Methods and tools for GDPR Compliance through

Privacy and Data Protection 4Engineering

D2.4 Overall system requirements

Project:	PDP4E
Project Number:	787034
Deliverable:	D2.4
Title:	Overall system requirements
Version:	v1.1
Date:	06/08/2019
Confidentiality:	Public
Author:	Yod Samuel Martín (UPM)

Funded by

Table of Contents

Document History

Version	Status	Date
V0.9	Complete draft	30/07/2019
V1.1	Reviewed and revised version	01/07/2019

Approval		
	Name	**Date**
Prepared	Yod Samuel Martín (UPM)	30/07/2019
Reviewed	Jabier Martinez (Tecnalia)	01/08/2019
Reviewed	Victor Muntés Mulero (Beawre)	01/08/2019
Reviewed	Gabriel Pedroza (CEA)	02/08/2019
Reviewed	David Sánchez (TRIALOG)	02/08/2019
Revised	Yod Samuel Martín (UPM)	06/08/2019
Authorised	Antonio Kung	09/08/2019
Circulation		
Recipient	**Date of submission**	
Project partners	09/08/2019	
European Commission	09/08/2019	

List of Figures

List of Tables

Abbreviations and Definitions

Abbreviation	Definition
CISO	Chief Information Security Officer
CJEU	Court of Justice of the European Union
CNIL	Commission Nationale de l'Informatique et des Libertés
CTO	Chief Technology Officer
DFD	Data-Flow Diagram
DPIA	Data Protection Impact Assessment
DPO	Data Protection Officer
EDPB	European Data Protection Board
GDPR	General Data Protection Regulation
ICT	Information and Communication Technologies
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
LINDDUN	Linkability, Identifiability, Non-repudiation, Detectability, Disclosure of information, Unawareness, Non-compliance
MDE	Model-Driven Engineering
OML	Open Modelling Language
PDP	Privacy and Data Protection
PDP4E	Privacy and Data Protection 4 Engineering
PMO	Project Management Office
ProPAn	Problem-based Privacy Analysis
SDLC	Systems Development Life Cycle
SIPOC	Supplier Input Producer Output Customer
STRIDE	Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege
SWEBOK	Software Engineering Body of Knowledge
UML	Unified Modelling Language
WP29	Article 29 Data Protection Working Party

Executive Summary

This document provides an overall vision of the functions expected from PDP4E methods and tools, which address several software and systems engineering disciplines where privacy and data protection related activities are to be introduced, namely:

- Risk Management,
- Requirements Engineering,
- Model-Driven Design, and
- Systems Assurance.

This document comes to provide an overall vision of the requirements that define the PDP4E toolset as a whole. However, methods and tools will be developed to cover each of the aforesaid disciplines in WP3, WP4, WP5 and WP6, each WP producing an individual subsystem that supports the respective discipline. Thus, instead of providing here a detailed description of the specific system requirements for each subsystem, this document delivers a framework of requirements that supports its refinement for the different disciplines in the respective WP, whose deliverables go into the details of their specific requirements.

From a discipline-centric perspective, we address the traditional activities for each discipline, here adapted to the context of privacy and data protection (so that they do not seem completely new to non-privacy-experts, yet they address privacy aspects). SIPOC diagrams (Suppliers, Inputs, Process, Outputs, and Customers) are used to formally model functional requirements of each subsystem. These diagrams also include information about the interactions between a subsystem and the outside, and the relation of the processes that implement those functions with external agents, hence indirectly illustrating potential dependencies among functions. Besides, we introduce some variability points in the process of each discipline, which represent some of the questions that the detailed requirements in each WP shall deal with.

From a cross-discipline perspective, some traits are shared among many of the steps in different methods. Generic use cases have been abstracted from the different methods, to show the commonalities among them:

1. A privacy and data protection framework (e.g. GDPR, but not only that) is modelled.
2. That framework is tailored to select those parts which are applicable to the needs of a project or organization.
3. Apart from that, one or more system views are modelled, including properties which are relevant regarding privacy and data protection.
4. The privacy and data protection framework is instantiated with the specific values of framework parameters in a given project.
5. The system model views are transformed, in order to address privacy and data protection properties and improve the system quality from the PDP perspective.
6. The system (or its models) are assessed against compliance with the given PDP framework.
7. External providers are also evaluated regarding compliance.

These use cases show a potential way of application of PDP4E toolset to development cycle engineering.

1 Introduction

1.1 Objective of the document

The objective of this document is to provide a holistic view of the features to be provided by the toolset developed in PDP4E from a functional perspective, taking into account the previously elicited multi-stakeholder needs, and the fact that the engineering disciplines to be addressed by PDP4E and the respective background tools are established beforehand. It should be noted that this document is provided as a general overview, but the features to be finally implemented by each of the tools will depend on their prioritization, according to their relevance for the demonstration scenarios, their feasibility in the scope of the project time frame, etc.

1.2 Structure of the document

The document is organized as follows. Section 1 introduces the objective, structure and relation with other deliverables. Then, functional system requirements are presented from different perspectives; always taking into account that we are creating a toolset composed of individual subsystems which autonomously address different disciplines. Section 2 provides an integral overview of the scope of the PDP4E toolset and introduces the standards and notations to be used in their detailed description in the subsequent sections. Section 3 presents the functions to be offered by the methods and tools from the perspective of each of the different disciplines undertaken by PDP4E, and some open considerations to be addressed by the respective WPs. Section 4 shows an orthogonal perspective: an abstraction of common traits to all the disciplines in cross-cutting use cases.

1.3 Relation with other deliverables

This document is the result of Task 2.4 Architectural analysis and overall system requirements. This deliverable represents a crossroads in the progress of the project:

- It grounds the multi-stakeholder requirements and needs (legal, business, and users') synthesized in D2.1 [1], D2.2 [2] and D2.3 [3].
- It paves the way for the technical specification of the methods and tools for each of the four disciplines, to be provided in their respective deliverables, namely Risk Management (D3.1 [4] and D3.4 [5]), Requirements Engineering (D4.1 [6] and D4.4 [7]), Model-Driven Design (D5.1 [8] and D5.4 [9]) and Systems Assurance (D6.1 [10] and D6.4 [11]).

It is also aligned with the results of the common architectural and methodological framework, gathered by D2.6 [12]. In any case, the alignment between both will improve in the following iterations during implementation and consolidation phases.

This document will be refined in subsequent versions (D2.5) as the development activities progress, the results from demonstration scenarios are fed back for their analysis in the next project iteration and they are reused to refine the requirements, and the architecture becomes eventually frozen (D2.7, D2.8).

2 Overview

2.1 Product scope

PDP4E aims to create a set of tools and methods for non-privacy-expert software and systems engineers to systematically apply privacy and data protection principles in the development projects they undertake, by integrating best privacy and data protection practice into the general-purpose engineering tools and methods they already employ as part of their daily activity. In particular, PDP4E is addressing a set of *pre-defined engineering disciplines* which represent some everyday activities of engineers, and chosen because they can appropriately address some of the needs posed by legal and regulatory privacy and data protection frameworks (see D.2.3 [3]).

These disciplines are *supported by tools*, whose background versions have already been used in order to provide similar functions to support those disciplines in other fields (e.g. safety), and in whose original development PDP4E partners had been heavily involved. It is thus within these disciplines that the different tools and methods are going to be entrenched: each discipline will be based on their own tools (according to the usual practice in the respective discipline), which will behave as loosely coupled subsystems, as any dependency would be solved through asynchronous file import and export. The tools provide non-privacy-expert engineers with software support to execute the methods defined in the project: some of the functions can be completely automated by the tool (e.g. model transforms or analysis according to predefined algorithms), others they will require a cooperation from the engineers, who may be guided by the tools to provide specific inputs, follow steps (in a wizard-like fashion), etc.

The disciplines and the respective software tools and functions that define the scope of the products to be developed in PDP4E include:

- **Risk Management**: traditional risk management activities are dealt with from the perspective of privacy and data protection (e.g. DPIAs), with the support of an extension of the MUSA decision support tool. That is, PDP threats and vulnerabilities are elicited, PDP risks likelihood and impact are analyzed, and risks are prioritized and treated through security and privacy controls. Other relevant tools which may be addressed and may require integration include open-source, traditional DPIA tools (e.g. CNIL's DPIA tool), as well as tools for the selection of vendors (i.e. data processors) depending on the controls they implement.
- **Requirements Engineering**: privacy and data protection requirements are specified, analyzed, and traced, leveraging the model-driven requirements management features implemented in Papyrus. PDP requirements include those directly derived from normative texts and others from the application of problem-frames based elicitation methods such as ProPAn; all of them particularized to the specifics of each project.
- **Model-Driven Design**: several views of system models are annotated with PDP related attributes (e.g. categories of personal data, data processing operations, controller and processor realms), analyzed against specific PDP constraints, and transformed according to predefined strategies to improve the compliance of the system with PDP attributes. All this is supported by the model-driven engineering features provided by Papyrus, on top of which PDP-oriented metamodels will be created that support the said attributes and transformations. Besides, newly created semi-automated tools will deal with the analysis of existent structured and unstructured personal data stores in order to create the annotations mentioned; and analysis tools based on the Frama-C framework will analyze PDP properties in source code files.

- **Assurance**: evidences are collected and, upon them, claims of compliance are issued to demonstrate compliance with a normative framework (which has been appropriately modelled beforehand).

2.2 Formalisms and notations employed

The arrangement of the document is freely inspired by the structure proposed by IEEE 830 [13], a standard for Software Requirement Specifications which provides a rigorous yet lightweight framework to specify such requirements. However, as we are creating a toolset composed by a set of autonomous subsystems, the organization of the requirements is adapted to such scenario. Thus, rather than describing the specific system requirements in detail, this document provides a framework of requirements that supports its refinement in the deliverables produced by each WP for the respective disciplines. First, functional requirements of each subsystem are expressed through SIPOC diagrams (Section 3), focusing on the interactions between a subsystem and the outside, together with the variability points that represent the questions that requirements in each WP shall respond to. Second, the common abstract use cases are elicited (Section 4.1), to allow organizing the methods according to a common structure.

2.2.1 SIPOC diagrams

In order to model the functions addressed by each subsystem in section 3, we have heavily leveraged the use of SIPOC diagrams. SIPOC (for Supplier – Input – Producer – Output – Customer) is both a general-purpose process modelling approach and a lightweight visual notation for process description, typically employed to define processes in the context of Six Sigma process improvement quantitative methods [15], and emphasizing the interactions between a (sub-)system and the outside. A SIPOC diagram provides, in five columns, from left to right:

- Suppliers which provide the process inputs (no process can be addressed unless there is someone providing all the inputs).
- Inputs needed for the process.
- Process, where transformation (from inputs to outputs) is sketched in the diagram (without going into details).
- Outputs, also known as units, generated or transformed by the process.
- Customers, to which the process outputs are addressed and who are interested in consuming them (no output can be generated if there is no one who wants it).

The use of SIPOCs in an organizational system whose autonomous parts are carrying out different functions has been proved useful to check the consistency between one another; in particular, in the field of privacy engineering [16]. Thus, when different SIPOCs are combined, it can be checked that e.g. all the inputs are either provided as outputs by another process or explicitly marked as external, all the suppliers of internal inputs match the roles responsible of the respective process, there are no dangling inputs or suppliers, etc. (and likewise regarding outputs and customers).

Strictly speaking, the standardized representation of a SIPOC diagram merely lists the above-mentioned elements in contiguous columns. However, the content of the process is typically decomposed according to a process diagram which lists the steps involved there (e.g. movements and transformations to produce outputs from inputs). Inputs and outputs are shown connected to the process step where they belong [17]. Different, non-standardized notations are employed to depict specific types of elements. In particular, we have chosen the one shown in Figure 1:

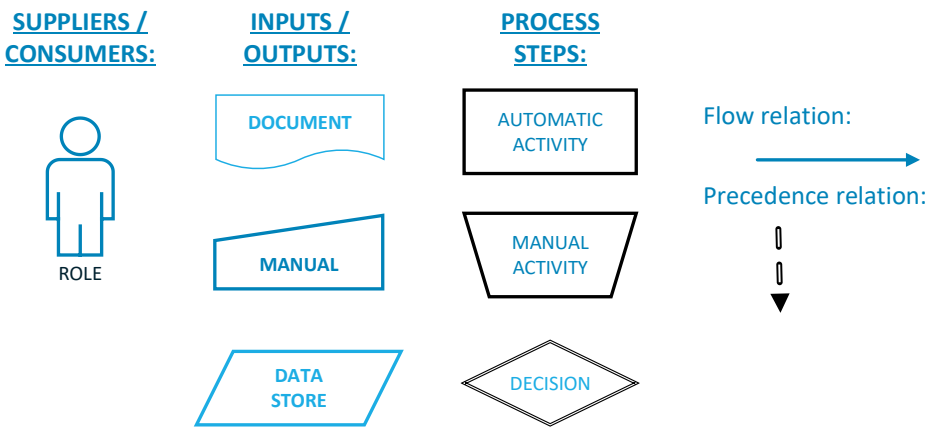

Figure 1. Notation for SIPOC diagrams used in this document.

Even though we distinguish between automatic and manual activities, in practice, most of the activities in PDP4E will include some components of both: manual activities will usually be supported by software tools that guide the process, while automatic activities may still require the input by the user to e.g. configure and start the process. All the roles, inputs, outputs and activities should be understood as an initial approach, which will be refined in the contents of the respective WP deliverables.

In order to facilitate the understanding of the SIPOC diagrams that follow, some of its features are highlighted next:

- A SIPOC supplier does not represent the agent that carries out a given process (as it would be in, e.g. a use case diagram), but the agent that, having carried out another process, provides a necessary input.
- A SIPOC is focused on reasoning why an activity within a process shall and can be carried out by showing which are the outputs expected by customers and the inputs requested from providers; thus, the actor who carries out the process is not explicitly identified.
- A SIPOC does not show a dataflow; thus, the agents are not global but related to the role they play in specific steps. Consequently, the same agent can appear more than once in a given diagram (as supplier, customer, or both).
- A SIPOC shows the linear dependencies between the stages of a process; thus, iterations are not explicitly rendered, but this doesn't mean that a given process cannot execute more than once in a project.

The functional specification for each discipline's tools, provided in section 3 and described around the SIPOC diagrams, is necessarily open and subject to refinement by the detailed specification of each work package. Thus, together with each SIPOC, we advance some of the considerations that each WP shall address when describing their methods and tool functionalities. These open issues are supplied as a kind of open parameters to the system requirement specifications, to be bound in the detailed specifications produced by the WP3-6. In any case, this is not a closed list, as those WPs specifications need alto deal with their own issues which are not considered at this stage.

2.2.2 Abstract use case descriptions

Uses cases are a well-known formalism to specify the functionality, context and added value of a system. UML defines a standard use case model that allows relating use cases with one another and with actors; however, the specific template to be used to define the contents of each use case is open. Consequently, different approaches exist to define use cases [18][19]: in our case, we'll define them sticking to a brief, black-box, essential style (i.e. without detailing all the steps

nor the internals of the implementation), described using consistent, structured prose written using a descriptive tone, and focusing on cross-cutting subfunction goals at the semantic interface between the system and actors.

The last phrase is worth further discussion. As above explained, we're defining the functions to be provided by a series of loosely coupled methods and tools for PDP engineering, developed as extensions of existent software and systems engineering methods and tools, each addressing its own discipline. On the one hand, we depart from background methods and tools which had been a priori created independently from one another. On the other one, following the method engineering paradigm, we strive for reusability of each element of the PDP4E toolset on their own, fostering the introduction in any other method. Thus, we both leverage and aim for such loose coupling. Anyway, each tool will implement different functions (corresponding to the respective discipline), and thus each has its own actors and internal use cases, different from those of one another. These tool-specific use cases will be described in the deliverables that contain the functional specifications of the respective tools (D3.1 [4], D4.1 [6], D5.1 [8] and D6.1 [10]). Nonetheless, the work in each of the different disciplines is progressing in parallel, always being aware of the inputs and outputs of other methods, and the dependencies between different tools. Thus, the use cases of one discipline satisfy all together the goals of a given user (the engineer of the respective discipline) and may interact with external use cases enacted by other disciplines by providing inputs and using their outputs.

Despite that each discipline addresses separate functions, we have considered that some common traits can be abstracted from different use cases of separate disciplines, and we have created a set of abstract use cases that encapsulate those common behaviours. They are shared, because they are common to several disciplines, and they are necessarily abstract because of the same reason. For instance, in a given project it may be the case that: in the Risk Management methods, threat trees can be pruned depending on assumptions made about the project; in the Requirements Engineering method, some GDPR-derived requirements are filtered out for not being applicable to the system at hand; and in Model-Driven Design or in Assurance, design patterns and argumentation patterns are respectively selected depending on the context of application in relation to the project scope. All of them follow the same approach: take an existing, externally provided knowledge base, and tailor it to the specifics of a given project by selecting the elements which are appropriate. The ultimate goal of this approach is to be able to define a coherent methodology in consistent terms, which encompasses the different methods provided by the different WPs to be applied in their respective disciplines; easing as well the reuse of knowledge among different disciplines.

As above mentioned, there is no standard template for the contents of each use case, and different authors have provided different variations. In our case, we have departed from templates proposed by Larman [19], Wiegers [20] and Cockburn [21], and adapted them to cover our abstract description in this first iteration. This template provides detailed, specialized use case attributes (e.g. relation to GDPR) to cover PDP4E needs. It should be noted that our template does not deal with the prioritization of these abstract use cases, since that will be addressed by the detailed specifications of each of the concrete use cases of the individual disciplines (described within the deliverables of the respective WPs). The use case template we have finally used follows.

Table 1. Use case template.

Use case name	
Purpose	Value provided by this use case.

Actors / Stakeholders	Actors (roles or external systems) that trigger the use case, get value from it, or are necessary for it to be completed.
Trigger	Event or action that initiates the use case.
Preconditions	Previously required activities or necessary inputs.
Assumptions	Assumptions regarding the use case (e.g. external available resources).
Post-conditions	Results guaranteed to be available after the use case has executed.
Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	
Overall description of the responses of the system to the user actions, focusing on the value provided by the system. As we are providing a brief essential version of the use cases, we have merged in a single field the overall description of the main scenario for the use case, together with the detailed description of its steps, if available, and, if applicable, any variations, exceptions, extensions or alternatives.	
Related use cases	References to other use cases related to the current one. In our case, two types of relations are expected: - Precedence relations (dependency relations with the custom stereotype <<precedes>>) - Generalization or specialization relations (to the specific use cases of each discipline).
Relation to GDPR	GDPR articles which come to be directly or indirectly satisfied by this use case. It corresponds to what is termed as 'business rules' in other, general-purpose templates (as business rules define general needs which are implemented by the use case).
Frequency of use	Relative frequency with which the use case will be enacted.

The set of such use cases will be summarized all together in a use case diagram. It shall be noted that we have added a <<precedes>> stereotype to some dependencies between use cases: this not standard UML but a custom stereotype defined by OML (Open Modelling Language) [23], which implies that a use case shall be enacted before another can start.

3 Functional description

3.1 Risk Management functionality

A risk is the (negative) effect of uncertainty on objectives, and the risk management process is the "*systematic application of management policies, procedures and practices to the activities of communicating, consulting, establishing the context, and identifying, analyzing, evaluating, treating, monitoring and reviewing risk*" [24]. Depending on were the focus of the uncertainty and the objectives is put, risk management may be applied at different levels: from an organization-wide perspective, to the risks related to the development process of a project, to the risks related to the features of a given system, product, or service.

Regarding the objectives, they can address different areas; in particular, from the perspective of GDPR, we will be dealing with privacy and data protection risks, where the objectives to be protected correspond to the freedoms and rights of the data subjects, according to the privacy of data protection principles of the respective normative framework. Given the interlinks of privacy and data protection to other aspects, such objectives may be expanded to address more general goals, such as the protection of the data subjects' financial, reputational or other interests.

Thus, the risk management functionality should mostly focus on the protection of the data subject rights, rather than on those of the organization. Moreover, even the organization which is running the risk management (or that on whose behalf it is being run) should be deemed as a potential source of risks to the data subject rights when carrying out the risk analysis. For instance, the fact that an organization amasses a large amount of personal data introduces a vulnerability due to the potential linkability of different datasets by the controller itself, which may pose risks if the controller exceeds from the initially specified purposes, e.g. the data subject being unfairly profiled. This approach complementary to but not the same as other risk management activities where the organization itself is the target to be protected, e.g. security risk management (to protect the information and communications assets of the organization), compliance risk management (to protect the organization from legal penalties, fines or liabilities), or reputational risk management (to protect the external value of the organizational image).[1] Notwithstanding, this does not preclude the organization from carrying out such other risk management activities, which, furthermore, may mutually benefit from the application of risk management methods to privacy and data protection. For instance, security measures are a prerequisite for appropriate privacy and data protection (as established by GDPR), a strategy to mitigate compliance risks may involve exhaustive data protection impact assessments, whose results may in turn be used to analyze the potential impact of a data breach in the organization reputation, etc. Anyway, this approach where an organization shall carry out risk management activities by putting on someone else's shoes is not alien to existent corporate practice, as it can

[1] As stated by WP29, "*the DPIA under the GDPR is a tool for managing risks to the rights of the data subjects, and thus takes their perspective, as is the case in certain fields (e.g. societal security). Conversely, risk management in other fields (e.g. information security) is focused on the organization.*"[14] This does not rule out that similar tools can be employed to address security and privacy risks. For instance, STRIDE and LINDDUN are similar but distinct methods for security and privacy threat analysis, respectively. Or, lists of security controls in ISO 27005 and NIST 800-53 appendixes F and G have their privacy counterparts following the same approach in ISO 27552 and NIST 800-53 appendix J. Even GDPR itself mentions both data protection and security risks. Nonetheless, in all those cases PDP risks still merit a separate treatment, as the assets and threat sources to privacy and data protection may be different from those in security.

be found in other areas ranging from occupational safety and health to environmental protection, more generally encompassed by the discipline of Impact Analysis.

It shall be noted that the role of risks in GDPR overflows the scope of Risk Management functionalities in PDP4E: some purely organizational risks which are not directly linked to engineer's activities are out of the scope of the project; likewise, some activities which are tightly linked to the execution of a DPIA and which are presented elsewhere as part of the risk management are addressed in PDP4E by other disciplines (e.g. an appropriate model of the processing activities and categories of personal data processed is a prerequisite for a DPIA, however it is produced by modelling tools rather than from risk management tools; or some requirements are better addressed from a goal-driven perspective than from a risk-oriented one).

Within the scope of the Risk Management activities in PDP4E, Figure 2 shows the different processes that may be involved from a high-level perspective.

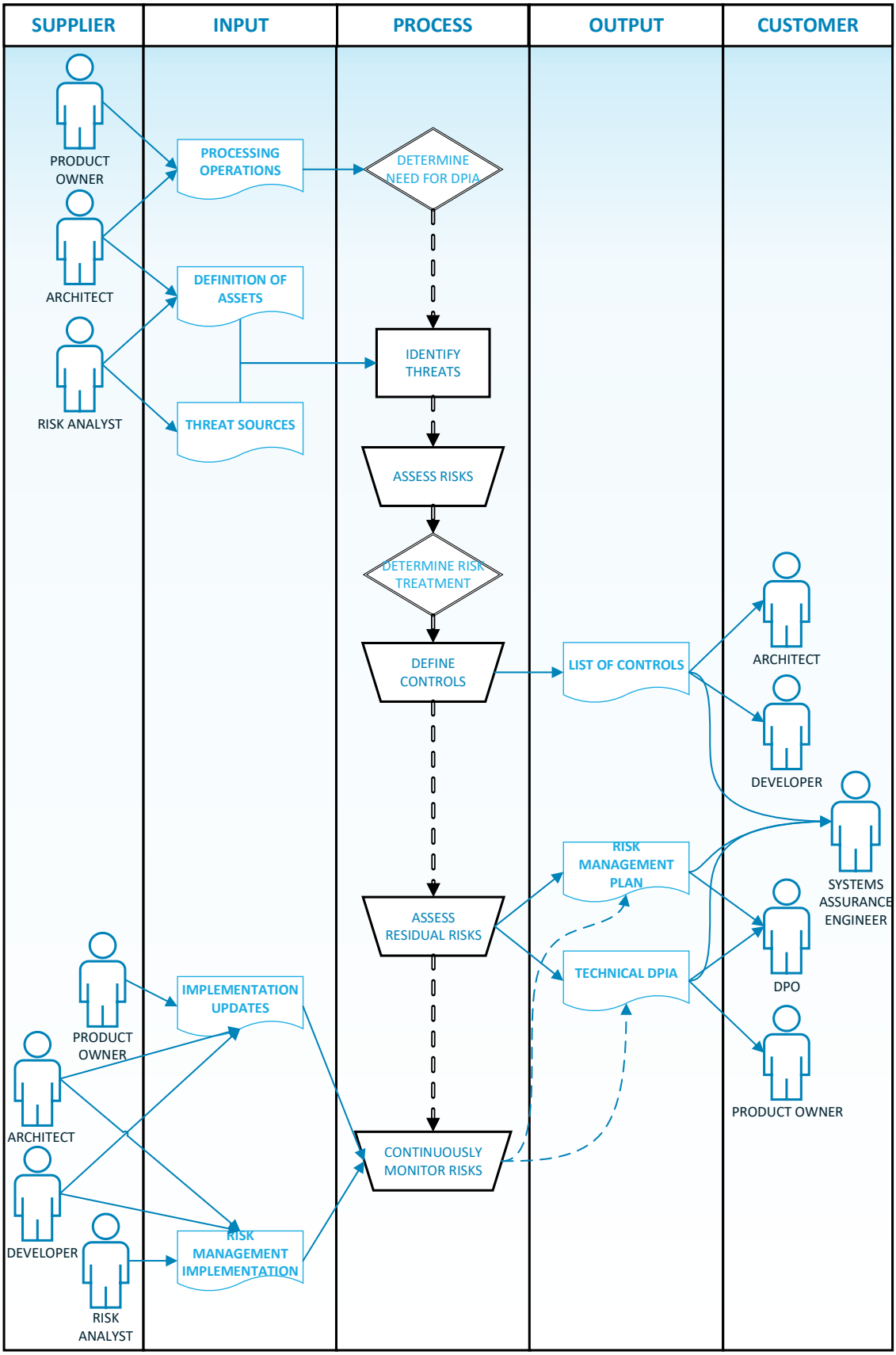

Figure 2. SIPOC diagram of the Risk Management functionality

0. An initial "step 0" deals with the *determination of the need for the Data Protection Impact Assessment*. GDPR explicitly mandates to carry out a DPIA when "*[a processing] is likely*

to result in a high risk to the rights and freedoms of natural persons". Before carrying out a DPIA, it is impossible to ascertain the risk level of a particular processing, but there are some kinds of processing activities which (as detailed by WP29 [14]) are more likely to result in such risks. In those cases, a DPIA shall be executed (regardless that it may be also useful elsewhere). An initial risk appraisal is carried out by examining whether the processing operations (i.e. "*taking into account the nature, scope, context and purposes of the processing*") fit into any of those high-risk categories.

1. Although the details of the risk management process will be provided in D3.1 and D3.4, we may advance that the bulk of the DPIA can be mapped to the activities of other Risk Management Frameworks [25][26][27] (given that GDPR itself does not mandate any specific procedure to carry out the DPIA). Thus, the first step consists in the *identification of threats*, which depends on the proper definition of the potentially threatened assets (specified as e.g. partial system models) and the potential sources and vulnerabilities causing those threats. It should be highlighted that these are not independent dimensions: the threats that may affect a given asset may well depend on the type of the asset itself.
2. Then, each of the corresponding *risks are assessed* and prioritized in terms of their consequences —known as impact— and likelihood (both which may in turn depend on other factors, including but not limited to the number of data subjects potentially affected).
3. Depending on the results of the risk assessment, different *treatments can be applied*. As the risk management process is carried out from a data subject's perspective, in practice, risks will not be transferred (i.e. insured) but almost always mitigated, and only seldom avoided (i.e. abandoning the processing activity) or assumed (only when they are low enough to be acceptable for the data subjects, not for the organization).
4. Those treatments materialize as *security and privacy controls*, whose specification is the main result of the risk management process, as they are the measures that will make risks decrease to an acceptable threshold. Note that controls are here just specified (i.e. defined as required), but they are to be implemented and/or enforced beyond the scope of risk management process itself.
5. After those security controls have been defined, *residual risks are assessed* which would remain once the controls are implemented (these residual risks should be below a given threshold of acceptability; otherwise, the supervisory authority shall be consulted according to GDPR Art. 36) and the overall results of the DPIA are reported (or more precisely, the results of the engineering-oriented Risk Management is incorporated to the overall DPIA which may address, as stated, other risk categories).
6. Risk management does not finish once the risk plan is issued, but it is subject to a *continuous monitoring* process whose results are to be reported as an update to the initial risk management plan, and which goes along the evolution of the system at hand, its behavior under operation, and the state of the art of threats and mitigation measures. Thus, it takes into account, among others:
    a. The progress in the implementation of the system, considering any new feature or functionality, plus, in particular, the security and privacy controls implemented.
    b. The security and privacy controls implemented by external chosen providers acting as data processors (which may be selected depending on such controls).
    c. Data breaches that may have occurred and their ex-post risk assessment.

Some open considerations regarding system requirements that shall be provided by the Risk Management method and tool specification include:

- Which system models (representing the assets) will be analyzed, what their format will be and which elements from the models will be taken into account for risk analysis.
- Which threat categories will be addressed (from among categories extracted from e.g. LINDDUN, STRIDE, non-compliance, project management risks, risks to the rights and freedoms of the data subject, etc.)
- How specific threats will be derived from the assets, and how to avoid that the instantiation of threats in a project results into an unmanageable explosion of their number.
- Which risk likelihood and impact estimators will be used to assess the risks, and how the risks will be prioritized.

3.2 Requirements Engineering functionality

According to the SWEBOK [28], a requirement is "*a property that must be exhibited by something in order to solve some problem in the real world*" and "*software requirements express the needs and constraints placed on a software product that contribute to the solution of some real-world problem.*" When such needs, constraints and properties constrain the solution rather than defining the functions that the system shall execute, we talk about 'non-functional requirements', which may refer to different categories. Privacy and data protection is a specific category of non-functional requirements which, thus, can be addressed through usual requirements engineering activities.

The GDPR provisions from which requirements will emanate are disseminated through the text of the regulation, but most of them appear contained by data processing principles (Chapter 2), rights of the data subjects (Chapter 3), and obligations of the controller and the processor (Chapter 4). The specific requirements to be applied may depend on the scope of the processing activities carried out by a data controller e.g. more stringent requirements apply when it deals with sensitive data, or with children's data, or when processing activities such as profiling are being carried out, or data is used for direct marketing purposes, or whether data is being transferred to third countries, or leased to processors, etc. Nonetheless, all those provisions are set in legal terms, and need to be translated into engineering terms and operationalized into proper requirements. Besides, all the requirements will always need to be instantiated for the specific project at hand, depending on the specific nature of processing activities, categories of personal data processed, purposes, lawfulness bases, etc. (e.g. if a processor should be selected depending on the measures it implements, the realization of the concepts of "processor" and "measures" will be specific to each given project).

Figure 3 shows the different processes that are involved in the scope of Requirements Engineering in PDP4E from a high-level perspective. It shall be noted, that in this case, the set of activities is not purely sequential, but there are several alternative paths and optional activities. These activities roughly follow the different Requirements Engineering Knowledge Areas established by SWEBOK [28], namely: elicitation, analysis, specification and validation (plus tracing, change management, and acceptance tests which appear within other knowledge areas).

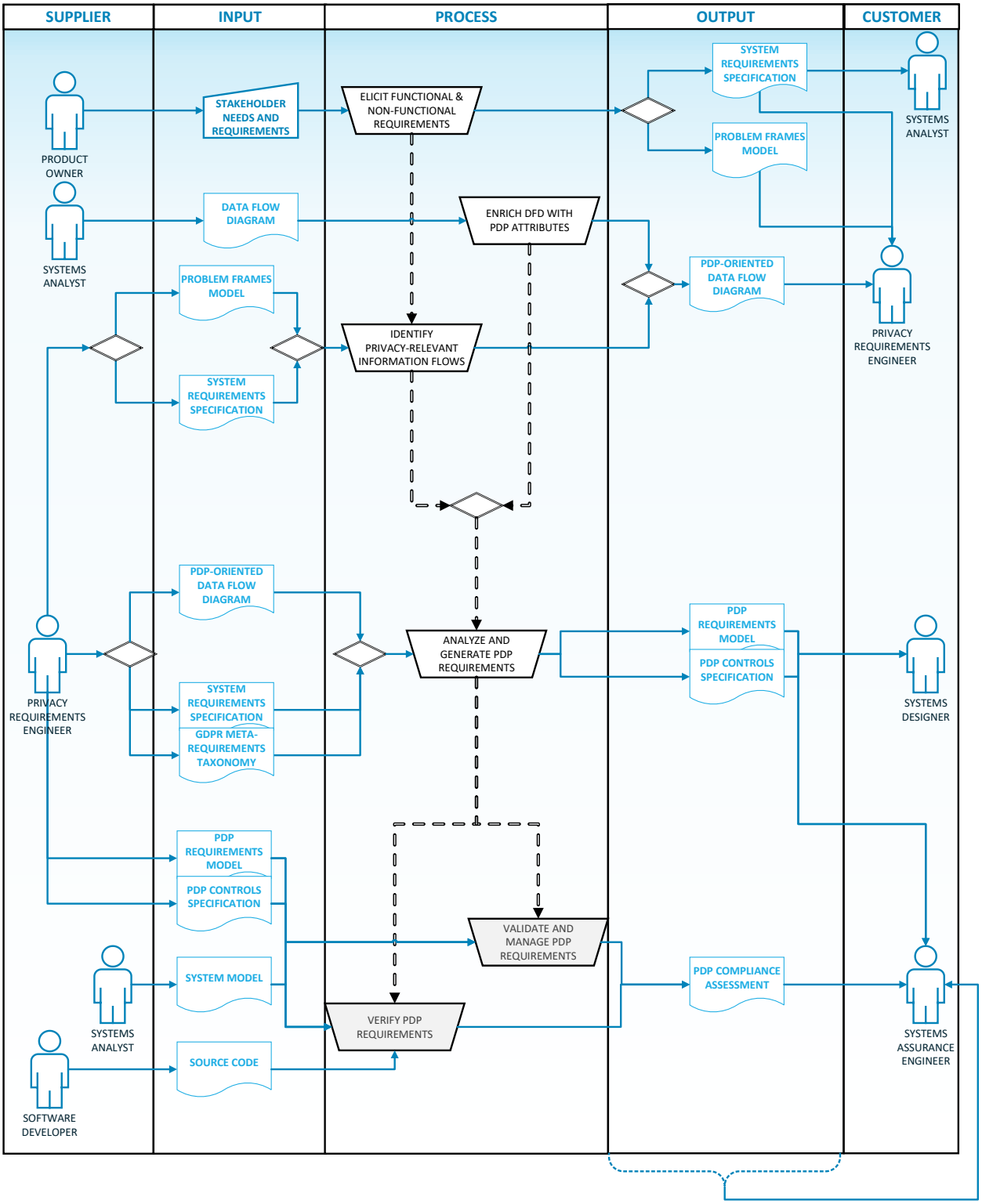

Figure 3. SIPOC diagram of the Requirements Engineering functionality

1. Privacy requirements, as non-functional requirements, are inherently relative: they can only be detailed in reference to other functional and non-functional features of the system at hand, which they implicitly qualify [29]. For instance, stating that "sensitive personal data shall be encrypted in transit" need to be detailed depending on the scope of 'sensitive personal data' and 'in transit' in relation to a specific system. Thus, *functional and other non-functional requirement categories shall be elicited* as the first step. The PDP4E privacy requirements methodology strongly emanates from the ProPAn method,

which is based on the "Problem Frames" requirements modelling paradigm; thus, a model according to that method (including different diagrams) may be created as a result of this step.

2. Once the relevant diagrams and models have been created, *privacy-relevant information flows are identified* (i.e. those which convey personal data, which transfer data between agents, etc.)
3. Problem-frames is not the only approach that can be used to elicit the privacy information flows. Alternatively, a Data-flow diagram may have been created following any other method whatsoever. This may be a more lightweight approach than sticking to the problem-frames approach; and it may be the only practical option if we are departing from an existing system whose dataflows have already been implemented. However, data-flow diagrams, in general, do not carry out the information that is needed for privacy and data protection requirements analysis (e.g. they even fall short of telling apart those flows which carry out personal data from those which don't). In any case, *privacy-enriched Data-Flow Diagrams* are key for our privacy and data protection requirements analysis, whether it is created as result of sticking to the problem-frames approach or created elsewhere.
4. Through a process of selection, refinement, and instantiation, *PDP requirements are analyzed and generated*. This activity may either depart from the above mentioned privacy-enriched Data-Flow Diagram (following the ProPAn approach) or from the system requirements together with a set of meta-requirements that capture GDPR provisions in engineering terms. In either case, project-independent requirements semantic templates are provided as an input, and instantiated by the process in the scope of the specific system. Using either approach (or both), project-specific requirements are generated, including the specification of privacy controls. To these privacy controls thus elicited, others will be added as a result of the risk management process. As with any other non-functional requirements, it may also be the case that conflicts or synergies arise among PDP requirements and between these and other categories, which forces to adjust some of them depending on their applicability.
5. Once requirements are defined, they are to be *validated and managed* afterwards. That is, requirements (including both PDP and others) should be checked e.g. against consistency criteria[2], and going along the development lifecycle, they are traced to design and implementation artefacts, and evolved according to the evolution of the system.
6. Finally, after the system (or a version) has been implemented, the fulfillment of *PDP requirements are verified***.** Different approaches may be applied (e.g. black-box testing, white-box testing, empirical user evaluation, etc.) depending on the level of verification to be addressed: in our case, we will apply formal source-code verification techniques that validate specific privacy and data protection properties (related to e.g. access control, compliance with purpose specification, etc.).

It should be noted that, from the perspective of the internal organization of PDP4E project, the two latest steps are not addressed by the WP in charge of Requirements Engineering activities; however, we have included them in this diagram in the sake of clarity and completion.

[2] Some acronyms such as SMART for 'Specific, Measurable, Achievable/Assignable, Relevant/Realistic and Time-bounded' or MECE for 'Mutually Exclusive – Collectively Exhaustive' try to reflect rules of thumb for consistency checks. However, in the case of PDP requirements, it is expected that these consistency checks mostly target the availability of personal data at different domains to carry out processing activities while guaranteeing protection goals.

Some open considerations regarding system requirements that shall be provided by the Requirements Engineering method and tool specification include:

- How to avoid the complexity of the problem frames approach currently proposed in the ProPAn method while maintaining its rigour.
- How to convert GDPR into operationalizable requirements, and how to manage the large quantity of requirements that may result from instantiating each of them depending on all the data categories, processing operations, etc. in a project.

3.3 Model-Driven Design functionality

According to the proponents of MDA (or Model Driven Architecture, itself one of the most known initiatives following the Model-Driven Engineering paradigm), this approach excels at providing value by [30]: using models as communication vehicles, deriving models and code from other models through automated transformation, enriching models, simulating and executing models, deriving information (e.g. documentation) from models, and guiding in the structuring of unstructured information. Many of these features indeed guide the use of MDE in PDP4E, as it is leveraged to:

- Annotate existent sources of data, both structured and unstructured with attributes related to personal and data protection aspects (the simplest example being the annotation of data fields and individual records as conveying 'personal data'). Although MDA does not provide a way to define such annotations, the software that scans the existent sources may leverage MDA models as the repository for the resulting enriched data.
- Analyze compliance of models with privacy and data protection principles, and apply strategies and tactics when possible to derive new models which are functionally equivalent yet privacy-enhanced.
- Derive rules to be satisfied by source code artifacts with relation to models that specify privacy and data protection attributes.
- Leverage the information of models to support system functionalities.

Although modelling activities themselves are not prescribed anywhere by GDPR, it is implicit that personal data categories, processing activities, lawfulness basis, processing purposes, etc. need to be modelled somehow so as to obey many of the requirements established by GDPR. As a matter of example, if we want to ensure that the collection of personal data is minimized according to the nature of the processing, such processing is limited to specified purposes, the disclosure of data is constrained based on need-to-know access control, etc.; then the engineer will first need to have a clear specification of what 'personal data', 'processing', 'purposes', or 'access' mean in the context of the specific system —for which modelling may provide support. All in all, modelling can be pivotal to implement the Privacy and Data Protection by Design paradigm.

Figure 4 shows the different activities that leverage the Model-Driven Engineering approach in PDP4E. Model-Driven Design activities in PDP4E do not follow a strictly sequential order: instead, there are different starting points from which the process can depart, and it can follow different paths (thus, the numbering below should be understood just as a hint). It should be noted that Requirements Management and Assurance activities in PDP4E also leverage the MDE philosophy; however, here we focus on those activities that mainly address the solution domain (while the former are targeting the problem domain, from either the specification or the validation side).

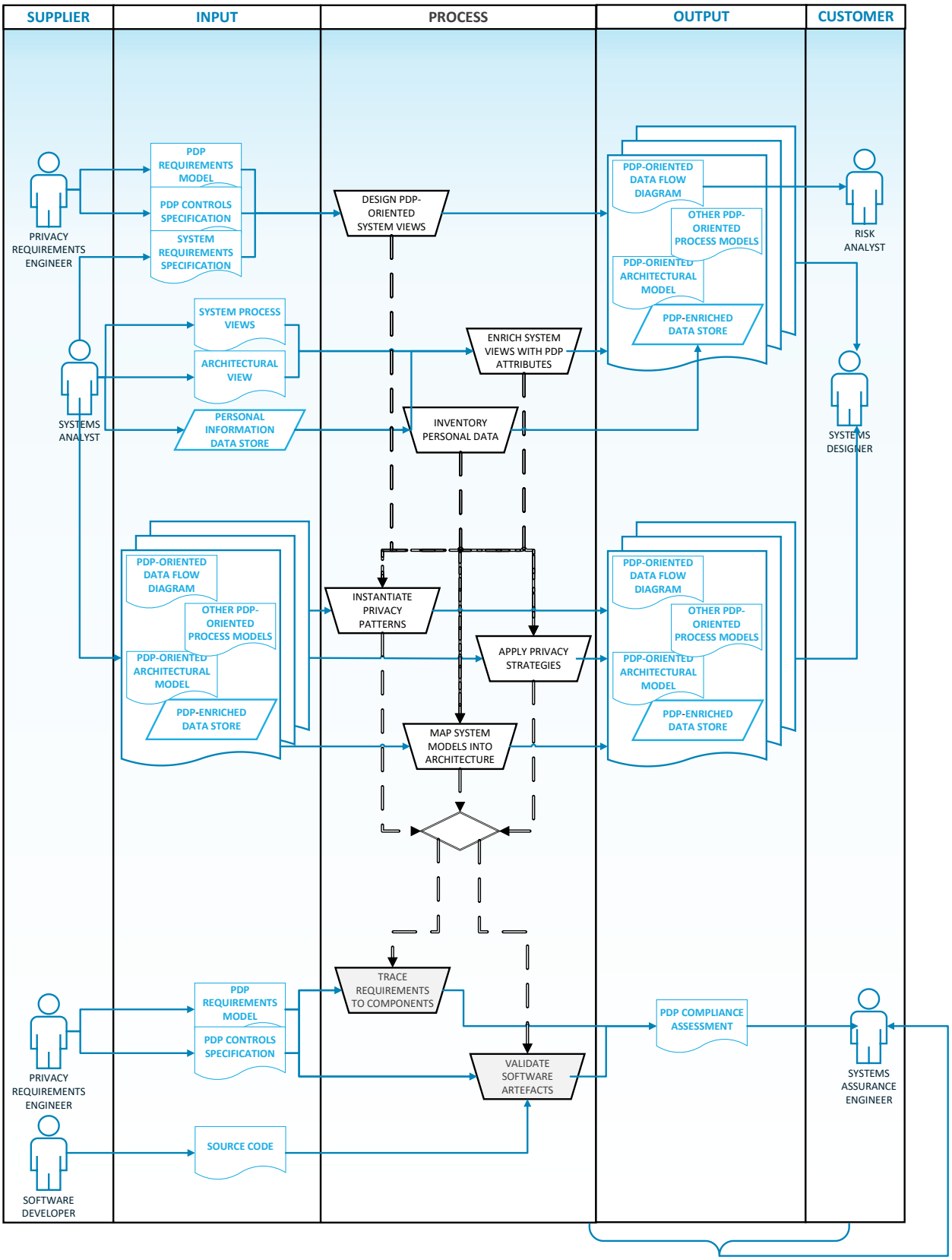

Figure 4. SIPOC Diagram of the Model-Driven Design functionality.

I. *Initial design modelling*, through different, parallel, supplementary alternatives:

I.A The process may depart from the requirements to produce a first version of the system design. More specifically, it may depart from: the system requirements

specification, from the specific privacy and data protection requirements, or from the specification of required privacy controls (as modelled in the Requirements Engineering activities). From those requirements, an initial design model is created by a system designer. This design may offer several perspectives, depending on the most salient viewpoints of the system at hand. But in the case of PDP4E, there are some specific models (PDP models in the following) which are more relevant than others and which are supported by our methods and tools.
As a result, models that represent the system from different perspectives can be generated, namely: a process model such as a Data-Flow Diagram (which are pivotal to the project overall, and especially for Risk Management), other process models (e.g. activity models, business process models), architectural models (focusing on components, connectors and deployment), and a structural model (which may well be the structural description of a data store e.g. a database schema). All those models are enriched with privacy and data protection relevant information, e.g. which fields of a database hold sensitive personal data, which data processor is in charge of storing that database, etc.

I.B As an alternative, the PDP design process may depart from an already existing initial design; or even from the design of an already existing system[3], which is to be enriched with the privacy and data protection attributes, to produce the same models just introduced above.

I.C Within that approach, a relevant perspective is that of the data inventory (or discovery, or mapping), which, departing from an existing datastore (including a schema with category data, plus possibly records with instance data), detects where personal data is stored and labels it accordingly (potentially combining automatic, and manual guided techniques); making existing (structured or unstructured) data emerge as personal data.

II. *Privacy and data protection improvement* of the design, where the system models (architectural, data, data flow, potentially other process models such as activity diagrams or business process models, etc.) are at least assessed, and ideally refined, extended or transformed to better abide by privacy and data protection by applying:

II.A Privacy patterns (reusable, technology-independent, well proven solutions, applicable in specific contexts.

II.B Privacy strategies and (distinct architectural goals, realized through different tactics).

II.C Allocation of system functions and privacy requirements into architectural components, taking into account the domain (e.g. a controller or a processor) to which they pertain.

III. *Model-driven validation*, including:

III.A Traceability from components to requirements, to ensure that all requirements have been addressed by the components that should include them.

III.B Validate compliance with privacy and data protection properties of software artifacts (in particular, annotated source code).

Some open considerations regarding system requirements that shall be provided by the Model-Driven Design method and tool specification include:

[3] According to the paradigm of Privacy by Design / Data Protection by Design, privacy and data protection should be considered since the onset of a project, rather than as an afterthought. Thus, it is discouraged that PDP considerations be delayed until a system design has already been created. Nonetheless, in reality, existing systems are often re-engineered when new regulations (such as GDPR) push systems improvements.

- Which system viewpoints to model, which metamodel to use, and how this metamodel should be extended with PDP attributes.
- How to ascertain which of the data in a system is personal data, which of the functions are data processing operations, etc.
- Which strategies, tactics and patterns implement, how to select them and how to implement them in a model-driven fashion (e.g. as model transforms).
- How to integrate control definitions into design.

3.4 Systems assurance functionality

System assurance activities aim at providing justifications to trusting that a system is working according to the specifications; in our case, according to GDPR and other PDP standards. Assurance is not focused itself on showing specific features (functional or non-functional), but on *demonstrating that there are evidences* which support that both the product and the process during which it was created stick to the respective specifications.

Figure 4 shows the most relevant activities in the assurance process for PDP4E.

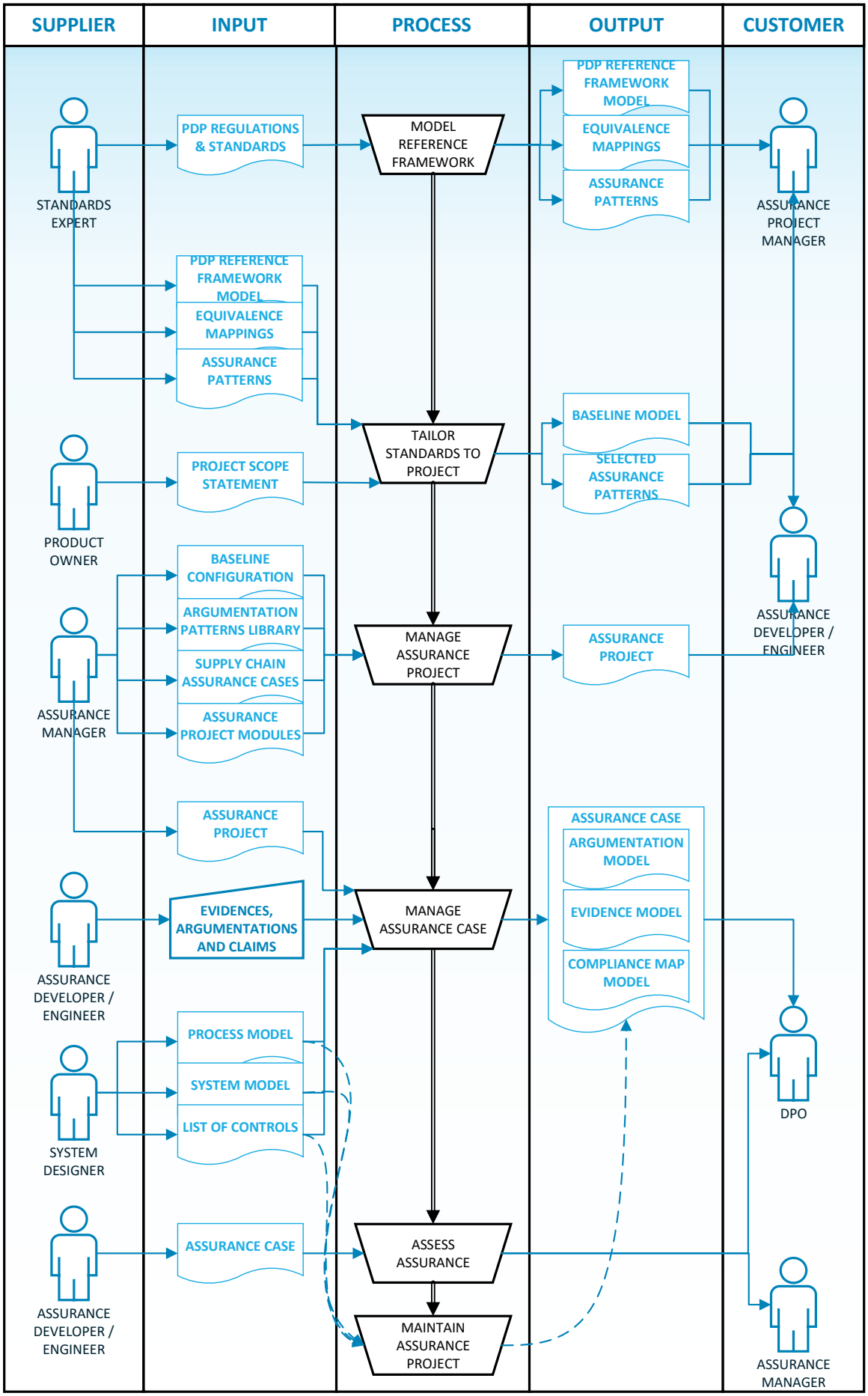

Figure 5. SIPOC Diagram of the Systems Assurance activities.

1. The first step to ensure compliance with a given PDP normative framework is to *model such reference framework* using the language (metamodel and visual notation) of the assurance tools. That reference framework may involve not only a model of GDPR (which will be provided as an external resource to the organizations carrying out the assurance process), but also one of other sector-, industry-, or organization- specific standards and regulations. The reference framework is modelled in two ways: a modelling of the process plus a set of argumentation patterns. Besides, if more than one standard is modelled, equivalence mappings between one another can be also output.
2. Reference *standards and regulations may need to be tailored depending of the scope* of a specific project. It is not that an organization may cherry-pick which parts of a regulation they wish to abide by, but some of those parts need not be applied because they refer to aspects out of the scope of this project (e.g. requirements applicable to profiling, international data transfers, etc. do not hold if such activities are not going to happen). When the assurance process is taking place on a new system, this tailoring is based on an initial project scope definition; however, if assurance is applied during reengineering, other inputs may be added.
3. For each development project, a parallel *assurance project shall be managed*, linked to a baseline (tailored standards), chosen argumentation patterns, and, if applicable, imported assurance cases generated by third parties, and the definition of assurance modules (to create an assurance case from several parts which map to e.g. project subsystems).
4. An *assurance case is created* and managed by managing evidences, developing claims and argumentations and mapping them to the evidences.
5. The *quality of the evidences* is assessed in relation to the system models to which they are traced.
6. When the project generates new evidences, the *assurance case is updated*, and any checks are reapplied.

Some open considerations regarding system requirements that shall be provided by the Systems Assurance method and tool specification include:

- How to model a legal text which is not structured as a process, such as GDPR (and its related guidance), as an assurance reference framework; and how to model the forces that and opening clauses leave room for variability in the application of GDPR.
- How to address processing operations, which are not required but subject to the reference framework, and whose evidences are not generated during systems development but during operation.
- How to ensure traceability and quality of evidences to the reference framework.
- How to model controls to be used in argumentations. In other words, how to integrate the definition and creation of control-specific argumentation patterns.

4 Cross-discipline abstract use cases

4.1 General description of common actors and use cases

In the previous section, we have gone through the functions to be provided by the tools that support the methods introduced in each of the different disciplines. Here, we will abstract the common methodological features of the different functionalities, into a set of common actors and use cases which share similar traits, even if they are applied to completely different disciplines.

These use cases go beyond pure system requirements and add a layer that provide hints on how the PDP4E toolset might be linked altogether and introduced in development cycle engineering. Due to the heterogeneity in the current level of detail among the results of the different disciplines, not all the use cases can be directly linked to contents of all the disciplines. In any case, these just show a potential application approach, which will depend on the specific scenario. In particular, the application demonstration pilots to be developed may respond to a different degree to these use cases.

Actors are stereotyped as either <<project>> or <<organization>> actors, depending on whether they hold a project-specific position, or a traversal role in the organization. For instance, a project manager or a software architect will be *the manager of this project*, or *the architect of this product*; while a Project Management Officer, a Chief Technology Officer, a Chief Information Security Officer, or a Data Protection Officer will respectively be the PMO, CTO, CISO, or DPO *of the company*. That said, we consider that project roles are not exclusive, e.g. a single project manager may be in charge of managing several projects, but we can still ask about the project manager of a given project. Likewise, we admit that organizational roles may be distributed among the members of a team, e.g. the DPO role in a large organization may be exerted by different individuals who split the work, as signalled by WP29 guidance [14].

In particular, we consider the following abstract **actors, responsibilities and skills**, not linked to a given specific discipline:

- ***<<organization>> Privacy Method Engineer*** is in charge on providing the global organization privacy and data protection knowledge, in the form of knowledge bases, (e.g. risk estimators, patterns, etc.) and methodologies. Of course, they will depart from the common privacy and data protection practice (e.g. the methods and knowledge bases provided by PDP4E), but they may also need to create methods appropriate to specific industries (e.g. codes of practice, pattern catalogues, assurance patterns), or tailor existent methods to the needs of a given organization (by filtering the relevant parts). They are an expert in the three of privacy and data protection, a given engineering discipline (risks, requirements, design, or assurance; depending on their specific role), and model and method engineering; plus they are familiar with the realm of the systems produced by the organization and their industry.
- ***<<project>> Discipline Manager*** oversees how a discipline is handled in a specific project; translates privacy and data protection knowledge, already provided in a close form, into the project models; and assesses (from the perspective of their discipline) whether a system meets the required privacy features (as described in the sources they are provided with). For instance, a project architect, given an existent definition of likely high-risk data processing activities, together with guidance to identify them, and potentially supported by a tool, will be able to locate risky activities in a given system, so that a DPIA can be carried out on them. They are experts on the discipline (in this case, risk management), and have a general understanding of privacy and data protection: they do not develop

new privacy and data protection knowledge, but they rely on existent knowledge to particularize it to a specific project.

- ***<<project>> Discipline Engineer*** implements the technical work of a discipline, under the supervision of the Discipline Manager, and always guided by methods and tools that encapsulate the Privacy and Data Protection knowledge. With respect to privacy and data protection, they always rely on existent knowledge encapsulated into methods and tools they are users of. For instance, a software designer may use a modelling tool to introduce a given pattern in their design, leveraging the pattern description, an example diagram, and potentially automated evaluation of the context of application.
- ***<<organization>> Data Privacy Officer*** (as defined in GDPR Art. 37) is in charge of informing, advising and monitoring all the privacy and data protection engineering use cases. This role has a high savvy on privacy and data protection, but not necessarily on engineering activities: they do not directly carry out any engineering activities, but they provide a supporting function, so as to ensure that the different engineering activities effectively align to the privacy and data protection framework. A role with similar, advisory functions and skills may exist even when a DPO is not legally mandated (as per GDPR Art. 37, Member States law, and according to WP29 guidance [14]).

These abstract actors will be realized as different concrete actors in the respective disciplines as presented in D3.1-D6.1 [4][6][8][10] (e.g. Risk Analyst, Privacy Design Engineer, Requirements Engineer, Development Customizer, Data Analyst Engineer, Design Engineer, Standards Expert, Assurance Manager, Assurance Engineer).

Regarding the use cases, they form all together an overall methodological cycle (see Figure 6) which can be partly mapped to a V-shaped structure (where the specification to the left is mirrored by validation to the right), which can be summarized as follows:

1. **Model PDP specific framework.** A privacy and data protection framework which an organization is aiming to stick to is modelled. This PDP framework includes all the external resources that are reused as such by the organization, be it as external requirements mandated by regulations or standards (parts of this framework such as the GDPR itself, will be indeed common to any organization), interpretations specific to a given industry or sector, best practices, etc.
2. **Tailor PDP framework to a project.** Before a development project is started, the framework is tailored, depending on its applicability to the context of the project (e.g. if a project does not include profiling activities, then any elements of the framework dealing with profiling are omitted).
3. **Model system.** The system is modelled from different perspectives which can be necessary for PDP activities. This is partly external to the PDP cycle, as modelling may have been carried out elsewhere for other purposes; and partly internal, as it still needs to account for privacy-relevant attributes.
4. **Instantiate PDP framework.** The framework is instantiated according to the specifics of the project (e.g. apply protection measures to sensitive data is particularized to the specific elements designed by ‘protection measures’ and ‘sensitive data’ in the context of the project).
5. **Elicit new system models.** New system models are elicited, by transforming, refining, improving, or enriching the original model so that the system appropriately deals with PDP properties.
6. **Validate and monitor continuously.** The compliance with the original PDP framework is assessed, ideally not once but continuously.

7. **Assess compliance of supply chain.** This compliance assessment includes the assessment of subsystems provided by external parties.

As above said, these use cases represent a common abstraction to the different disciplines: each discipline will have their own use cases, some of which represent specializations of the former (as explained in the "Related Use Cases" section of the descriptions below), and some others will be specific to each discipline without a generic counterpart. Details of the discipline-specific use cases can be found in D3.1 regarding Risk Management [4], D4.1 re. Requirements Engineering [6], D5.1 re. Model-Driven Design [8], and D6.1 re. Assurance [10].

It should be noted that the numbering of these uses cases is a mere indication that reflects the precedence relationships (see Figure 6 below). However, they do not represent strictly sequential steps (as long as the precedence is respected), and different disciplines may be running different use cases at the same time.

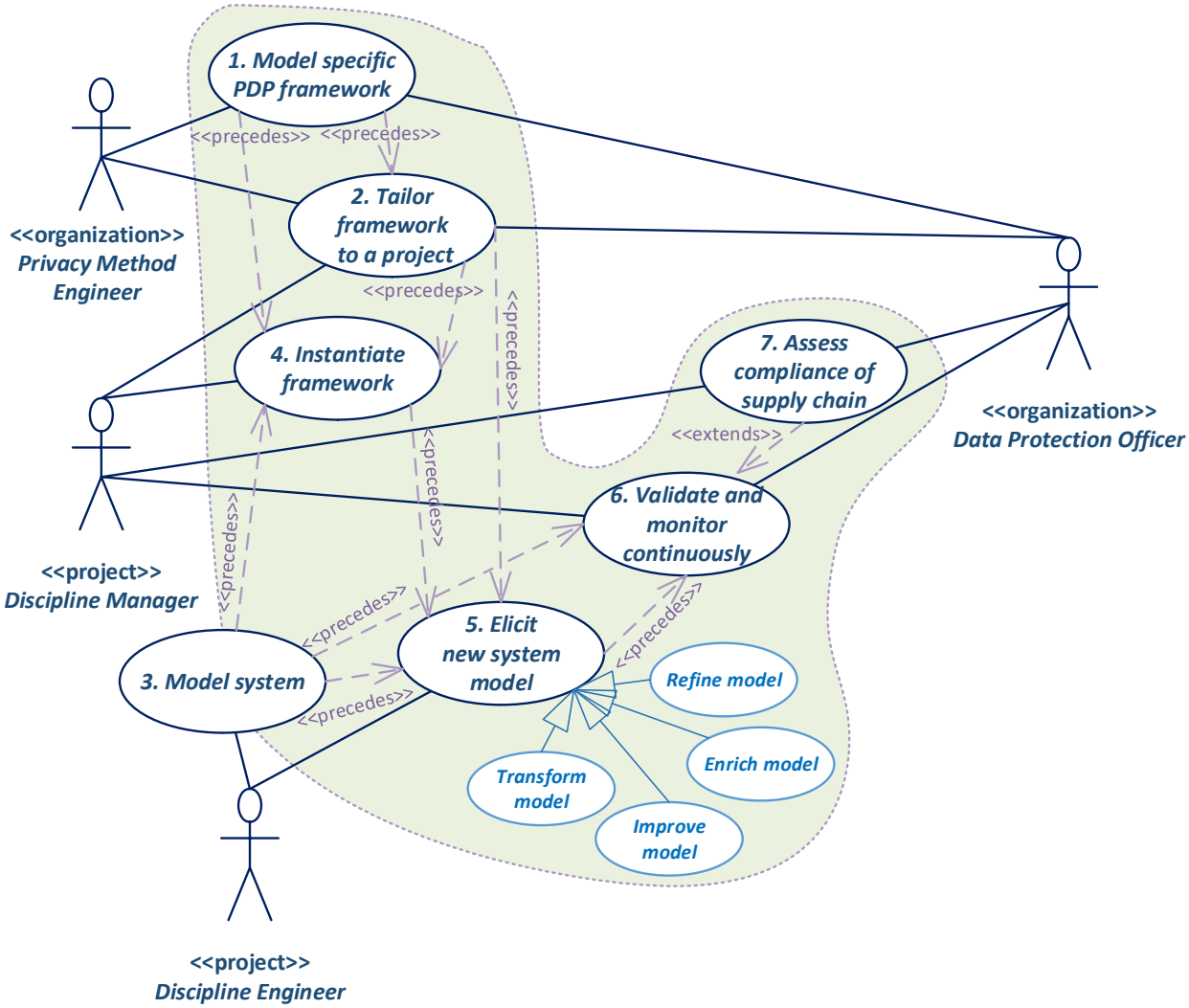

Figure 6. Diagram of abstract use cases.

4.2 Individual description of the abstract use cases

4.2.1 Use case 1: Model specific privacy and data protection framework

UC1 - Model specific privacy and data protection framework	
Purpose	Provide an organization's project staff with a model of all the applicable normative sources (which supplement GDPR).
Actors / Stakeholders	Privacy Method Engineer, DPO
Trigger	An organization (individually or in association with others) decides to apply a specific privacy and data protection framework in its processes. Or an existent privacy and data protection framework, which has already been modelled, is supplemented with updated or new stipulations that shall be included in the updated model.
Preconditions	GDPR constitutes a general, compulsory, privacy and data protection legal framework that all the organizations shall abide by; hence a model of GDPR (e.g. the one provided by PDP4E) is assumed to be available as an input resource to the method.
Assumptions	N/A
Post-conditions	A model of the applicable normative framework is provided to the organization's project staff.

Functionality description:

(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)

In this use case, a Process Engineer models the privacy and data protection framework that is applicable to a given specific sector, industry or organization.

Besides GDPR itself, other privacy and data protection normative and regulatory sources may be applicable under specific circumstances. Thus, *models for all those other sources that supplement GDPR shall have been defined before a development project is started*. These may include:

- Other legal instruments: national derogations (despite being directly enforceable in all Member States, GDPR itself has some opening clauses which allow for some aspects to be regulated at national level[4]) and other related regulations and directives, be them generic (e.g. ePrivacy, NIS, etc.) or sector-specific (e.g. for smart grids, connected vehicles), which may impact aspects of privacy and data protection.
- Quasi-regulations: CJEU rulings, together with guidance from WP29 and EDPB (despite the latter not being legally binding) supplement GDPR with hermeneutics that help interpret it. Such interpretation is inherently evolving as new documents are issued by those institutions. (Although the scope of application of some of these instruments

[4] A detailed discussion of the opening clauses of GDPR can be found (in German) at Kühling, J., Martini, M., & Heberlein, J. (2016). *Die Datenschutz-Grundverordnung und das nationale Recht: erste Überlegungen zum innerstaatlichen Regelungsbedarf*. Monsenstein und Vannerdat. http://www.foev-speyer.de/files/de/downloads/Kuehling_Martini_et_al_Die_DSGVO_und_das_nationale_Recht_2016.pdf . An graphical summary represented as a mind-map, is available in English at https://www.flickr.com/photos/winfried-veil/29706462112/

	may be quite generic, these resources still entail modelling after the methodology has been specified.) - Co-regulations: codes of conduct (applicable to e.g. specific industries), codes of practice, and voluntary certification schemes extend and refine the contents of GDPR with further stipulations that may address e.g. some concrete techniques to be applied to comply with specific obligations. - Self-regulations: binding corporate policies (which address conditions of international transfers that guarantee protections equivalent to those of GDPR), as well as in-house conventions, may supplement GDPR with organization-specific, internally enforceable obligations as well as voluntary conventions or best practices captured. - Technical standards (e.g. ISO standards, be them global or industry-specific) may help implementing GDPR by providing the operationalization in technical terms of different contents of GDPR (e.g. ISO/IEC 29134 details how a PIA is carried out, ISO/IEC 29100 details privacy principles into technical requirements, ISO/IEC 27552 will define privacy controls, ISO/IEC 27550 defines privacy engineering processes) The privacy and data protection framework model may address several aspects in different dimensions [31]: - From a theoretical perspective, it can address ontological aspects (what the concept of privacy and data protection to be considered, and possibly other related concepts concepts), deontological (what are the mandated contents required by the framework), situational (how the framework is tailored to e.g. specific domains or contexts) and epistemological (what is widely known as e.g. best practice). - From a method engineering perspective, it can address both product-oriented aspects (what is the result of the method e.g. system quality attributes) and others related to development process lifecycle (how the method is carried out e.g. development tasks or activities). Thus, a privacy protection framework may be modelled into different types of knowledge bases, depending on the discipline whose perspective is used, namely threat bases, risk estimators, meta-requirements, design patterns, assurance reference framework, argumentation patterns, etc.
Related use cases	- This use case shall *precede* any other use case (the privacy framework a project is going to abide by shall be known before the project starts); in particular, the modelling of the privacy framework shall precede the tailoring and the instantiation of such framework. - Although the Risk Management discipline does not explicitly list any use case implementing this one, its method consumes a knowledge base of privacy threats (to be avoided) and controls (to be implemented) that constitute part of such framework, and which implicitly means they need to have been created beforehand. - Requirements Engineering method defines project-independent meta requirements which capture the contents of product-oriented requirements derived from GDPR, standards such as ISO 29100 and privacy properties (as defined by Hansen [32]). The Requirements Engineering use case Update meta-requirements allows adding requirements from an extended data protection framework.

	- Model-Driven Design does not explicitly provide for this use case, yet it departs from an implicit knowledge base of strategies and patterns (considered as a prerequisite to the method). - Assurance method defines a project-independent normative framework which captures process-oriented requirements from GDPR, its interpretation through WP29 and EDPB guidance, etc.; as well as assurance patterns which capture best practice for assurance, and compliance mappings to standards such as ISO 29134. This is addressed by the use case Capture Information from Standards, Define Equivalence Mapping, and Define Argument Pattern. For each discipline, other specific framework parts (e.g. additional standards or legal rules) can be defined and added by this U.C. at that stage.
Relation to GDPR	GDPR provides for the development of these ancillary privacy and data protection framework resources: codes of conduct (Art. 40, Art. 41) and certifications (Art. 42) which can be applied to several areas of data protection (Art. 24.3, Art. 25.3, Art. 28.5, Art. 32.3, Art. 46.2.f), binding corporate rules (Art. 47) regarding international data transfers; EDPB guidelines, recommendations and best practices which cover several areas (Art.70.1.d-m, Rec. 136); national supervisory authorities tasks (Art. 57) and Member State derogations (through different articles[5]).
Frequency of use	A few times per organization. Larger organizations working on several industries may need to instantiate this use case more often. Small organizations may not need to instantiate it ever: even if sector-specific rules are applicable, the modelling of the privacy and data protection framework may have already been carried out by other organizations (e.g. trade organizations, external consultancy firms, etc.)

4.2.2 Use case 2: Tailor framework applicable to a project

UC2 -Tailor framework applicable to a project	
Purpose	Determine the specific contents of the privacy and data protection framework that are applicable to a given development project.
Actors / Stakeholders	Privacy Method Engineer, Discipline Manager, DPO
Trigger	The lifecycle of a new development project is initiated.
Preconditions	General (i.e. GDPR) and specific normative framework have been modelled.
Assumptions	N/A
Post-conditions	A PDP framework baseline applicable to the specific project is available.

[5] Kühling et al., op. cit.

Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	
	The project management staff shall determine which parts of the regulatory framework apply to a given project. With that aim, they shall: 1. Select the data protection framework applicable to the project (depending on, e.g. the industry, organization, country, etc.) 2. Select the clauses of the framework that are in the scope of the role that the organization is playing with respect to this project. For each variability point in the framework, select the specific clauses which are applicable, by instantiating the values of the applicability criteria that hold in this project. The privacy and data protection framework may leave room to variability under different considerations (see below for some GDPR variability points, they may also exist within other specific regulations), depending on the role of the organization (e.g. data controller, processor), their size, the processing activities it undertakes, etc. In general, only some of the contents of the framework apply to a given organization or project (e.g. a code of conduct of an unrelated industry shall not be considered, nor national implementations of other Member States apply, etc.).
Related use cases	- The tailoring of a given privacy and data protection framework logically *follows* the modelling of such framework. - The tailoring of a given privacy and data protection framework *precedes* the instantiation of such framework. - Risk Management method includes the pruning of threats from the knowledge base, depending on project contents; this is defined as part of step 3 of LINDDUN, but may be carried out independently and earlier in the process. - Requirements Engineering method may include requirements which are only applicable in some given contexts (and which are automatically pruned as part of the use case Generate Requirements or the method step Generate Requirement Candidates, when the meta-requirements are instantiated). - Model-Driven Design method introduces the selection of design strategy and the application of privacy patterns, each of which are only applicable under some given contexts. For now, this selection is manual (thus it does not appear as a detached use case in this discipline). - Assurance method may include argumentation patterns which are only applicable under specific circumstances, roles which do not always exist (e.g. DPO), and activities which are not always required. This is addressed by the use case Define Assurance Project Baseline.
Relation to GDPR	According to GDPR (only a few of the variability points are herein listed[6]):

[6] A detailed list of variability points in GDPR can be found by browsing this tool https://privacypatterns.cs.ru.nl/tool/

	- A Data Protection Impact Assessment (Art. 35, WP29 guidance) is only required under some circumstances (although it may be advisable anytime whatsoever) which are deemed to represent a high risk to data subject, e.g. the organization is carrying out profiling activities, systematic monitoring of data subjects, it is processing sensitive personal data, etc. - Small organizations are not required in general to keep records of processing activities (Art. 30.5) - Industries which are sworn to professional secrecy (Art. 90) may limit their cooperation with supervisory authorities in what involves data covered by that obligation of secrecy. - Etc. Besides, the scope of most GDPR clauses is limited to organizations playing different roles. For instance: - If an organization (in a specific project) will only be carrying out processing activities on behalf of others, the data processors' obligations (Art. 28, Art. 32, etc.) apply, but not those exclusive of data controllers' (Art. 24, etc.). - If an organization (in a specific project) will not be making automated decisions, then the related data subject rights (Art. 22) are not relevant. Opening clauses (see note 4 above) leave room to Member States to introduce some specific regulations or derogations in specific scopes shall be also considered.
Frequency of use	At most a few times per project: every time a development project is initiated, the project management staff will configure the requirements that are applicable to that project. Sometimes, not all the applicability criteria may be determined at the inception of a project (e.g. not all the processing activities are determined beforehand): in that case, this use case will need further iterated once the development is more mature. As it may be the case that the same requirements are common to several projects of the same type, it may also happen that this use case is only instantiated once for several projects (especially if the organization usually undertakes similar projects).

4.2.3 Use case 3: Model system

Model system	
Purpose	Provide a systematic view of those system aspects which are relevant for privacy and data protection, with well-defined semantics that support the coming PDP activities, by offering both a common, shared understanding for humans to communicate and a formalized

	representation which can be used by software tools to depict, analyze or transform it.
Actors / Stakeholders	Discipline Engineer
Trigger	A SDLC process may include modelling activities at different stages, typically related to different disciplines (e.g. requirements may be modelled at some point different from that of architecture). If the SDLC process is iterative, each modelling activity will likewise take place several times, as the project iterations ensue. The modelling activities are defined by the SDLC itself rather than by PDP4E: PDP4E methods need to embedded in the SDLC and may introduce constraints in modelling, but they do not tell the SDLC on where modelling should take place. Thus, it is such progress of the SDLC which acts as the external trigger for the respective modelling activities (depending on the respective process the SDLC abides by)
Preconditions	N/A
Assumptions	The SDLC process followed in a project includes modelling activities, according to the modelling languages employed by PDP4E.
Post-conditions	System models have been created from different perspectives, as appropriate to address the different disciplines dealt with by PDP4E.
Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	Modelling activities can follow a number of techniques, depending on, among other aspects, the viewpoint to be modelled, the domain of the system, and the SDLC process to be followed. As a matter of example, a structural domain model can be built by applying textual analysis techniques to a given specification, combined with pattern selection methods. PDP4E is in principle agnostic regarding the specific techniques to be used to elicit each of the system models used by our method; nonetheless, in practice the modelling approaches are heavily dependent on the features supported by the PDP4E background tools. Normally different model views of the same system are created which address the interests of different stakeholders: each model view (usually referred to as 'model' for short) represents the system from a given viewpoint, by abstracting those system aspects which are relevant to address the respective concerns. In the case of PDP4E, each discipline will require their own models (although a single discipline may require more than one model and, conversely, the same model can be reused by several disciplines) to later carry out their analysis and transformations. Model views also include any constraints which apply to the system (from the respective viewpoint) and describe all the concepts which are specific to the said model view.
Related use cases	- System modelling activities shall *precede* the PDP4E activities which act on the generated models to transform, enrich, or refine them. - System modelling activities may *precede* the framework instantiation according to the specifics of a project endeavor. It is possible, but difficult to instantiate a given framework without a partial yet clearly defined system model. For instance, if an

	(abstract or meta) requirement needs to be applied to all the 'processing operations' whose 'purpose' is that of 'user profiling', it is difficult to determine its specific reach in a given project without a model of the processing operations and their purposes. - Model-Driven Design method has modelling at its core (as its name indicates), and includes three specializations of this use case dealing with different models: Design Data Elements, Design Process Elements. And Design Architecture Elements. - Risk Management requires that these system models have been defined, as a prerequisite for the Import System Data Use case. In particular, the Risk Management method depends on architecture and DFD models to model the assets subject to risks. - Requirements Engineering method includes three specializations of this use case which introduce a model view of the system requirements: Add functional requirements, Add non-functional requirements, and Add specific information dedicated to the GDPR. Plus, the method specifies two other system viewpoints (not all of them always enacted): problem frames model, and data-flow diagram (which provides a functional system view, to be created outside the Requirements Engineering method). - Assurance method entails the definition of a process model that captures the specific process followed by an organization in a project (and which must align to the reference framework), an evidence model (which captures the instances of the artefacts), and an argumentation model (which provides structured claims of compliance). This is addressed by the use cases Develop claims and links to evidence, Characterize artefact and Specify Artefact Lifecycle.
Relation to GDPR	GDPR does not prescribe the use of any concrete engineering techniques, neither does it with regards to modelling specifically. However, modelling may prove pivotal in the support of different parts of GDPR: - GDPR does demand that controllers specify (e.g. as per Art. 30, Art. 35, etc.) which are the personal data processing operations they carry out, the categories of personal data they deal with, the concerned data subjects, the protection measures taken, etc. All these can be captured as attributes in a model. - According to the accountability principle, the controller (and processor, if applicable) shall be able to demonstrate compliance, for which models can act as appropriate evidence, as they offer the dual role of being useful for human communication and appropriate for automated analysis. - Besides, these models are key to support data subjects rights specified in Chapter 3: e.g. it is difficult to provide the data subjects with personal information stored about them if it cannot be easily gathered, which requires, at a minimum, that all personal information related to that data subject is appropriately

	'labelled' as personal information and linkable from the data subject's identity.
Frequency of use	This use case is enacted whenever the SDLC abided by this project involves modelling activities. This will range from at least once per modelling perspective to once per iteration.

4.2.4 Use case 4: Instantiate framework

Instantiate framework	
Purpose	Enact the privacy and data protection activities applicable to a given project, considering the specifics of that project.[7]
Actors / Stakeholders	Discipline Manager
Trigger	The activities from a given discipline start within the SDLC (Systems Development Lifecycle) of a given project. In that context, the PDP activities need to be parameterized to the specifics of the project.
Preconditions	The privacy and data protection framework to be applied by an organization to a project is available. An initial system model is available.
Assumptions	N/A
Post-conditions	For each discipline, the project development staff counts with a knowledge base instantiated and particularized to the specifics of this project.
Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	
	Given a privacy framework, tailored to the scope of a project, all the references to placeholder concepts are instantiated as many times as necessary, and replaced with the actual values of the element in the given project. For instance, if an abstract requirement, pattern, etc. refers to 'sensitive personal data', it is instantiated by generating as many concrete requirements as categories of sensitive personal data actually exist in the system developed by the project.
Related use cases	- The instantiation of the PDP framework necessarily *follows* the modelling of that framework, and the tailoring of the framework to the project. - Besides, in order to instantiate a PDP framework according to the specifics of a given project, it is recommended that the system at hand had been modelled, at least partially from the perspective of the given discipline. - Risk Management method includes, as part of the Vulnerability Analysis and Threat Analysis use cases, the population of threats

[7] The terms 'instantiation' is employed here in a narrow sense: given the privacy framework (e.g. meta-requirements, threat patterns, design patterns, assurance patterns) created by PDP4E or extended from it by an organization as a project-independent resource, it acts as a template which needs to be populated with the specific contents of a given project (e.g. what personal data, processing activities, data protection measures etc. are included) to make specific instances (e.g. requirements, risks, data structures, argumentations) We do not imply that PDP4E toolbox will allow a straightforward application irrespective of the regulation, domain and use case.

	extracted from the knowledge base (LINDDUN step 2), which are instantiated depending on the vulnerabilities applicable to each data flow element, and pruned depending on system trustworthiness assumptions (LINDDUN step 3). - Requirements Engineering includes the use case Generate Requirements (corresponding to the method steps Generation of Privacy Requirements Candidates and Adjust Privacy Requirements), where meta-requirements are instantiated and parameterized with project elements. - Model-Driven Design method includes this use case subsumed within the implementation of selected privacy strategies and patterns. - Assurance method includes the definition of compliance mappings, which maps evidences generated by a given project to reference artefacts defined in the framework, and instantiation of argumentation patterns. This is addressed by the use cases Define compliance means and Apply an argument pattern.
Relation to GDPR	GDPR defines a set of concepts in article 4 such as personal data, processing, controller, processor, consent, etc. But the concepts need to be mapped to concrete instances in any given project (which will have its own 'personal data', 'processing' activities, etc.) This is pivotal for GDPR compliance, as all these concepts are then extensively used throughout the rest of GDPR.
Frequency of use	Once for each activity in a discipline method that is parameterized depending on the instances of privacy concepts. Plus, once every time the parameter values changes (i.e. the system is updated).

4.2.5 Use case 5: Elicit new system model

Elicit new system model	
Purpose	Derive a system model from another, previously existing, which caters for PDP considerations.
Actors / Stakeholders	Discipline Engineer
Trigger	The elicitation of new systems models comes once the original system model has been created, but before taking it as valid and moving to a further step in the SDLC.
Preconditions	A system model is provided as an input.
Assumptions	N/A
Post-conditions	A different system model is generated as an output. The new model may provide a more accurate description of PDP aspects in the system, a different viewpoint, or even the model of an alternative implementation of the system which is better from a PDP perspective.

Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	
	Departing from an existing system model, a new model is created (manually, automatically or semi-automatically), following one or several of the following approaches (depending on the PDP discipline we are dealing with): - Transform an existing model view from a given viewpoint to that of another viewpoint (e.g. departing from a process model view, derive a requirements model view, or vice versa). - Improve an existing model view by creating another model (of the same viewpoint), but which displays improved privacy and data protection features, effectively changing the system (e.g. replace a system architecture with another which better implements data minimization). - Enrich an existing model view with attributes relevant to privacy and data protection, according to a given modelling profile (e.g. label deployment nodes with the data controllers and data processors which are responsible of them, label some data items as personal data or as sensitive) - Refine an existing model view with further details of the same type (e.g. add operations which were undefined, add extensions to the model, etc.)
Related use cases	- Elicitation of new system models shall *follow* the initial definition of system models to be used as an input. - Elicitation of new system models shall *follow* the instantiation of the privacy framework in a given project (the scope of the privacy framework shall be delimited before any transform is applied). - Elicitation of new system model shall *follow* the validation of the model (and later, continuous monitoring). - Risk Management method derives threat models from the architectural and data-flow models (combined with external resources such as definitions of threat sources in the respective knowledge base). - Requirements Engineering method based on ProPAN enacts this use case by deriving privacy-oriented Data Flow Diagrams, either by transforming a problem frames model, or by enriching an existent general-purpose DFD with privacy attributes. Then it derives in turn a privacy requirements model from that DFD (with the support of a catalog of meta-requirements). - Model-Driven Design method includes four activities that enact this use case: the enrichment of system data-oriented models, that of data-process oriented models, plus the respective application of strategies to either (transforming design models to others which provide better support to privacy attributes, according to the different privacy strategies). The use cases in this discipline do not provide yet the details of those activities, which are encompassed by several use cases named Update / Refine Data, Process or Architecture Elements. - Assurance method derives an update of argumentation, evidence and compliance map models from process models and system models created in other methods. This is addressed by part of the use case Develop claims and links to evidence.

Relation to GDPR	As modelling is not explicitly mandated by GDPR (see this field in the modelling use case), neither are these model transformation or refinement activities explicitly considered, so any link to GDPR would be implicit and indirect through other activities. Nonetheless, the contents of the specific model transformation activities that enact this use case can be mapped to several contents of the GDPR: - When GDPR reads that risks are elicited taking into account the nature, scope and purpose of processing operations, it implicitly means that the former are being translated into the latter. - When technical and organizational measures are introduced in a system, they are effectively modifying the system model. - Etc.
Frequency of use	Once per project iteration and discipline: whenever there is a change in the system models, the transformation or refinement activities shall be carried out again on the updated system model. If the SDLC process includes several iterations of system modelling, so shall this use case be enacted for each iteration.

4.2.6 Use case 6: Validate and monitor continuously

Validate and monitor continuously	
Purpose	Check the compliance of a system (as defined by its models) against the privacy and data protection framework.
Actors / Stakeholders	Discipline Manager, DPO
Trigger	When a new system model is created as a result of the previous use case, but also when the system is updated, and periodically to ensure that compliance is being kept.
Preconditions	A system model has been created to be subject to validation.
Assumptions	N/A
Post-conditions	The compliance of the system against the PDP framework is ensured (or, otherwise, compliance failures have been detected).
Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	
	Once a given system model has been created (which can range from quite abstract functional models, to requirement models, to architectural and detailed design models, to the system source code), their compliance against the privacy framework is assessed. This assessment may encompass, for instance: - Validation of requirements completeness against privacy properties. - Assessment of the suitability of an architectural design with respect to data minimization strategies. - Formal verification of privacy and data protection properties. - Checks that all the required activities have been carried out in the development process.
Related use cases	- The validation activities shall *follow* the creation of the model (original or transformed) that they aim to validate.

	- The validation activities are extended by the specific case of validation of externally supplied systems (including data processors which provide external data processing services). - Risk Management method includes the validation of the elicited controls in the Estimate Residual Risks use case which depends on the Risks Assessment use case (i.e. assess residual risks are at acceptable levels), and the continuous Monitoring of Control Implementation once the system has been implemented. - Model-Driven Design includes analysis of different system models to assess their compliance with given privacy properties (e.g. minimization, when strategies cannot be directly applied to improve the model) under the use cases Analyze Data Model, and Analyze Process Model; likewise the use case Analyze External Artefacts deals with the verification of some PDP requirements in the source code (previously appropriately annotated). - Requirements Engineering, Model-Driven Design and Assurance methods cooperate to ensure requirements are addressed through their traceability to system components. - Assurance method includes the validation of the assurance case (i.e. checking the evidences and argumentations) against the reference framework. This is mostly addressed by the use case Monitor Argumentation Status, and secondarily by Evaluate Artefact and Monitor Status of Assurance Project
Relation to GDPR	This use case is especially relevant to: - All the GDPR parts which establish requirements, be them targeting the development process or the systems produced, e.g. principles in Chapter 2, data subject rights in Chapter 3 and obligations in Chapter 4. - The accountability principle, as the results of this activity shall be used to demonstrate compliance. - The DPIA, as it shall be updated together with the system. - Certification purposes. In general, this use case can be related to GDPR as a whole, as the ultimate goal is to check compliance with the privacy and data protection framework.
Frequency of use	This use case is enacted at least once every system model is created. It should be also enacted whenever changes are introduced into a system, to ensure that their PDP properties still hold. And ideally, it should be carried out periodically once a system has been released, so as to prevent compliance mishaps.

4.2.7 Use case 7: Assess compliance of supply chain

Assess compliance of supply chain	
Purpose	As part of the assessment of the system compliance, ensure that the inclusion of subsystems provided by third parties keeps to the same level of compliance.

Actors / Stakeholders	Discipline Manager, DPO
Trigger	Together with a compliance assessment; in particular, whenever an external provider of functionality is to be procured.
Preconditions	N/A
Assumptions	External providers (e.g. data processors) are involved in parts of the system, and they provide system models compatible with the PDP4E approach.
Post-conditions	An assessment of suitability of an external provider is produced.
Functionality description: **(Main scenario and, if applicable normal flow steps, variations, exceptions, extensions and alternatives)**	A system is seldom created by a single organization. Typically, it depends on a hardware software, and communications platform where it executes, embeds software libraries procured as commercial off-the-shelf, etc. Quite often, it depends of remote infrastructure, platforms and services (following the as-a-service paradigm). This use case assesses the compliance of third-party providers against a given privacy framework. It may be unfeasible to apply the same assessment techniques on third-party components and services as in those developed internally, as the access to the former may be not so open. Thus, other approaches can be followed here: - Evaluate the potential risks derived from the fact that personal data leave the domain of the data controller to that of an external processor (regardless the behavior expected from the data processor). - Evaluate if processors implement (or declare to do so) the security and privacy controls that are required as a result of risk and requirements analysis. - Introduce the assurance cases provided by suppliers of different subsystems in a global system assurance case. - Etc.
Related use cases	- The assessment of the compliance of the supply chain *extends* the continuous validation (precedence relationships are inherited as well). - In the context of vendor risk management, Risk Management may help with the selection of processors according to their implementation of the controls elicited in Risk Management activities. - Model-Driven Design does not explicitly include activities that enact this use case; however, the architectural and process models include the modelling of external entities (processors), and privacy strategies also account for them (e.g. separation strategy). - Assurance includes functions to create composite assurance cases, where external providers manage their own assurance case module, to be integrated in the overall assurance case. This is not yet developed in the assurance use cases (although part of this is addressed by the use case Manage Agreement on Compliance Means), but it is explicitly considered in the method (in the form of away-elements and modules).

Relation to GDPR	GDPR introduces the concept of "[data] processor" (Art. 30), which carries out personal data processing activities on behalf of a data controller. Such data processor is required to implement similar protection measures (Art. 32) to those of the data controller; but the controller itself shall choose "only processors that provide sufficient guarantees" and have a contract signed with them that basically ensures processor's compliance on their part. GDPR reflects other situations as well where different providers intervene in data processing, e.g. joint controllers, international transfers, etc. It shall be noted that the GDPR does not include any provision regarding the responsibility of subsystem providers which do not directly operate them (e.g. operating system vendors, third party library developers, etc.).
Frequency of use	This use case should be enacted for each discipline whenever a compliance assessment is being carried out and external providers are involved. If there is a procurement processed defined, the corresponding activities should be embedded within. If an external provider updates their subsystem, they shall be reassessed; and, if possible, periodically as well (as an external system, it is more difficult to know beforehand whether changes have been introduced).

5 References

[1] PDP4E Deliverable D2.1 Multi-stakeholder specification. March 2019.

[2] PDP4E Deliverable D2.2.1 Technical gap analysis and synthesis of user requirements v1. March 2019.

[3] PDP4E Deliverable D2.2.2 Technical gap analysis and sinthesis of user requirements. March 2019.

[4] PDP4E Deliverable D3.1 Specification and design of risk management tool for data protection and privacy. July 2019.

[5] PDP4E Deliverable D3.4 Risk management methods for data protection and privacy. July 2019.

[6] PDP4E Deliverable D4.1 Specification and design of requirements engineering tool for privacy and data protection. July 2019.

[7] PDP4E Deliverable D4.4 Requirements engineering methods for privacy and data protection. July 2019.

[8] PDP4E Deliverable D5.1 Specification and design of model-driven design tool for privacy and data protection. July 2019.

[9] PDP4E Deliverable D5.4 Methods for data protection model-driven design. July 2019.

[10] PDP4E Deliverable D6.1 Specification and design of assurance tool for data protection and privacy. July 2019.

[11] PDP4E Deliverable D6.4 Assurance methods for data protection and privacy. August 2019.

[12] PDP4E Deliverable D2.6 Overall architecture and methodological framework. July 2019.

[13] IEEE Recommended Practice for Software Requirements Specifications," in IEEE Std 830-1998 , vol., no., pp.1-40, 20 Oct. 1998 doi: 10.1109/IEEESTD.1998.88286

[14] Article 29 Working Party. (2017). Guidelines on Data Protection Impact Assessment (DPIA) and determining whether processing is “likely to result in a high risk” for the purposes of Regulation 2016/679. WP248 rev. 01 As last Revised and Adopted on 4 October 2017. The Working Party on The Protection of Individuals With Regard to The Processing of Personal Data. Retrieved from https://ec.europa.eu/newsroom/document.cfm?doc_id=47711

[15] ISO 13053-2:2011. Quantitative methods in process improvement -- Six Sigma -- Part 2: Tools and techniques. International Organization for Standardization (ISO), 2011.

[16] Alberto Crespo García, Nicolás Notario McDonnell, Carmela Troncoso, Daniel Le Métayer, Inga Kroener, David Wright, José María del Álamo, Yod Samuel Martín. PRIPARE Deliverable D1.3. Updated Privacy and Security-by-design Methodology. PRIPARE consortium, 2015. Available at http://prioareproject.eu/wp-content/uploads/2013/11/PRIPARE_Deliverable_D1.3_v1.0.pdf

[17] Rasmusson, David. SIPOC picture book: A visual guide to SIPOC/DMAIC relationship. Oriel Incorporated, 2006.

[18] Cockburn, Alistair. "Structuring use cases with goals." Journal of Object-Oriented Programming 10.5 (1997): 56-62.

[19] Craig Larman. Use-Case Model: Writing Requirements In Context. In Applying UML and Patterns: An Introduction to Object-oriented Analysis and Design and the Unified Process, chapter 6, pp. 45-82. Prentice Hall Professional, 2002

[20] Karl Wiegers, Joy Beatty. Use Case Template. Seilevel, 2013. Companion content to Software Requirements. Pearson Education, 15 ago. 2013 Available at https://www.microsoftpressstore.com/store/software-requirements-9780735679665#downloads

[21] Alistair Cockburn. 2000. Writing Effective Use Cases (1st ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[22] Cockburn, Alistair. (1998). Basic Use Case Template. http://web.archive.org/web/20060613173021/http://alistair.cockburn.us/usecases/uctempla.htm

[23] Donald Firesmith, Brian Henderson-Sellers, and Ian Graham. 1998. OPEN Modeling Language (Oml) Reference Manual. Cambridge University Press, New York, NY, USA.

[24] ISO Guide 73:2009 Risk management – Vocabulary. International Organization for Standardization (ISO), 2009.

[25] Lund, Mass Soldal, Bjørnar Solhaug, and Ketil Stølen. Model-driven risk analysis: the CORAS approach. Springer Science & Business Media, 2010.

[26] ISO 31000:2018. Risk management – Guidelines. International Organization for Standardization (ISO), 2018.

[27] ISO/IEC 29134:2017. Information technology -- Security techniques -- Guidelines for privacy impact assessment. International Organization for Standardization (ISO), 2017.

[28] Pierre Bourque, Richard E. (Dick) Fairley (eds.). Guide to the Software Engineering Body of Knowledge. Version 3.0. SWEBOK. IEEE, 2014. ISBN-13: 978-0-7695-5166-1 . Available from https://www.computer.org/education/bodies-of-knowledge/software-engineering/v3

[29] Chung L., Nixon B.A., Yu E., Mylopoulos J. (2000) Introduction. In: Non-Functional Requirements in Software Engineering. International Series in Software Engineering, vol 5. Springer, Boston, MA

[30] Jon M. Siegel (ed.) Object Management Group. Model Driven Architecture (MDA) MDA Guide rev. 2.0 OMG Document ormsc/2014-06-01. Available at https://www.omg.org/cgi-bin/doc?ormsc/14-06-01

[31] Martín García, Yod Samuel, and José María del Álamo Ramiro. "A metamodel for privacy engineering methods." CEUR Workshop Proceedings, 2017.

[32] Hansen, Marit, Meiko Jensen, and Martin Rost. "Protection goals for privacy engineering." 2015 IEEE Security and Privacy Workshops. IEEE, 2015.