign.

PDPbD-Req03	**DesignDataElements**
	WP5
Description	The PDPbD framework shall provide the Design Engineer the ability to model elements related to data structures and data instances.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have

[4] External data sources like for instance DBpedia (https://wiki.dbpedia.org/) or WikiData (https://www.wikidata.org/).

Type	Functional
Non-functional category	
Rationale	A data structure (e.g., SQL data base, excel table, etc.) can be modelled within the framework.

PDPbD-Req04	**ImportDataElements**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to import data structures and instances as model elements
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Could have
Type	Functional
Non-functional category	
Rationale	A data structure (e.g., SQL data base, excel table, etc.) can be imported within the framework as a model element.

PDPbD-Req05	**UpdateDataElements**
	WP5
Description	The PDPbD framework shall provide the Design Engineer the ability to update the attributes of data elements in the model.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	

Rationale	Data element attributes need to be editable.

PDPbD-Req06	**AnalyseDataModel**
	WP5
Description	The PDPbD framework shall provide the Design Engineer or the Data Analyst Engineer the ability to apply well-defined techniques on data elements in the model seeking privacy and data protection.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer, Data Analyst Engineer
Priority	Could have
Type	Functional
Non-functional category	
Rationale	Privacy and data protection techniques can be applied on data-oriented models as specified in ISO 27550 [16].

PDPbD-Req07	**DesignProcessElements**
	WP5
Description	The PDPbD framework shall provide the Design Engineer the ability to model elements related to processes involving data structures and data instances.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	

Rationale	System or software processes involving data need to be modelled.

PDPbD-Req08	**UpdateProcessElements**
	WP5
Description	The PDPbD framework shall provide the Design Engineer the ability to update the attributes of process elements in model
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	
Rationale	Process element attributes need to be editable.

PDPbD-Req09	**AnalyseProcessModel**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer or the Data Analyst Engineer the ability to apply well-defined techniques on process elements in the model seeking privacy and data protection.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer, Data Analyst Engineer
Priority	Could have
Type	Functional
Non-functional category	
Rationale	Privacy and data protection techniques can be applied on process-oriented models as specified in ISO 27550 [16].

PDPbD-Req10	**MaintainTraceabilityData-ProcessModels**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to add / edit traceability between the elements within and between data and process models.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Must have
Type	Functional
Non-functional category	
Rationale	Traceability between models and between elements within models is mandatory to ensure consistency and framework functionality.

PDPbD-Req11	**DesignArchitectureElements**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to model elements related to an architecture supporting data and process related elements.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	
Rationale	An architecture supporting data and process-oriented models need to be modelled.

PDPbD-Req12	**UpdateArchitectureElements**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to update the attributes of model elements within the architecture model
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	
Rationale	Architecture element attributes need to be editable.

PDPbD-Req13	**AllocateProcessToArchitectureModels**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to settle associations to allocate elements from the process model to elements within the architecture model.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	
Rationale	Allocation from process-oriented models towards an architecture model is part of the design space exploration activities [17][18] adapted to the purposes of PDP4E[5].

[5] Notice that the typical notion of design space exploration is adapted for the purposes of PDP4E. The PDP design moves from data and process oriented models to a functional and components architecture. The design space is explored during the allocation of processes and functions to components.

PDPbD-Req14	**RefineArchitectureModel**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to refine the architecture model thus providing internal and detailed views.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Could have
Type	Functional
Non-functional category	
Rationale	The refinement of the architecture model can be necessary in preparation for the deployment phase.

PDPbD-Req15	**MaintainTraceabilityRefinements**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to add / modify / keep traceability between related elements across different refinement models.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Must have
Type	Functional
Non-functional category	
Rationale	Traceability between related elements across refinements ensure overall model consistency and framework functionality.

PDPbD-Req16	**ReferenceExternalArtefacts**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the ability to add / modify / keep links from architecture elements to external artefacts, e.g., files including ANSI-C code.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Should have
Type	Functional
Non-functional category	
Rationale	Links to external artefacts help to detail/complete architecture model specification. They can be necessary to analyse privacy-related properties related to requirements to be fulfilled.

PDPbD-Req17	**AnalyseExternalArtefacts**
	WP5
Description	The PDPbD framework shall provide for the Design Engineer the necessary model references and information to apply algorithms to validate properties on external artefacts, e.g., code validation.
Assigned WP	WP5
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Could have
Type	Functional
Non-functional category	
Rationale	The properties to be validated on external artefacts are related to privacy and data protection.

PDPbD-Req18	CorrelateRequirementsToDesignModels
	WP5
Description	The PDPbD framework shall provide for the Design Engineer with the ability to correlate elements within data, process, and architecture models with the requirements to be fulfilled.
Assigned WP	WP5, WP4
Relation to other requirements	
Actor	(Privacy) Design Engineer
Priority	Must have
Type	Functional
Non-functional category	
Rationale	Correlation between design model elements and the requirements to be fulfilled is mandatory to improve certainty on privacy and data protection by design.

4.3 Requirements modelling

The requirements specified in section 4.2 have been modelled relying upon Papyrus-Req[6]. An overview of the model is presented in Figure 3. The figure corresponds to a typical SysML requirements diagram. For the deployment of the PDPbD framework, we adopt a MDE approach and consequently the requirements model will guide engineers during the development cycle. One of the salient features of the MDE approach is that it allows the development of new MDE tools. The features offered by Papyrus and Papyrus-Req can be thus exploited for the development of the extensions and customizations. For now, the requirements model mainly ensures their traceability and fulfilment. However, in other phases of the development cycle, the models will be elaborated as a basis prior to deployment.

[6] Available within Papyrus SysML module. In https://www.eclipse.org/papyrus/relatives.html.

Figure 3. Overview of the Papyrus model showing the requirements for the PDPbD framework

5 PDPbD Framework Use Cases

The Figure 4 shows an overview of the use cases involving the PDPbD Framework. The figure is obtained from the respective model (Papyrus Use Case diagram). For now, the use cases are purely functional and correspond to the requirements listed in previous section 4. As long as new non-functional requirements are elicited, the architecture shall also evolve and the use cases shall be accordingly updated. For now, the main actors intervening in PDPbD uses cases are the Data Analyst Engineer, the Design Engineer who can be also supported by the Privacy Design Engineer.

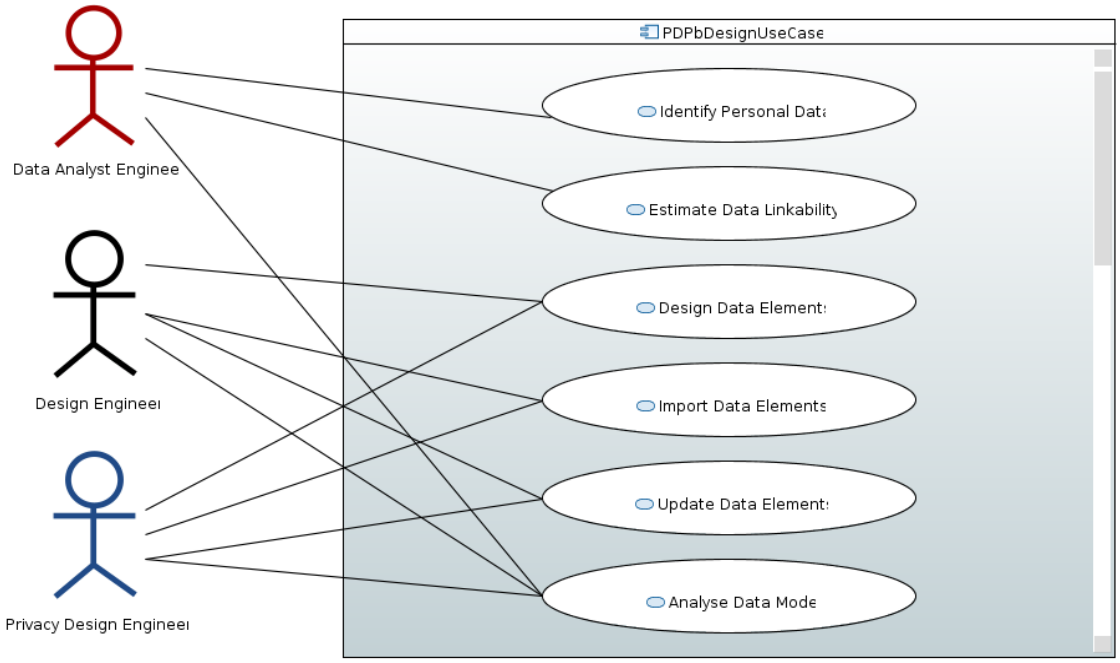

Figure 4. Excerpt of the Use Cases for the PDPbD Framework

- *Data analyst engineer:* this actor is able to apply techniques to conduct analyses over data structures. He intervenes in use cases where privacy related analyses on data and data models are carried out. The data analyst engineer mainly performs the identification of personal data and linkability estimations.
- *Design engineer:* it is expected that the design engineer is able to intervene in use cases where privacy-related tasks are conducted. The PDPbD tool should support non-savvy privacy engineers. Since the framework follows a MDE approach, the design engineer should create, import and update models and also maintain traceability and consistency between models and their refinements.
- *Privacy Design Engineer:* the privacy expert can also intervene during design tasks. He can collaborate with, guide or even supervise the design engineer during the design activities. The privacy engineer can conduct modeling activities as the design engineer and also work together with the data analyst model and data analysis.

Notice that there is no restriction for an organization to follow a specific distribution of roles among personnel. Thus, two or more roles can be played by the same person. For now, after this first iteration, the model includes 18 use cases which are expected to evolve according to tool deployment and consolidation steps.

6 PDPbD framework modules and interfaces

The PDPbD framework is initially composed by five modules:

- Personal Data Detector
- Papyrus module for Data-oriented models
- Papyrus module for Process-oriented models
- Papyrus module for Architecture models
- Module for requirements/properties V&V

A more detailed view of these modules is shown in Figure 5. The components are described in the following subsections 6.1 to 6.4.

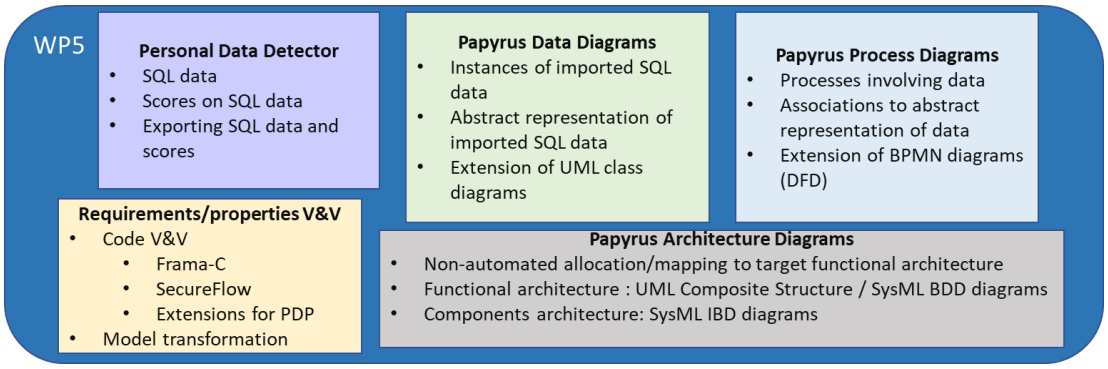

Figure 5. Modules of the privacy and data protection by design framework developed in WP5

6.1 Personal Data Detector Module

6.1.1 Overall description

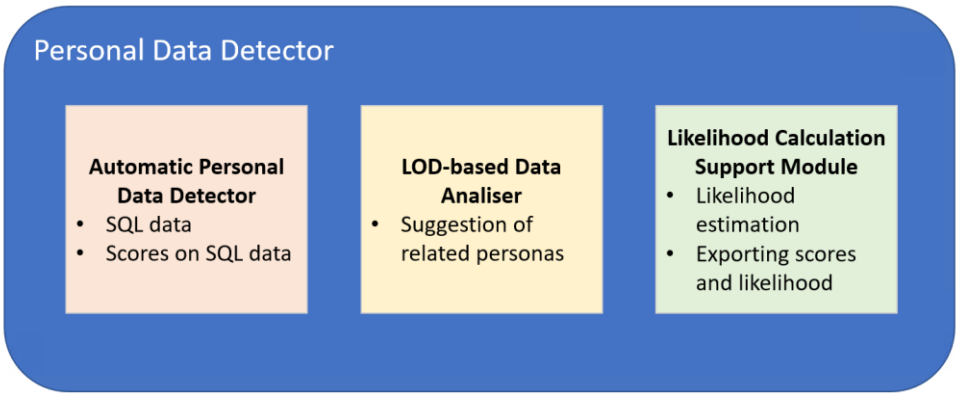

Figure 6. General Architecture of the Personal data detector

The Personal data detector is a component that will be implemented as part of the foreground of the PDP4E project. This component plays a crucial role in the accurate labelling of personal data, as it supports users to correctly detect and classify personal data in the context of the system under analysis.

The architecture of the Personal data detector consists of three main modules shown in Figure 6. Note that these three modules are aligned with the methodology described in D5.4 for the Personal data detector. We provide a detailed description of these three modules in the following sections.

6.1.2 Functional specification

Following we present a list with the main functions of the three modules of the Personal data detector.

MODULE NAME	MAIN FUNCTIONS
AUTOMATIC PERSONAL DATA DETECTOR	This module scans and detects personal data in SQL databases. The main inputs of this module are the SQL databases used as data storages in a data processing application. In particular, the tool will access: • The database schema including the catalogue with all the relations in the schema and the information about the attributes in each relation. • The data content of the different tables in the database schema This module will automatically scan the available information, including: • Looking for patterns that may represent specific types of personal data such as passport numbers, social security numbers, addresses, etc. • Establishing relationships between different attributes in different relations to look for instance for non-declared foreign-primary key relationships that indicate that two tables are linked. • Expanding pattern recognition for an attribute in a record to a broader context including related attributes (as designated by foreign keys or the above inference). The output of this module will be a set of scores both for relations and attributes within these relations that indicate the probability of a particular dataset of containing personal data in a particular system.
LOD-BASED DATA ANALYSER	Using the initial set of scores created by the Automatic personal data detector, the data analyser will try to extend the domain knowledge of the data in the system by exploring Linked Open Data (LOD) (such as WikiData[7] or DBpedia[8]). The goal of the detector will be to find extended concepts that can be classified as persons and may be linked with the data in the system in a way that it could represent a threat to the data subjects. In particular, keywords will be mined in the SQL database. Keywords can be extracted from the metadata in the schema (relation names, attribute names, etc.), extracted from the content of the data or provided in the form of tags by users. Given a keyword, this module will look for relationships between that keyword and persons (e.g. patients, operators, pilots, drivers, employees) that can be linked to that keyword. The main idea is finding

[7] WikiData (https://www.wikidata.org/).
[8] DBpedia (https://wiki.dbpedia.org/)

	links between entities that we contain in the data stores in the system with external stakeholders that are not present in these data stores but could be linked to these entities. This problem is inspired by the use case of PDP4E in the automotive sector, where the owner of the system may want to broadcast the position and direction of smart vehicles without realizing that this may disclose the route of the passengers or the driver of that vehicle jeopardizing their privacy and even safety and security. The owner of the system does not have information in their data stores about the driver or the passengers, but still it needs to consider the position and direction of the vehicle as personal data as an attacker could easily link these data with data about the occupants of the vehicle including their identity, through other means such as data sources external to the system. The LOD-based data analyser will help the owner of the system to reason about these hidden relationships.
LIKELIHOOD CALCULATION SUPPORT MODULE	For those concepts that were not previously classified as personal data, this tool will guide users in the estimation of the likelihood of a particular dataset being linked to one of the personas found by the LOD-based Data Analyser. The extended domain knowledge generated by the LOD-based Data Analyser will help in estimating the data that is required to achieve linkability, ranking likelihood in a scale from very low to very high. No matter what the likelihood is, the data in the database linked to these subjects will be recorded as personal data and the corresponding scores will be updated.

6.1.3 Module interfaces

Following we provide a list of the main inputs and outputs of each module of the Personal data detector. All these components will be created as foreground of the project.

MODULE NAME	INPUTS AND OUTPUTS

AUTOMATIC PERSONAL DATA DETECTOR	**INPUTS**:
	• Credentials to connect to the database and gain access to:
	  ○ The database schema including the information about the attributes in each relation.
	  ○ The data content of the different tables in the database schema
	OUTPUTS:
	• A set of scores both for relations and attributes within these relations that indicate the probability [0..1] of a particular dataset containing actual personal data in the particular domain of a system.

LOD-BASED DATA ANALYSER	**INPUTS**: • A set of scores, both for relations and attributes within these relations, that indicate the probability [0..1] of a particular dataset of containing actual personal data in the particular domain of a system. • Credentials to connect to the database to gain access to:

	○ The database schema including the information about the attributes in each relation. ○ The data content of the different tables in the database schema • APIs to linked open data resources: this may include using different access systems provided by existing open data sources such as WikiData, DBPedia, schema.org, etc. • A list of tags provided by the user for each relationship indicating the types of concepts represented in a particular relation. **OUTPUTS**: • A set of concepts representing potential types of data subjects (e.g. patient, driver, passenger) that may be related to the concepts represented in the SQL database of the system under analysis.
LIKELIHOOD CALCULATION SUPPORT MODULE	**INPUTS**: • A set of scores, both for relations and attributes within these relations, that indicate the probability [0..1] of a particular dataset of containing actual personal data in the particular domain of a system. • A set of concepts representing potential data subjects that may be related to the concepts represented in the SQL database of the system under analysis. **OUTPUTS**: • A set of updated scores, both for relations and attributes within these relations, that indicate the probability [0..1] of a particular dataset of containing actual personal data in the particular domain of a system.

6.1.4 Requirements coverage

- PDPbD-Req01: The PDPbD framework shall provide for the Data Analyst Engineer the ability to estimate whether personal-sensitive data are present within structured and unstructured sources.
- PDPbD-Req02: The PDPbD framework shall provide for the Data Analyst Engineer the ability to estimate the potential links between personal-sensitive data and external data sources.

6.2 Module for Data-oriented Models

6.2.1 Overall description

This module is developed on top of Papyrus [15] (see Figure 7). The module is a specialization of UML [11] to support modelling of data-related elements. The model includes one or more diagrams and is referred as data-oriented model. Along with the diagram front-end customization, the module is extended with functions to support the application of design strategies and related techniques for privacy and data protection (see deliverable D5.4 [6]). To ensure information flow along design, the module shall support importing of main outcomes from the Data Detector Module specified in section 6.1. In particular, the module shall generate

a UML data-oriented model representing a particular dataset including probabilities for personal and linkable data.

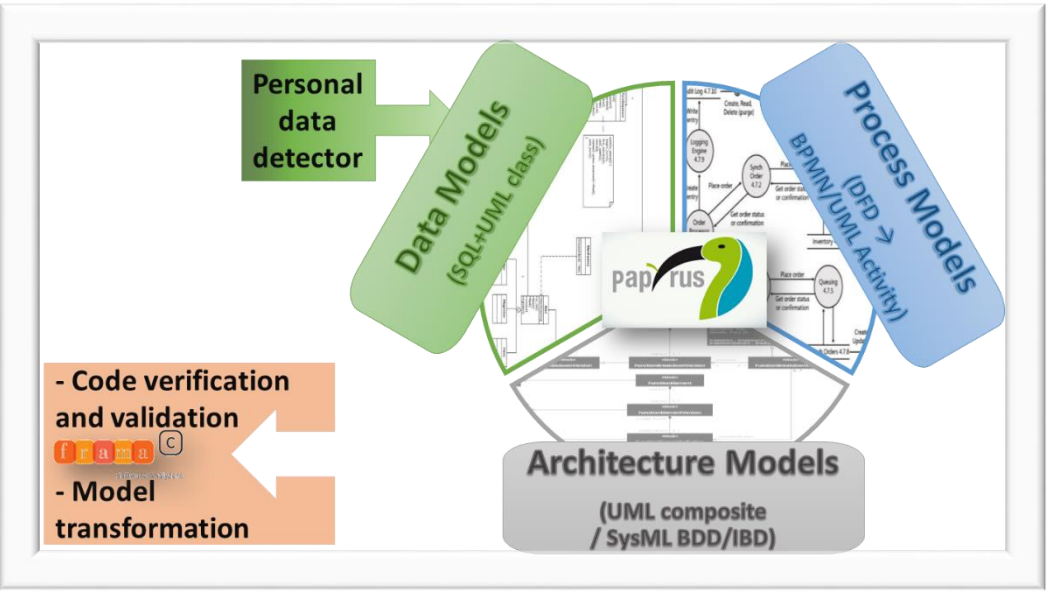

Figure 7. Overview of the PDPbD framework including data, process and architecture models

6.2.2 Functional specification

MODULE NAME	MAIN FUNCTIONS
PAPYRUS MODULE FOR DATA-ORIENTED MODELS	• Design of model elements representing data, data structures, and data instances. The model elements are stereotyped, and the stereotypes are aligned with the outputs from the Personal Data Detector specified in section 6.1. • Design of associations to connect/link model elements representing data, data structures and data instances • Update/complete/delete model elements • Import datasets including probabilities for detected personal data and linkability estimations as obtained from the Personal Data Detector specified in section 6.1 • Provide support to apply privacy and data protection techniques to data-oriented models as per suggested in deliverable D5.4, section 3.6 [6]. The following list includes candidate MDE techniques that can be leveraged for that purpose: o Library of data-oriented design strategies o Library of data protection techniques o Customized contextual menus

6.2.3 Module interfaces

MODULE NAME	INPUTS AND OUTPUTS
PAPYRUS MODULE FOR	**INPUTS**:

DATA-ORIENTED MODELS	• Instances of datasets obtained from structured/unstructured sources and analysed by the Personal Data Detector specified in section 6.1. The datasets can include probabilities related to detected personal data and data linkability estimations. • Plaintext written specification of a data structure to be modelled **OUTPUTS**: • UML data-oriented model including: o Elements representing data structures o Associations between model elements o Specialized attributes including outcomes from analyses o Traceability links to support model consistency and reusability o Traceability links for requirements management (validation, satisfaction). A candidate for this purpose is a data-requirement matrix

6.2.4 Requirements coverage

- PDPbD-Req03: The PDPbD framework shall provide for the Design Engineer the ability to model elements related to data structures and data instances.
- PDPbD-Req04: The PDPbD framework shall provide for the Design Engineer the ability to import data structures and instances as model elements.
- PDPbD-Req05: The PDPbD framework shall provide for the Design Engineer the ability to update the attributes of data elements in the model.
- PDPbD-Req06: The PDPbD framework shall provide for the Design Engineer or the Data Analyst Engineer the ability to apply well-defined techniques on data elements in the model seeking privacy and data protection.
- PDPbD-Req-18: The PDPbD framework shall provide for the Design Engineer with the ability to correlate elements within data, process, and architecture models with the requirements to be fulfilled.

6.3 Module for Process-oriented Models

6.3.1 Overall description

This module is to be developed on top of Papyrus [15] (see Figure 7). The module is a specialization of the BPMN language [10] to support modelling of process-oriented elements. The model includes one or more diagrams and is also referred as process-oriented model. It mainly implements the notion of Data Flow Diagram (DFD) [19], [20]. Along with diagram front-end customization, the module is added up with functions to support the application of design strategies and related techniques for privacy and data protection (see deliverable D5.4 [6]). To ensure information flow along design models, the module shall support tight association of elements within data-oriented models as designed in the Papyrus module specified in section 6.2. More specifically, the module shall keep consistency between DFDs and data-oriented models.

6.3.2 Functional specification

MODULE NAME	MAIN FUNCTIONS
PAPYRUS MODULE FOR PROCESS-ORIENTED MODELS	• Design of model elements representing: o processes involving data, o external entities, o logical and physical borders, o processing and storage units. The model elements are stereotyped, and the stereotypes are aligned with the definition of Data Flow Diagrams. • Design of associations to connect/link model elements representing processes involving data and related DFD notions • Update/complete/delete model elements • Settle traceability links with model elements as designed in the module for data-oriented models. • Provide support to apply privacy and data protection techniques to process-oriented models (i.e. DFDs) as per suggested in deliverable D5.4, section 3.7 [6]. The following list includes candidate MDE techniques that can be leveraged for that purpose: o Library of process-oriented design strategies o Library of data protection techniques o Customized contextual menus o Implementation of design patterns (see D5.4 [6], section 3.5): - Provider-receiver proof pattern - Proof of endorsement pattern

6.3.3 Module interfaces

MODULE NAME	INPUTS AND OUTPUTS
PAPYRUS MODULE FOR PROCESS-ORIENTED MODELS	**INPUTS**: • Data-oriented model as obtained from Papyrus module specified in section 6.2. • Specification of a system or software process involving data **OUTPUTS**: • Specialized BPMN model (DFD) including: o Elements representing processes involving data o Associations between model elements o Specialized attributes including outcomes from analyses o Traceability links to support model consistency and reusability o Traceability links for requirements management (validation, satisfaction). A candidate for this purpose is a process-requirements matrix

6.3.4 Requirements coverage

- PDPbD-Req07: The PDPbD framework shall provide for the Design Engineer the ability to model elements related to processes involving data structures and data instances.
- PDPbD-Req08: The PDPbD framework shall provide for the Design Engineer the ability to update the attributes of process elements in model.
- PDPbD-Req09: The PDPbD framework shall provide for the Design Engineer or the Data Analyst Engineer the ability to apply well-defined techniques on process elements in the model seeking privacy and data protection.
- PDPbD-Req10: The PDPbD framework shall provide for the Design Engineer the ability to add / edit traceability between the elements within and between data and process models.
- PDPbD-Req-18: The PDPbD framework shall provide for the Design Engineer with the ability to correlate elements within data, process, and architecture models with the requirements to be fulfilled.

6.4 Module for Architecture Models

6.4.1 Overall description

This module is developed on top of Papyrus [15] (see Figure 7). Two candidate languages are currently considered to support architecture models: UML composite structures [11] and SysML Block Definition Diagrams [12]. Refinements of architecture models can be performed via SysML Internal Block Diagrams [12]. The set of models supported by this module are also referred as architecture models. Along with interface customization, the module is added up with functions to support the allocation of process-oriented models from the module specified in section 6.3. The allocation tasks are briefly described in deliverable D5.4 [6]. To ensure design flow, a tight association between data and architecture models is to be ensured.

6.4.2 Functional specification

MODULE NAME	MAIN FUNCTIONS

PAPYRUS MODULE FOR ARCHITECTURE MODELS	• Design of model elements representing:
	  ○ Systems and SW architectures,
	  ○ Internal subcomponents,
	  ○ Communication ports,
	  ○ Interfaces
	  ○ Connectors,
	The model elements are stereotyped and these stereotypes can be added up with privacy and data protection concepts.
	• Design of associations to connect/link model elements
	• Update/complete/delete model elements
	• Settle traceability links from model elements as obtained from the module for process-oriented models specified in section 6.3.
	• Provide support to conduct architecture refinements

6.4.3 Module interfaces

MODULE NAME	INPUTS AND OUTPUTS

PAPYRUS MODULE FOR ARCHITECTURE MODELS	**INPUTS**:
	• Process-oriented model as obtained from Papyrus module specified in section 6.3.
	• Plaintext written specification of a system or software architecture involving data
	OUTPUTS:
	• Specialized architecture model(s):
	  ○ UML composite structure
	  ○ SysML Block Definition Diagram
	  ○ SysML Internal Block Diagram
	• Architecture models can include:
	  ○ Associations between model elements
	  ○ Traceability links to support model consistency and reusability
	  ○ Traceability links for requirements management (validation, satisfaction). A candidate for this purpose is a process-requirements matrix

6.4.4 Requirements coverage

- PDPbD-Req11: The PDPbD framework shall provide for the Design Engineer the ability to model elements related to an architecture supporting data and process related elements.
- PDPbD-Req12: The PDPbD framework shall provide for the Design Engineer the ability to update the attributes of model elements within the architecture model.
- PDPbD-Req13: The PDPbD framework shall provide for the Design Engineer the ability to settle associations to allocate elements from the process model to elements within the architecture model.
- PDPbD-Req14: The PDPbD framework shall provide for the Design Engineer the ability to refine the architecture model thus providing internal and detailed views.
- PDPbD-Req15: The PDPbD framework shall provide for the Design Engineer the ability to add / modify / keep traceability between related elements across different refinement models.
- PDPbD-Req16: The PDPbD framework shall provide for the Design Engineer the ability to add / modify / keep links from architecture elements to external artefacts, e.g., files including ANSI-C code.
- PDPbD-Req-18: The PDPbD framework shall provide for the Design Engineer with the ability to correlate elements within data, process, and architecture models with the requirements to be fulfilled.

6.5 Module for Code Validation

Privacy and Data Protection requirements stated at the architectural level may be verified by different means. From architecture-level design to binary-level implementation, a large range of possibilities exist depending on the kind of property to guarantee and the level of assurance expected. In particular, some of these properties can be verified at the source code level by

using static analysis tools. In such cases, specifications correspond to low-level privacy and data protection requirements while implementations are a source code module for which the satisfaction of the specification is expected. The module for code validation is an important component of the PDPbD architecture since (1) it helps to produce formal evidence of the requirements fulfilment and (2) it increases the certainty and soundness on the properties validated.

In the context of PDP4E, a tool dedicated to verification of data flow in C-based programs will be used and developed to cover privacy and data protection requirements and validated on a use case for which modules written in this language handle personal data. This module can be helpful to verify low-level properties related to improper variable usage (e.g., interference), or data transmission over unintended channels (data leaks). In the scope of PDP4E, we plan to explore whether other high level properties or principles, like for instance purpose, could be defined in terms of basic ones and proved.

Interaction of Frama-C/SecureFlow with other languages, such as C++ for instance, would improve the applicability of this validation method by increasing its application domain to a larger set of codebases. Though under investigation in other projects, current and foreseeable status of this line of work is too early-stage to be applicable in the scope of PDP4E.

6.5.1 Overall description

Frama-C (http://frama-c.com) [21] is an open source industrial-strength framework for analysis of C source code. Its license is LGPL v2.1. Its Eclipse-like software architecture, presented in Figure 8 - Frama-C Software Architecture, allows any developer to plug custom code analyses or program transformations (developed in OCaml) in the framework. For that purpose, Frama-C is based upon a *kernel*. Its internals are in charge of parsing, typing and linking the C source codes in order to build a normalized abstract syntax tree (AST) suitable for code analysis. It is shared by all plug-ins that consequently analyze the very same code. The Frama-C kernel also provides several services that help plug-in interactions, ease code manipulations often performed by code analyzers and program transformers and provide a uniform setting for the end-users (e.g. a common GUI). Several general-purpose libraries are also provided.

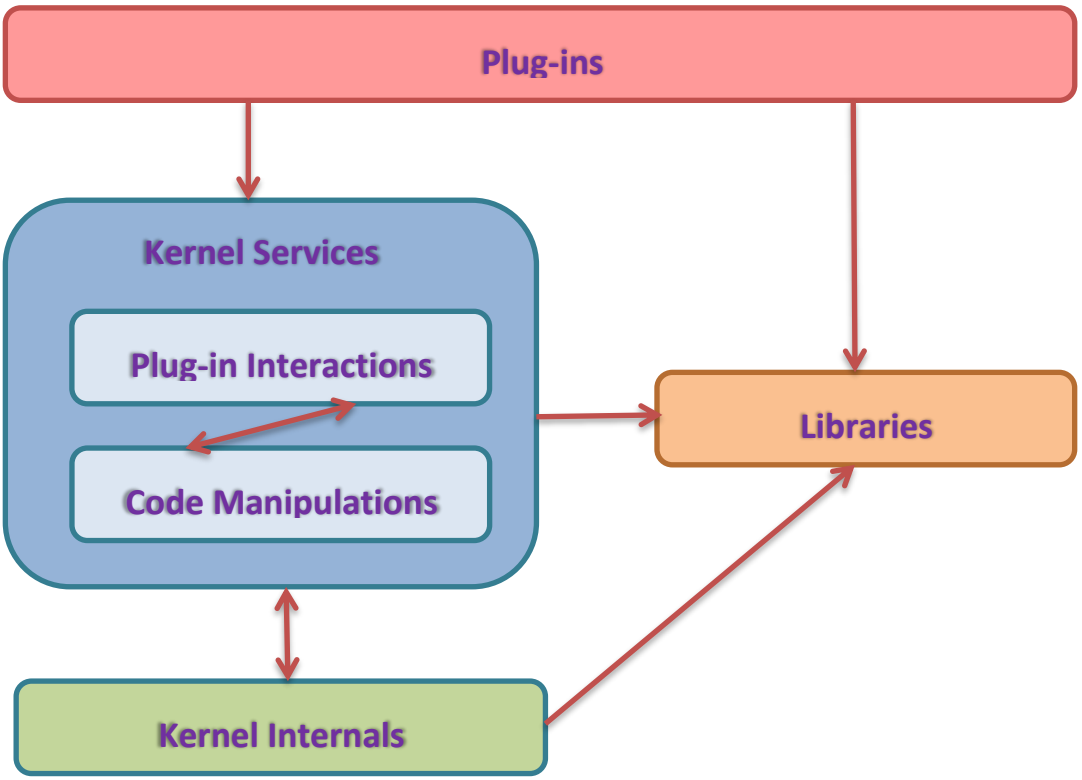

Figure 8 - Frama-C Software Architecture

6.5.2 Functional description

Frama-C and its plug-ins analyze C source codes together with formal specifications written in the ACSL specification language [22]. This behavioral specification language is particularly suitable to express functional program properties.

The official open-source distribution contains 28 plug-ins (latest version: Frama-C 19.0 Potassium released in June 2019). The three main important ones are dedicated to program verification as described in the following items:

- Plug-in *WP* is dedicated to proving the functional behaviors of a program by ensuring that each function satisfies its specification (written as an ACSL contract).
- Plug-in *Eva* is suitable to statically verify that a program does not contain any undefined behaviors (as defined in the ISO C99 norm).
- Plug-in *E-ACSL* checks at runtime (during program execution) that ACSL properties are valid. It is also able to check that the current execution has no undefined behaviors.

In addition to the publicly distributed plug-ins, CEA LIST also develops several close-source plug-ins. Among others, plug-in *SecureFlow* is dedicated to verifying the absence of information flow leakage [23]. It aims at verifying non-interference properties such as ensuring that no confidential data (e.g. the contact list of some user) leaks on a public channel (e.g. Internet). This plug-in should be of primary importance to check privacy properties at source code level, but the possibility of developing custom plug-ins might be of interest in that respect too.

6.5.3 Requirements coverage

- PDPbD-Req17: The PDPbD framework shall provide for the Design Engineer the necessary model references and information to apply algorithms to validate properties on external artefacts, e.g., code validation.

7 Summary

This report includes the first specification of the architecture to support design engineers when seeking privacy and data protection by design. The PDPbD framework integrates five modules and covers relevant design tasks like personal data identification and linkability estimations, modelling of data structures and instances, modelling of processes involving data (in the form of DFDs), modelling of system architecture and validation of requirements at different levels, including at code level. The code validation has been identified as a relevant component of the architecture given that it can produce evidence of requirements fulfilment and increase certainty on privacy properties at code level. The specification relies upon the Eclipse and Papyrus platforms which are the basis for Papyrus-based extensions. The modules for personal data identification and linkability analyses are independent modules to be developed in the scope of PDP4E. The module for code validation mostly relies upon Frama-C and SecureFlow. The document includes a set of functional requirements elicited to support non-savvy privacy engineers to accomplish design tasks oriented to achieve privacy and data protection. The requirements intersect the designer needs and the obligations imposed by the GDPR. A first description of the different PDPbD modules has been given including the functions and interfaces to be implemented. Each module is associated with the requirements covered. The overall architecture is aligned with and provides support to the method for PDPbD specified in [6]. This specification is expected to evolve according to the deployment, harmonization and consolidation phases in the work plan.

8 Bibliography

[1] The PDP4E consortium, Deliverable D3.1, *"Specification and design of risk management tool for data protection and privacy"*, Technical report of PDP4E, June 2019.

[2] The PDP4E consortium, Deliverable D3.4, *"Risk management methods for privacy and data protection"*, Technical report of PDP4E, June 2019.

[3] The PDP4E consortium, Deliverable D4.1, *"Specification and design of requirements engineering tool for privacy and data protection"*, Technical report of PDP4E, June 2019.

[4] The PDP4E consortium, Deliverable D4.4, *"Requirements engineering methods for privacy and data protection"*, Technical report of PDP4E, June 2019.

[5] The PDP4E consortium, Deliverable D5.1, *"Specification and design of model-driven design tool for privacy and data protection"*, Technical report of PDP4E, June 2019.

[6] The PDP4E consortium, Deliverable D5.4, *"Methods for data protection model-driven design"*, Technical report of PDP4E, June 2019.

[7] The PDP4E consortium, Deliverable D6.1, *"Specification and design of assurance tool for data protection and privacy"*, Technical report of PDP4E, June 2019.

[8] The PDP4E consortium, Deliverable D6.4, *"Assurance methods for data protection and privacy"*, Technical report of PDP4E, June 2019.

[9] The PDP4E consortium, Deliverable D2.4, *"Overall system requirements"*, Technical report of PDP4E, June 2019.

[10] The Object Management Group, *"Business Process Model And Notation"*. BPMN Specification 2.0, Available in https://www.omg.org/spec/BPMN/2.0/About-BPMN/, 2019.

[11] The Object Management Group, "Unified Modelling Language". UML Specification 2.5.1, Available in https://www.omg.org/spec/UML/About-UML/, 2019.

[12] The Object Management Group, *"System Modelling Language"*. SysML Specification 1.6, Available in https://www.omg.org/spec/SysML/About-SysML/, 2019.

[13] The Object Management Group, *"UML Profile MARTE Language"*. Specification 1.2, Available in https://www.omg.org/spec/MARTE/About-MARTE/, 2019.

[14] The Object Management Group, *"Semantics of a Foundational Subset for Executable UML Models"*. Specification 1.4, Available in https://www.omg.org/spec/FUML/About-FUML/, 2019.

[15] The Eclipse Foundation, *"Eclipse Papyrus Modelling Environment"*, Available in https://www.eclipse.org/papyrus/, 2019.

[16] The International Organization for Standardization, *"ISO/IEC PRF TR 27550 - Information technology -- Security techniques -- Privacy engineering"*, Available in https://www.iso.org/standard/72024.html, 2019.

[17] S. Pandey, M. Glesner and M. Muhlhauser, *"Architecture level design space exploration and mapping of hardware," International Symposium on Signals, Circuits and Systems, 2005. ISSCS 2005.*, Lasi, Romania, 2005, pp. 553-556 Vol. 2.

[18] J. Peng, S. Abdi and D. Gajski, *"Automatic model refinement for fast architecture exploration [SoC design]," Proceedings of ASP-DAC/VLSI Design 2002. 7th Asia and South Pacific Design Automation Conference and 15h International Conference on VLSI Design*, Bangalore, India, 2002, pp. 332-337.

[19] Lucid Software Incorporated, *"What is Data Flow Diagram"*, Available in https://www.lucidchart.com/pages/data-flow-diagram, 2019.

[20] Kim Wuyts and Wouter Joosen, LINDDUN privacy threat modeling: a tutorial, Technical Report (CW Reports), volume CW685, Department of Computer Science, KU Leuven, July 2015, Available in https://linddun.org/downloads/LINDDUN_tutorial.pdf, 2019.

[21] Florent Kirchner, Nikolai Kosmatov, Virgile Prevosto, Julien Signoles, and Boris Yakobowski. Frama-C: A Software Analysis Perspective. Journal of Formal Aspects of Computing, January 2015.

[22] Patrick Baudin, Jean-Christophe Filliâtre, Claude Marché, Benjamin Monate, Yannick Moy and Virgile Prevosto. ACSL: ANSI/ISO C Specification Language. http://frama-c.com/acsl.html.

[23] Gergö Barany and Julien Signoles. Hybrid Information Flow Analysis for Real-World C Code. In International Conference on Tests and Proofs (TAP), July 2017.

[24] P. Krief, “Open-source licensing model and IPR governance framework,” Deliverable D7.5, Oct. 2018.

[25] L. Vogel, J. Bresson, and D. Roy, “Eclipse Platform project,” 24-Mar-2016. [Online]. Available: https://wiki.eclipse.org/Platform.