Problem-based Privacy Analysis (ProPAn) – A Computer-aided Privacy Requirements Engineering Method

Von der Fakultät für Ingenieurwissenschaften,
Abteilung Informatik und Angewandte Kognitionswissenschaft
der Universität Duisburg-Essen

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

Rene Meis
aus
Borken (Westf.)

1. Gutachterin: Prof. Dr. Maritta Heisel
2. Gutachterin: Prof. Dr. Simone Fischer-Hübner

Tag der mündlichen Prüfung: 20.12.2018

Abstract

With the advancing digitalization in almost all parts of our daily life, e.g., electronic health records and smart homes, and the outsourcing of data processing, e.g., data storage in the cloud and data analysis services, computer-based systems process more and more data these days. Often the processed data originate from natural persons (called data subjects) and are hence personal data possibly containing sensitive information about the individuals. Privacy in the context of personal data processing means that personal data are protected, e.g., against unwanted access and modification, that data subjects are aware about the processing practices of the controller that processes their data, and that data subjects keep control over the processing of their personal data. Privacy regulations, such as the EU General Data Protection Regulation (GDPR), aim at protecting data subjects by empowering them with rights and by putting obligations on controllers processing personal data. Not only administrative fines defined in regulations are a driver for the consideration of privacy in the development of a software-based system, also several data breaches occurred in the last years have shown that a poor consideration of privacy during the system and software development may ultimately lead to a loss of trust in and reputation of the controller.

To avoid the occurrence of data breaches and to be compliant with privacy regulations, privacy should to be considered in system and software development as a software quality from the beginning. This approach is also known as privacy-by-design. There are several challenges for privacy-by-design methods that are still not fully addressed by existing methods. First, diverse notions of privacy exist. Most of these privacy notions are non-technical and have to be refined to more technical privacy requirements that can be related to the system. Second, the system has to be analyzed for its personal data processing behavior. That is, it has to be determined which personal data are collected, stored, and provided to others by the system. Third, the privacy requirements have to be elicited that are actually relevant for the system. Fourth, the privacy risks imposed by or existing in the system have to be identified and evaluated. Fifth, measures that implement the privacy requirements and mitigate the privacy risks of the system have to be selected and integrated into the system. Sixth, privacy regulations mandate to assess the impact of the personal data processing on the data subjects. Such a privacy impact assessment (PIA) may be performed as part of a privacy-by-design method. Seventh, the conduction of a privacy-by-design method should be supported as good as possible, e.g., by a systematic method, supportive material, and computer support.

In this thesis, I propose the privacy requirements engineering method *Problem-based Privacy Analysis (ProPAn)*. The ProPAn method aims to address the aforementioned challenges starting with a system's functional requirements as input. As part of ProPAn, I provide a privacy requirements taxonomy that I derived from and mapped to various other privacy notions. This privacy requirements taxonomy addresses the first challenge mentioned above. The ProPAn method is the main contribution of my thesis and addresses the second to seventh challenge mentioned above. To address the fifth challenge in the ProPAn method, I propose an aspect-oriented requirements engineering framework that allows to model cross-cutting functionalities and to modularly integrate them into a system's functional requirements. The seventh challenge is addressed by ProPAn's computer support for the execution of the method and the documentation and validation of the method's artifacts in a machine-readable model.

Zusammenfassung

Mit der fortschreitenden Digitalisierung in beinah allen Bereichen unseres täglichen Lebens, z.B. elektronische Patientenakten und Smart Homes, und dem Outsourcing von Datenverarbeitung, z.B. Datenspeicherung in der Cloud und Datenanalysediensten, verarbeiten computerbasierte Systeme immer mehr Daten. Häufig stammen die verarbeiteten Daten von natürlichen Personen (betroffenen Personen) und sind daher personenbezogene Daten, die möglicherweise sensible Informationen über die einzelnen Personen enthalten. Im Kontext der Verarbeitung personenbezogener Daten heißt Privacy, dass personenbezogene Daten geschützt werden, z.B. gegen ungewollten Zugriff und Änderung, dass betroffene Personen sich über die Verarbeitung der personenbezogenen Daten bewusst sind, und dass die betroffenen Personen die Kontrolle über die Verarbeitung ihrer Daten behalten. Datenschutzregularien, wie die EU Datenschutz-Grundverordnung (DS-GVO), verfolgen das Ziel betroffene Personen zu stärken, indem ihnen Rechte gegeben werden, und Verantwortlichen, die personenbezogene Daten verarbeiten, Pflichten auferlegt werden. Nicht nur Geldbußen, die in Regularien definiert sind, sind ein Treiber für die Berücksichtigung von Privacy bei der Entwicklung von softwarebasierten System, auch einige Verletzungen des Schutzes personenbezogener Daten in den letzten Jahren haben gezeigt, dass eine unzureichende Berücksichtigung von Privacy während der System- und Softwareentwicklung letztlich zu einem Verlust von Vertrauen in den Verantwortlichen und dessen Ansehen führen können.

Um Verletzungen des Schutzes personenbezogener Daten zu vermeiden und konform zu Datenschutzregularien zu sein, sollte Privacy von Anfang an während der System- und Softwareentwicklung als Softwarequalität berücksichtigt werden. Dieser Ansatz ist auch als Privacy-by-Design bekannt. Es gibt einige Herausforderung für Privacy-by-Design-Methoden, die noch nicht vollständig von existierenden Methoden adressiert werden. Erstens gibt es verschiedenste Privacybegriffe. Die meisten dieser Begriffe sind nicht-technisch und müssen zu technischeren Privacyanforderungen verfeinert werden, die zu dem System in Bezug gesetzt werden können. Zweitens muss das System bezüglicher seiner Verarbeitung von personenbezogenen Daten untersucht werden. Das heißt, dass festgestellt werden muss, welche personenbezogenen Daten vom System gesammelt, gespeichert, und an Andere weitergegeben werden. Drittens müssen die Privacyanforderungen erhoben werden, die tatsächlich für das System relevant sind. Viertens müssen die Risiken bezüglich Privacy, die das System verursacht oder beinhaltet, identifiziert und evaluiert werden. Fünftens, müssen Maßnahmen, die Privacyanforderungen implementieren und Risiken bezüglich Privacy reduzieren, ausgewählt und in das System integriert werden. Sechstens schreiben Datenschutzregularien vor, die Folgen der Verarbeitung personenbezogener Daten auf die betroffenen Personen zu untersuchen. Eine solche Datenschutz-Folgenabschätzung kann als Teil einer Privacy-by-Design-Methode durchgeführt werden. Siebtens sollte die Durchführung einer Privacy-by-Design-Methode so gut wie möglich unterstützt werden, z.B. durch eine systematische Methode, unterstützendes Material, und Computerunterstützung.

In meiner Dissertation stelle ich die Anforderungsanalysemethode *Problembasierte Privacy Analyse (ProPAn)* vor, in der die Berücksichtigung von Privacy integriert ist. Die ProPAn-Methode verfolgt das Ziel, die zuvor genannten Herausforderungen zu adressieren, beginnend mit den funktionalen Anforderungen des Systems als Eingabe. Als Teil der ProPAn-Methode stelle ich eine Privacyanforderungstaxonomie bereit, die ich aus verschiedenen Privacybegriffen abgeleitet habe und zu diesen in Beziehung setze. Die ProPAn-Methode ist der Hauptbeitrag meiner Dissertation und adressiert die zweite bis siebte zuvor genannte Herausforderung. Um

die fünfte Herausforderung zu adressieren, präsentiere ich ein aspektorientiertes Anforderungsanalyseframework, das es erlaubt Querschnittsanforderungen zu modellieren und modular in die funktionalen Anforderungen eines Systems zu integrieren. Die siebte Herausforderung wird von ProPAns Computerunterstützung für die Anwendung der Methode, und der Dokumentation und Validierung der Methodenartefakte in einem maschinenlesbaren Modells adressiert.

Acknowledgments

First of all, I want to thank my wife Priscilla and our sons Tom and Malte for their continuous support, patience, and love they give to me. Their support helped me to recover from exhausting working sessions, and to stay motivated to finish my thesis. I am very grateful for the support of my parents. Without their support during my earlier studies, I would not have been able to pursue a doctoral degree. I am also grateful for the support of my parents-in-law. Due to their support, I got some extra time to write this thesis.

I thank my supervisor Maritta Heisel for her trust in me, the freedom I had to perform my duties, her encouragement, her feedback on my research, and her scientific guidance that led me through the processes of writing my doctoral thesis.

I also thank my former colleagues and associates of the working group software engineering at the University Duisburg-Essen Azadeh Alebrahim, Kristian Beckers, Nicolas Diaz Ferreya, Stephan Faßbender, Alexander Fülleborn, Nazila Gol Mohammadi, Denis Hatebur, Nelufar Ulfat-Bunyadi, Ina Wetzlaf, and Roman Wirtz for the interesting, constructive, and fruitful discussions during FoKos and Promi-Workshops, and the collaborative work on various research papers.

I especially thank Kristian Beckers and Stephan Faßbender who introduced me into the world of scientific writing and supported me during my first publications.

I also want to thank Simone Fischer-Hübner, Marit Hansen, Christos Kalloniatis, Ekkart Kindler, Thomas Santen, Ketil Stølen, Marten van Sinderen, Torben Weis, all other researchers that I met at conferences, summer schools, meetings, and external visits, and the anonymous reviewers of my research papers for the time they have taken to read or listen to my research, and to discuss my results and future directions.

Last, but not least, I thank Joachim Zumbrägel who swiftly solved technical problems with the network, printers, and SVN, and Katja Krause who supported me solving organizational issues at the University Duisburg-Essen.

Contents

List of Figures

Listings

List of Tables

List of Abbreviations

ABAC Attribute-Based Access Control

ABC Attribute-Based Credential

AORE Aspect-Oriented Requirements Engineering

AORE4PF Aspect-Oriented Requirements Engineering for Problem Frames

BPMN Business Process Model and Notation

DFD Data Flow Diagram

DFG Data Flow Graph

DPD EU Data Protection Directive

DPIA Data Protection Impact Assessment

EHR Electronic Health Record

EHS Electronic Health System

EMF Eclipse Modeling Framework

EU European Union

FIP Fair Information Practice

FOL First Order Logic

GDPR EU General Data Protection Regulation

GPS General Positioning System

HAZOP Hazard and Operability Studies

ID Identifier

IEC International Electrotechnical Commission

ISO International Organization for Standardization

ITS Intelligent Transportation Systems

MSC Message Sequence Chart

NIST National Institute of Standards and Technology

NSA American National Security Agency

OCL Object Constraint Language

OSN Online Social Network

OWL Web Ontology Language

PCM Privacy Conceptual Model

PA-DFD Privacy-aware Data Flow Diagram

PET Privacy Enhancing Technology

PIA Privacy Impact Assessment

PII Personally Identifiable Information

PIM Personal Information Map

PLA Privacy Level Agreement

PMRM Privacy Management Reference Model and Methodology

PNF Privacy Normative Framework

PRIOP Privacy and Operability Studies

ProPAn Problem-based Privacy Analysis

PTA Privacy Threshold Analysis

RQ Research Question

SEMDM Metamodel for Software and Systems Development Methodologies

SGAM European Smart Grid Architecture Model

SMS Study Management System

SSL Secure Sockets Layer

UDEPF University Duisburg-Essen Problem Frames

UML Unified Modeling Language

Part I.

Foundations

Chapter 1

Motivation and Overview

In this chapter, I first motivate the problem addressed by my dissertation in Section 1.1. The research questions that are tackled in my thesis are introduced in Section 1.2. Section 1.3 provides an overview of the contributions of my thesis and their relation to the research questions. Section 1.4 describes the structure of the thesis and which contributions are described in which chapters. The publications on which this thesis is based are listed in Section 1.5.

1.1. Motivation

The topic of privacy and data protection receives more and more attention in our society. The public awareness for privacy was especially raised after Edward Snowden's revelations about the massive data collection and processing actions of the American National Security Agency (NSA) (Eaton and Piven, 2014). Security researchers also regularly raise the awareness for privacy in our society by revealing insufficient data protection measures in services or products offered by companies. For example, the "Hello Barbie", a doll for children produced by Martell, allows a child to ask questions that the Barbie doll then answers using an internet service. Security researchers have shown that it is easily possible to eavesdrop the communication of the "Hello Barbie" and hence, to spy the child playing with the doll (Gibbs, 2015). A similar example are the teddy bears from CloudPets. Voice messages can be exchanged between a teddy bear of CloudPets and a computer or smart phone. The voice messages are stored in a database of the provider. The teddy bear shall allow a child to stay in touch with a relative that lives far away or needs to stay aboard for some period. The security researcher Hunt (2017) has shown how the database which contains all recorded voice messages that are exchanged using teddy bears of CloudPets can be accessed with access to all stored voice messages. In 2011, it was uncovered that Apple's iPhones store regularly the location of the smart phone (in the form of its GPS coordinates) together with a timestamp in a secret file on the phone. These location data are also copied to the user's computer when the iPhone is synchronized with it (Arthur, 2011). The recorded location data are so fine-grained that it shows where an iPhone user was during the day, when, and how long he or she was there. Anyone who has access to the iPhone or the computer with which it was synchronized potentially has access to this sensitive personal data.

These examples show that the digitalization of our everyday life does not only provide new possibilities to keep in touch with people and to simplify things by providing convenience services, but that it also introduces privacy issues that were not existing before.

Another example, are online social networks (OSNs) that are used by millions of people world wide to provide information to or communicate with other members of the OSNs. OSN users shall benefit from a simpler way to maintain social contacts, e.g., to keep in touch with their families and friends, in comparison to the offline world. However, OSNs introduce manifold threats to the privacy of their users. On one hand, threats emerge from the OSN's provider, because most OSNs are centralized, i.e., all collected data are controlled by the OSN's provider.

In consequence, all data shared in the OSN are also shared with the service provider. Most service providers use the data provided by the users, e.g., to sell targeted advertisements to third parties. In other words, the users' personal data are the capital of OSNs. Hence, a privacy issue of OSNs is that the providers goal is to collect as much personal data of their users as possible to use these personal data to generate profit by selling services based on these data to third parties, e.g., targeted advertisements. Other privacy issues of OSNs emerge from their users. Often OSN users are not aware of the consequences of sharing certain information and they later regret that they have shared that information (Wang et al., 2011). Sharing the wrong information with the wrong audience can lead to situations such as job loss, identity theft, or bad image.

The key point that makes privacy an issue in the digital age is that computer systems are able to collect, store, and process large amounts of data. It is even possible to derive sensitive information from in the first place uncritical information. For example, researchers have shown that it is possible to deduce behavioral patterns of household members, e.g., absence or sleep times, from the electricity consumption measured by a smart electricity meter (Jin et al., 2014). Other researchers have shown how personal views can be derived from a person's Facebook likes[1] Youyou et al. (2015).

Hadar et al. (2017) published a study about the software developers' privacy mindset. This study points out an additional issue, namely, practical software engineers mostly understand privacy as security, or more specifically as confidentiality. Other aspects such as transparency and intervenability are not recognized as privacy properties by the software engineers participating in the study. The study shows that there is a need for tools that support developers to identify the privacy needs of the software-to-be, because these are not limited to the privacy as confidentiality paradigm.

In summary, every digital innovation and digitalization of our daily life has the potential to harm our privacy. Hence, privacy should be considered right from the beginning in development of products and services. Addtionally, it has to be understood that privacy is not limited to security. Several organizations and authorities acknowledge these issues and formulated privacy principles that serve as guidelines to consider privacy right from the beginning in the development of products and services. The approach to consider privacy as integral part of the development is also called *privacy-by-design*. The most prominent privacy principles were formulated by Cavoukian (2011). She proposes the following seven principles:

1. Proactive not Reactive; Preventative not Remedial
2. Privacy as the Default Setting
3. Privacy Embedded into Design
4. Full Functionality — Positive-Sum, not Zero-Sum
5. End-to-End Security — Full Lifecycle Protection
6. Visibility and Transparency — Keep it Open
7. Respect for User Privacy — Keep it User-Centric

In Section 2.4, I provide an overview of the privacy principles proposed by different organizations and authorities and discuss their relation to each other. From Cavoukian's principles, we

[1] On Facebook, it is possible to mark posts, comments, and profiles. This marking is called *like*. Such a like indicates that the user is in favor with the post, comment, or profile. Depending on a user's privacy settings the made likes are visible to the public, friends, or are private.

can already see that these are rather coarse grained and do not provide sufficient details on how to systematically address privacy-by-design in the development of a product or service.

The emerging discipline privacy engineering (Spiekermann and Cranor, 2009; Gürses and del Álamo, 2016) is concerned with the development of methods and tools following the privacy-by-design principles. In general, privacy engineering targets systems engineering, i.e., privacy engineering is concerned with all steps of systems development including the development of software and hardware, and even organizational structures and processes. A sub-discipline of privacy engineering is privacy requirements engineering. In this discipline, the privacy-by-design principles are integrated into the requirements analysis phase of the software development.

In my thesis, I contribute to the field of privacy requirements engineering. My objective is to support a systematic consideration of privacy in the early phases of software development, namely during the requirements engineering phase. I propose the Problem-based Privacy Analysis (ProPAn) method in this thesis. The ProPAn method consists of several steps that can be performed by requirements engineers with the support of privacy and domain experts. Most steps of the ProPAn method are tool-supported, i.e., the created outputs are electronically stored and computer-readable, and parts of the ProPAn method are automated. I give an overview of the ProPAn method in Chapter 8.

1.2. Research Questions

My main research question that arises from the previously given motivation is the following.

> **How can requirements engineers be supported to consider privacy as a software quality during the requirements analysis starting with a given set of functional requirements?**

The main research question can be refined into the following more detailed research questions:

RQ 1 What kinds of privacy requirements exist?

Different notions and concepts of privacy and data protection exist and there is no complete and commonly agreed taxonomy of privacy requirements for software systems. A commonly agreed taxonomy of privacy requirements would have the advantage that privacy analyses of different software systems would be comparable and to have a reference list of privacy requirements.

RQ 2 Which knowledge in addition to a software's functional requirements is needed for a privacy analysis of these?

To understand the privacy needs of a software system, it is necessary to understand the environment it is integrated into and also the entities it possibly indirectly influences. For example, a software system may process personal data of persons that are not directly interacting with the software. Hence, a software's functional requirements do not suffice to perform a proper privacy analysis.

RQ 3 How can requirements engineers systematically identify the personal data the software system shall process and the data subjects of these data?

Privacy becomes an issue for software systems if these are concerned with the processing of personal data. Hence, we have first to identify whether, what, and whose personal data are processed by the software system.

RQ 4 How to support requirements engineers to understand how the identified personal data are processed by the software system?

After having identified the personal data that are processed by the software system, we have to understand how that data are processed. That is, we have to find out which parts of the software system collect, store, and further provide personal data to other entities.

RQ 5 How to systematically derive a software's privacy requirements?

Requirements engineers have to identify the privacy requirements relevant for the software system they analyze. The privacy requirements depend on how the software system processes the personal data it is concerned with and on the stakeholders' needs.

RQ 6 How to identify and assess the risks of threats to a software's privacy requirements?

Having identified which privacy requirements a software system shall satisfy, we need to assess under which circumstances the software system can meet its privacy requirements and which situations may threaten the privacy requirements' satisfaction. The risk implied by identified privacy threats needs to be assessed to decide whether these need to be treated or whether they are acceptable.

RQ 7 How can requirements engineers be supported to treat privacy risks and to implement privacy requirements?

To treat a privacy threat or to implement privacy requirements, different privacy enhancing technologies (PETs) exist. However, the selection of the right PET and the integration of it into the existing set of functional requirements is not trivial.

RQ 8 Can artifacts of a privacy requirements analysis be used to create the documentation of a privacy impact assessment?

Privacy impact assessments (PIAs) (also known as data protection impact assessments) are mandatory in different legislations. For example, the EU General Data Protection Regulation (European Commission, 2016) prescribes an assessment of the impact of a software system's data processing activities on the privacy of the data subjects. The artifacts produced during a privacy-by-design process are likely to be valuable for a PIA.

1.3. Contributions

To answer the previously introduced research questions, my thesis contains three main contributions. These are a privacy requirements taxonomy, the Problem-based Privacy Analysis (ProPAn) method, and the Aspect-Oriented Requirements Engineering for Problem Frames (AORE4PF) framework. Figure 1.1 shows the contributions of my thesis and their relation to each other and the research questions. Several contributions are based on other contributions. This is visualized with a dashed, directed edge between two contributions. A contribution is connected to the research question it addresses with a dotted, directed edge.

1.3.1. Privacy Requirements Taxonomy

Our proposed Privacy Requirements Taxonomy aligns different privacy notions from the literature into a homogeneous taxonomy of privacy requirements. It contains as sub-contributions taxonomies for the privacy goals transparency (Transparency Requirements Taxonomy) and intervenability (Intervenability Requirements Taxonomy). These two privacy goals and their role as

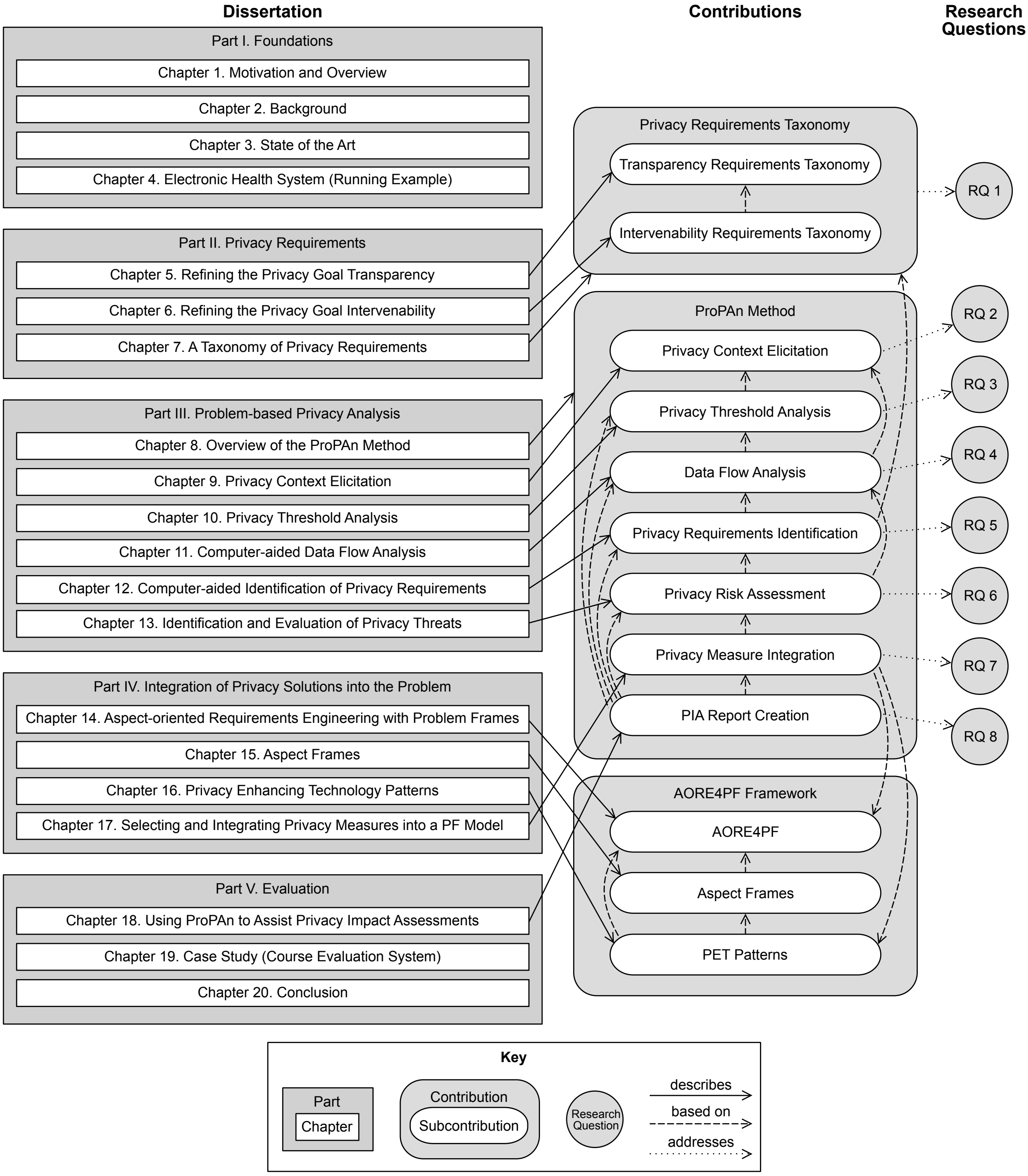

Dissertation
Contributions
Research Questions
Part I. Foundations
Chapter 1. Motivation and Overview
Chapter 2. Background
Chapter 3. State of the Art
Chapter 4. Electronic Health System (Running Example)
Part II. Privacy Requirements
Chapter 5. Refining the Privacy Goal Transparency
Chapter 6. Refining the Privacy Goal Intervenability
Chapter 7. A Taxonomy of Privacy Requirements
Part III. Problem-based Privacy Analysis
Chapter 8. Overview of the ProPAn Method
Chapter 9. Privacy Context Elicitation
Chapter 10. Privacy Threshold Analysis
Chapter 11. Computer-aided Data Flow Analysis
Chapter 12. Computer-aided Identification of Privacy Requirements
Chapter 13. Identification and Evaluation of Privacy Threats
Part IV. Integration of Privacy Solutions into the Problem
Chapter 14. Aspect-oriented Requirements Engineering with Problem Frames
Chapter 15. Aspect Frames
Chapter 16. Privacy Enhancing Technology Patterns
Chapter 17. Selecting and Integrating Privacy Measures into a PF Model
Part V. Evaluation
Chapter 18. Using ProPAn to Assist Privacy Impact Assessments
Chapter 19. Case Study (Course Evaluation System)
Chapter 20. Conclusion
Privacy Requirements Taxonomy
Transparency Requirements Taxonomy
Intervenability Requirements Taxonomy
ProPAn Method
Privacy Context Elicitation
Privacy Threshold Analysis
Data Flow Analysis
Privacy Requirements Identification
Privacy Risk Assessment
Privacy Measure Integration
PIA Report Creation
AORE4PF Framework
AORE4PF
Aspect Frames
PET Patterns
RQ 1
RQ 2
RQ 3
RQ 4
RQ 5
RQ 6
RQ 7
RQ 8
Key
Part
Chapter
Contribution
Subcontribution
Research Question
describes
based on
addresses

Figure 1.1.: *Overview of dissertation structure, contributions, and research questions*

software requirements have not been deeply studied in the literature before. As shown in Figure 1.1, the taxonomy of intervenability requirements is based on the taxonomy of transparency requirements, because transparency is a prerequisite for intervenability.

The Privacy Requirements Taxonomy addresses RQ 1, by providing an overview of privacy requirements and a discussion of how the taxonomy is related to other privacy notions.

1.3.2. AORE4PF Framework

The AORE4PF Framework consists of the extension of Jackson's problem frames approach with aspect-oriented requirements engineering concepts (Rashid et al., 2002), called aspect-oriented requirements engineering for problem frames (AORE4PF), a collection of Aspect Frames that are patterns representing classes of functional cross-cutting concerns , and a collection of PET Patterns that present the information how privacy enhancing technologies (PETs) can be integrated into functional requirements to satisfy certain privacy requirements. Both Aspect Frames and PET Patterns are based on the AORE4PF's notation. Furthermore, the PET Patterns are based on the Aspect Frames that assist the creation of the PET Patterns.

The AORE4PF Framework itself does not address one of the introduced research questions, but the Privacy Measure Integration of the ProPAn Method is based on AORE4PF and PET Patterns. Thus, the AORE4PF Framework implicitly contributes to RQ 7 by providing the foundations for the integration of privacy solutions into a set of functional requirements.

1.3.3. ProPAn Method

The ProPAn Method is the main contribution of this thesis. With the ProPAn Method, I developed a systematic and computer-aided method to perform a privacy analysis of a software, system starting from its functional requirements. The ProPAn Method is model-based. That is, every step of the ProPAn Method is based on models produced in previous steps and produces new model artifacts or output based on the input models. The ProPAn Method starts with a set of functional requirements represented as problem diagrams (Jackson, 2000), which are stored in a problem frame model (see Section 2.3). This contribution consists of seven sub-contributions.

1.3.3.1. Privacy Context Elicitation

In the first step of the ProPAn Method, we identify additional privacy-relevant domain knowledge based on questionnaires and add this knowledge to the problem frame model that also contains the functional requirements.

This sub-contribution addresses RQ 2 by providing means to identify from the requirements model the additional domain knowledge that is needed for a privacy analysis and to add this domain knowledge to the requirements model.

1.3.3.2. Privacy Threshold Analysis

To assess whether a detailed privacy analysis is necessary, we perform a so-called Privacy Threshold Analysis. During this threshold analysis, we identify from the functional requirements and domain knowledge the personal data that are processed by the software system, the data subjects, i.e., the individuals related to the personal data, and the properties of the personal data. Properties of personal data are, e.g., the sensitivity and linkability to their data subjects. From the functional requirements and domain knowledge, we then over-approximate the data flows in the software system. The over-approximation contains all data flows of the software system that are possible due to the documented requirements and domain knowledge. Additionally, the over-approximation includes some false-positives, i.e., some identified data flows do not occur in practice. Based on the data flows and the identified personal data, we determine whether the software system might include or introduce privacy issues or not.

The Privacy Threshold Analysis is based on the Privacy Context Elicitation, because we need the additional domain knowledge for the identification all personal data that are processed by the software system and all data flows occurring in it.

RQ 3 is addressed by the Privacy Threshold Analysis by providing a systematic method that allows requirements engineers to identify the personal data processed by the software system and the related data subjects.

1.3.3.3. Data Flow Analysis

During the Data Flow Analysis, we systematically elicit the data flows implied by the functional requirements and domain knowledge that were only over-approximated in the Privacy Threshold Analysis. As result of this step, we obtain a model that contains the information at which entities of the software system which personal data are available, how these are linkable to each other, and between which entities which personal data flows and due to which functional requirements and domain knowledge.

The Data Flow Analysis is based on the personal data identified in the step Privacy Threshold Analysis. Additionally, it needs the given functional requirements and the domain knowledge elicited during the Privacy Context Elicitation.

The systematic Data Flow Analysis addresses RQ 4, because it helps to elicit the information how the identified personal data are processed by the software system.

1.3.3.4. Privacy Requirements Identification

From the elicited information how the personal data are processed, we derive the privacy requirements implied by this planned processing in the step Privacy Requirements Identification.

The Privacy Requirements Identification is based on the Data Flow Analysis and the proposed Privacy Requirements Taxonomy. That is, we generate instances of privacy requirements from the proposed taxonomy using the elicited data flows.

By deriving privacy requirements in a semi-automated manner, the Privacy Requirements Identification addresses RQ 5.

1.3.3.5. Privacy Risk Assessment

During the Privacy Risk Assessment, we analyze under which circumstances the elicited privacy requirements might be violated and the risk implied by such a violation. To identify situations in which privacy requirements are violated, we consider deviations of the software systems expected behavior.

The Privacy Risk Assessment is based on the privacy requirements that result from the Privacy Requirements Identification and the expected data flows elicited during the Data Flow Analysis.

The Privacy Risk Assessment contributes to RQ 6 by providing a method to identify threats to the elicited privacy requirements and to further assess the threats' implied risks.

1.3.3.6. Privacy Measure Integration

To mitigate unacceptable privacy risks and to implement privacy requirements, appropriate privacy-enhancing technologies (PETs) have to be selected and integrated into the software system, or assumptions about the behavior of certain entities have to be made, e.g., compliance to regulations and contracts. This selection and integration of PETS and assumptions is supported during the Privacy Measure Integration. We propose an aspect-oriented approach for the representation and integration of PETs.

The Privacy Measure Integration is based on the identified privacy requirements of the step Privacy Requirements Identification and the unacceptable risks of the step Privacy Risk Assessment. The selection of the PETs is based on the proposed PET Patterns and their integration is based on AORE4PF.

RQ 7 is addressed by the Privacy Measure Integration, because this step provides support for the treatment of privacy risks and the implementation of privacy requirements.

1.3.3.7. PIA Report Creation

As already stated, privacy impact assessments (PIAs) are mandatory in different legislations. A PIA shall document, among others, what kinds of personal data a software system processes, for which purpose the data are processed, which risks this processing potentially implies, and how these risks shall be mitigated (Wright et al., 2011). The artifacts produced by several steps of the ProPAn method are used to derive information needed to create a PIA report during the PIA Report Creation.

The PIA Report Creation is based on the personal data identified during the Privacy Threshold Analysis, the flows of the personal data elicited during the Data Flow Analysis, the privacy requirements derived in the Privacy Requirements Identification, the risks identified during the Privacy Risk Assessment, and the measures selected during the Privacy Measure Integration.

RQ 8 is addressed by the PIA Report Creation by providing a method to use artifacts of the ProPAn method to create the documentation for a PIA.

1.4. Dissertation Overview

My thesis is structured into five parts. Figure 1.1 shows the parts and the chapters they contain. Additionally it shows which chapters describe which contributions.

Part I motivates my thesis and provides its background. This chapter (Chapter 1) provides the motivation and an overview of my thesis. Chapter 2 introduces the relevant background as foundation of the contributions of this thesis. Chapter 3 describes the state of the art in privacy (requirements) engineering and discusses the gaps in the research that I want to fill with this thesis. Chapter 4 introduces a small electronic health system (EHS) that is used as running example throughout the thesis.

In Part II, I provide a taxonomy of privacy requirements that is used as basis for the later privacy analysis. Chapter 5 and Chapter 6 introduce Transparency and Intervenability Requirements Taxonomys, respectively. The Privacy Requirements Taxonomy, which includes the taxonomies for the privacy goals transparency and intervenability, used in this thesis is introduced and discussed in Chapter 7.

The ProPAn Method is described in Part III. Chapter 8 provides an overview of the steps of the ProPAn Method. The Privacy Context Elicitation that aims at the identification of additional domain knowledge for a privacy analysis is discussed in Chapter 9. Chapter 10 describes how ProPAn's Privacy Threshold Analysis is performed based on the given functional requirements and the additionally elicited domain knowledge. This Privacy Threshold Analysis also includes the identification of the personal data that are processed by the software system and the corresponding data subjects. The Data Flow Analysis that guides requirements engineers to assess how the software system collects and stores personal data, provides personal data to others, and deduces other data from them is described in Chapter 11. In Chapter 12, I explain how privacy requirements can be deduced from the elicited processing behavior during the Privacy Requirements Identification. ProPAn's Privacy Risk Assessment that guides the identification of situations that imply privacy risks and the evaluation of these risks is described in Chapter 13.

Part IV is concerned with the integration of privacy solutions into the software system's functional requirements to mitigate privacy risks and implement privacy requirements. For this, I first introduce the aspect-oriented extension of Jackson's problem frames approach AORE4PF in Chapter 14. AORE4PF allows to describe how cross-cutting functionality can be handled in combination with the problem frames approach. As most solutions for quality requirements are cross-cutting, I propose to consider also privacy solutions, i.e., privacy enhancing technologies, as cross-cutting functionalities, so-called aspects. Chapter 15 introduces four Aspect Frames that are patterns for classes of cross-cutting functionalities and assist the formulation and modeling of aspects. Then I introduce PET Patterns in Chapter 16. A PET pattern presents a PET in a given structure and an aspect-oriented notation. This presentation shall help requirements engineers to assess the properties of a PET and to compare it to other PETs. Additionally, a PET pattern contains the information into what kind of functional requirements the PET may be integrated into and how. Chapter 17 then describes my proposed Privacy Measure Integration, which is a method that helps requirements engineers to select and integrate PETs as aspect into the requirements model based on the PET patterns.

The last part of my thesis (Part V) aims at the evaluation of the contributions of my thesis and concludes it. In Chapter 18, I explain how artifacts of the ProPAn method can be used to support the PIA report creation. This chapter shows the benefits of the ProPAn Method, i.e., information that is needed for a PIA is systematically elicited and documented in a re-usable way by the ProPAn Method. Chapter 19 contains the application of the ProPAn Method to a real world case study. This case study is concerned with the development of a course evaluation system that shall allow students to provide feedback on the courses they participated in. This feedback shall be made available to the teachers of these courses in a form that does not allow a teacher to identify the student that provided a specific feedback. I finally conclude my thesis in Chapter 20.

1.5. Publications

My thesis is based on 14 publications. I am the lead author of all these publications. Table 1.1 lists these publications and references for each publication the chapters that are based on it.

Table 1.1.: *Publications on which the chapters of this thesis are based*

Publication	Chapters
Rene Meis, Maritta Heisel, and Roman Wirtz. A taxonomy of requirements for the privacy goal transparency. In *Trust, Privacy, and Security in Digital Business*, LNCS 9264, pages 195–209. Springer, 2015. doi: 10.5220/0005518500430052. URL `http://dx.doi.org/10.5220/0005518500430052`	5
Rene Meis and Maritta Heisel. Understanding the privacy goal intervenability. In *Trust, Privacy, and Security in Digital Business*, volume 9830 of *LNCS*, pages 79–94. Springer, 2016c. doi: 10.1007/978-3-319-44341-6_6. URL `https://link.springer.com/chapter/10.1007/978-3-319-44341-6_6`	6
Rene Meis and Maritta Heisel. Computer-aided identification and validation of intervenability requirements. *Information*, 8(30), 2017b. ISSN 2078-2489. doi: 10.3390/info8010030. URL `http://www.mdpi.com/2078-2489/8/1/30`	6, 7, 12
Rene Meis and Maritta Heisel. Computer-aided identification and validation of privacy requirements. *Information*, 7(28), 2016b. ISSN 2078-2489. doi: 10.3390/info7020028. URL `http://www.mdpi.com/2078-2489/7/2/28`	7, 8, 12
Rene Meis. Problem-Based Consideration of Privacy-Relevant Domain Knowledge. In *Privacy and Identity Management for Emerging Services and Technologies*, volume 421 of *IFIP Advances in Information and Communication Technology*. Springer, 2014. doi: 10.1007/978-3-642-55137-6_12. URL `http://dx.doi.org/10.1007/978-3-642-55137-6_12`	9

Kristian Beckers, Stephan Faßbender, Stefanos Gritzalis, Maritta Heisel, Christos Kalloniatis, and Rene Meis. Privacy-aware cloud deployment scenario selection. In *Trust, Privacy, and Security in Digital Business*, LNCS 8647, pages 94–105. Springer, 2014a. doi: 10.1007/978-3-319-09770-1_9. URL `http://dx.doi.org/10.1007/978-3-319-09770-1_9`	9
Kristian Beckers, Stephan Faßbender, Maritta Heisel, and Rene Meis. A problem-based approach for computer aided privacy threat identification. In D. Ikonomou and Bart Preneel, editors, *Privacy Technologies and Policy*, volume 8319 of *LNCS*, pages 1–16. Springer, 2014b. doi: 10.1007/978-3-642-54069-1_1. URL `http://dx.doi.org/10.1007/978-3-642-54069-1_1`	10
Rene Meis and Maritta Heisel. Systematic identification of information flows from requirements to support privacy impact assessments. In *ICSOFT-PT 2015 - Proc. of the 10th Int. Conf. on Software Paradigm Trends*, pages 43–52. SciTePress, 2015. doi: 10.5220/0005518500430052. URL `http://dx.doi.org/10.5220/0005518500430052`	11
Rene Meis and Maritta Heisel. Supporting privacy impact assessments using problem-based privacy analysis. In *Software Technologies - 10th International Joint Conference, ICSOFT 2015, Revised Selected Papers*, volume 586 of *Communications in Computer and Information Science*, pages 79–98. Springer, 2016a. ISBN 978-3-319-30141-9. doi: 10.1007/978-3-319-30142-6_5. URL `http://dx.doi.org/10.1007/978-3-319-30142-6_5`	11, 18
Rene Meis and Maritta Heisel. Towards systematic privacy and operability (PRIOP) studies. In *ICT Systems Security and Privacy Protection*, volume 502 of *IFIP AICT*, pages 427–441. Springer, 2017c. doi: 10.1007/978-3-319-58469-0_29. URL `http://dx.doi.org/10.1007/978-3-319-58469-0_29`	13
Stephan Faßbender, Maritta Heisel, and Rene Meis. Aspect-oriented requirements engineering with problem frames. In *ICSOFT-PT 2014 - Proc. of the 9th Int. Conf. on Software Paradigm Trends*, pages 145–156. SciTePress, 2014. doi: 10.5220/0005001801450156. URL `http://dx.doi.org/10.5220/0005001801450156`	14
Stephan Faßbender, Maritta Heisel, and Rene Meis. A problem-, quality-, and aspect-oriented requirements engineering method. In *Software Technologies - 9th International Joint Conference, ICSOFT 2014, Vienna, Austria, August 29-31, 2014, Revised Selected Papers*, volume 555 of *Communications in Computer and Information Science*, pages 291–310. Springer, 2015. doi: 10.1007/978-3-319-25579-8_17. URL `http://dx.doi.org/10.1007/978-3-319-25579-8_17`	14
Rene Meis and Maritta Heisel. Aspect frames – describing cross-cutting concerns in aspect-oriented requirements engineering. In *Proceedings of the 22nd European Conference on Pattern Languages of Programs*, number 25 in EuroPLoP '17, page 28. ACM, 2017a. doi: 3147704.3147732. URL `https://doi.org/10.1145/3147704.3147732`	15
Rene Meis and Maritta Heisel. Pattern-based representation of privacy enhancing technologies as early aspects. In *Trust, Privacy, and Security in Digital Business*, volume 10442 of *LNCS*, pages 49–65, Cham, 2017d. Springer International Publishing. doi: 10.1007/978-3-319-64483-7_4. URL `https://doi.org/10.1007/978-3-319-64483-7_4`	16

CHAPTER 2

Background

In this chapter, I introduce the background of my thesis. In Section 2.1, I introduce requirements engineering and the terminology that I use in my thesis. The requirements engineering approach on which I base my privacy requirements engineering methodology is introduced in Section 2.2. The tool support that I developed for model-based requirements engineering following the problem frames approach is introduced in Section 2.3. Based on this tool support, I also developed the ProPAn tool. Finally, I introduce the concept of privacy and its consideration in information systems in Section 2.4.

2.1. Requirements Engineering Terminology

The research presented in my thesis contributes to the requirements engineering phase of software development. The analysis of the requirements is the first phase of traditional and agile software development processes (Pfleeger, 1998; Beck, 2000). The goal of this phase is to understand the problem that the software that shall be developed (software-to-be) shall address and which entities are involved in this process and will interact with the software-to-be.

Requirements can be managed in different ways. Requirements can be documented as plain text in a document or structured using requirements tools such as IBM rational doors (IBM, 2017). In agile software development, the requirements are documented as so-called user stories (Beck, 2000). A user story represents a single user's need, e.g., a usage scenario. There exist also several graphical and model-based notations to represent and organize requirements, e.g., UML use cases (Bock et al., 2015), goal-oriented notations such as KAOS (van Lamsweerde et al., 1998), i^* (Yu, 1997), TROPOS (Fuxman et al., 2001) and GBRAM (Antón, 1996), and problem-oriented notations such as the problem frames approach (Jackson, 2000).

The understanding of the term *requirement* varies in the literature and the above mentioned notations. In general a requirement is an optative statement that the software-based system shall satisfy if the software-to-be is integrated into it. The understanding of the term requirement ranges from high-level *goals* of involved stakeholders, e.g., "The health of patients shall be improved", to precise *specifications* that a software needs to address, e.g., "If the received vital signs are out of the predefined ranges, then issue an alarm".

In my thesis, I use the terminology introduced by Jackson (2000). This terminology is visualized in Figure 2.1. In Jackson's terminology, the software development problem is about the construction of a Machine (the software that shall be developed) to be integrated into an Environment consisting of Domains, e.g., humans, technical devices, and physical data representations. The machine and the domains are both represented as squares in Figure 2.1. The System consists of the machine and the environment. Throughout the thesis, I use the terms system and system-to-be, and also the terms machine and software-to-be synonymously.

The domains of the system (including the machine) are connected by *interfaces* (depicted as solid lines in Figure 2.1). These interfaces consist of *phenomena* (p with subscript in Figure 2.1)

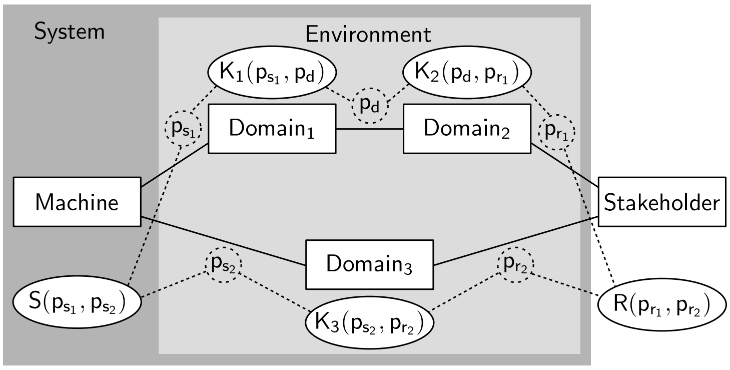


Figure 2.1.: *Illustration of Jackson's terminology for requirements engineering*

that are shared among the connected domains, i.e., every phenomenon of an interface can be observed by every connected domain. Each phenomenon is controlled by exactly one domain. Phenomena can be *causal*, e.g., events, actions, messages, and operations, or *symbolic*, e.g., data and states. Controlling a causal phenomenon means to issue the phenomenon and controlling a symbolic phenomenon means to manage the phenomenon's value.

A requirement (R) is an optative statement that describes the desired behavior of the system established by the machine from the point of view of a Stakeholder. A requirement (R) *refers to* and *constrains* phenomena of domains of the environment that the Stakeholder can observe or is aware of. Consequently, a requirement is likely to be about phenomena that the machine is not able to observe or control, and possibly about domains that the machine is not (directly) connected to. The phenomena that the requirement is referring to are also called *requirement phenomena.* In Figure 2.1, $\mathsf{p}_{\mathsf{r}_1}$ and $\mathsf{p}_{\mathsf{r}_2}$ represent the requirement phenomena. The requirement R is a statement about these phenomena which is expressed using the notation $\mathsf{R}(\mathsf{p}_{\mathsf{r}_1}, \mathsf{p}_{\mathsf{r}_2})$.

The task of requirements engineers is to derive the *specification* of the machine. A specification is an optative statement about the behavior of the machine referring to phenomena observable by the machine and constraining phenomena controlled by it. The phenomena the specification is referring to are also called *specification phenomena.* In Figure 2.1, this is represented using the notation $\mathsf{S}(\mathsf{p}_{\mathsf{s}_1}, \mathsf{p}_{\mathsf{s}_2})$.

To derive the specification, requirements engineers need to translate the requirement phenomena ($\mathsf{p}_{\mathsf{r}_1}$ and $\mathsf{p}_{\mathsf{r}_2}$) into specification phenomena ($\mathsf{p}_{\mathsf{s}_1}$ and $\mathsf{p}_{\mathsf{s}_2}$). This process is also called *requirements progression* (for details see also the work of Seater et al. (2007)). During this progression *domain knowledge* is identified and documented. Domain knowledge consists of indicative statements about the domains of the environment that can either be *facts*, i.e., truths that are valid under all circumstances, or *assumptions*, i.e., statements that are assumed to be valid, but that may be violated under specific circumstances. The domain knowledge refers to and constrains specification phenomena, requirement phenomena, and possibly also phenomena in the environment that are neither observable or controlled by the machine, nor referenced in the requirement (e.g., p_d in Figure 2.1). Figure 2.1 contains the domain knowledge $\mathsf{K}_1(\mathsf{p}_{\mathsf{s}_1}, \mathsf{p}_\mathsf{d})$ explaining the relation between the phenomena $\mathsf{p}_{\mathsf{s}_1}$ and p_d, $\mathsf{K}_2(\mathsf{p}_\mathsf{d}, \mathsf{p}_{\mathsf{r}_1})$ explaining the relation between the phenomena p_d and $\mathsf{p}_{\mathsf{r}_1}$, and $\mathsf{K}_3(\mathsf{p}_{\mathsf{s}_2}, \mathsf{p}_{\mathsf{r}_2})$ explaining the relation between the phenomena $\mathsf{p}_{\mathsf{s}_2}$ and $\mathsf{p}_{\mathsf{r}_2}$

The result of the requirements progression is the specification of the machine and the domain knowledge that explains the relation between the specification and requirement phenomena. Additionally, it needs to be argued that the specification together with the domain knowledge imply the satisfaction of the requirement. Zave and Jackson (1997) denote this necessary entailment relationship as $S, K \vdash R$, where S is the set of specifications, K the set of domain knowledge, and R the set of requirements. In terms of Figure 2.1, we obtain the entailment relationship

$$\mathsf{S}(\mathsf{p}_{\mathsf{s}_1}, \mathsf{p}_{\mathsf{s}_2}), \mathsf{K}_1(\mathsf{p}_{\mathsf{s}_1}, \mathsf{p}_{\mathsf{d}}), \mathsf{K}_2(\mathsf{p}_{\mathsf{d}}, \mathsf{p}_{\mathsf{r}_1}), \mathsf{K}_3(\mathsf{p}_{\mathsf{s}_2}, \mathsf{p}_{\mathsf{r}_2}) \vdash \mathsf{R}(\mathsf{p}_{\mathsf{r}_1}, \mathsf{p}_{\mathsf{r}_2}).$$

The kind of requirements that Jackson considers in his terminology are functional requirements that are already sufficiently decomposed to assign the responsibility for their satisfaction to a part of the software-to-be. In practice, the stakeholders' needs, also called *goals*, need first to be refined. van Lamsweerde et al. (1998) propose in their KAOS method to iteratively refine the overall goal of the software development problems into sub-goals until these can be assigned to agents that are responsible for their satisfaction. An agent can be the software-to-be or parts of it, or entities of the environment into which the software shall be integrated into. van Lamsweerde et al. call goals assigned to a part of the software-to-be requirements and those assigned to entities of the environment expectations. During the refinement, it may be necessary to introduce knowledge about the environment (domain knowledge). For each refinement, it has to be argued why the sub-goals satisfy their parent goal. This argumentation is similar to the entailment relationship introduced above. The above stated entailment relationship is actually stating that the refinement of the requirement into the specification and additional domain knowledge is sufficient to satisfy this requirement. van Lamsweerde et al. and Jackson have a similar understanding of the term requirement as also argued by van Lamsweerde (2009).

In addition to functional requirements, requirements engineers are also concerned with the consideration of non-functional requirements (Chung and do Prado Leite, 2009). Non-functional requirements are also known as quality requirements and *software qualities*. I use the latter term throughout my thesis. Examples for software qualities are performance, reliability, security, and privacy. Jackson (2000) states that software qualities are indicative statements about the system that still need to be refined to functional requirements. A goal of my thesis is to provide a method to systematically refine the software quality privacy into functional requirements.

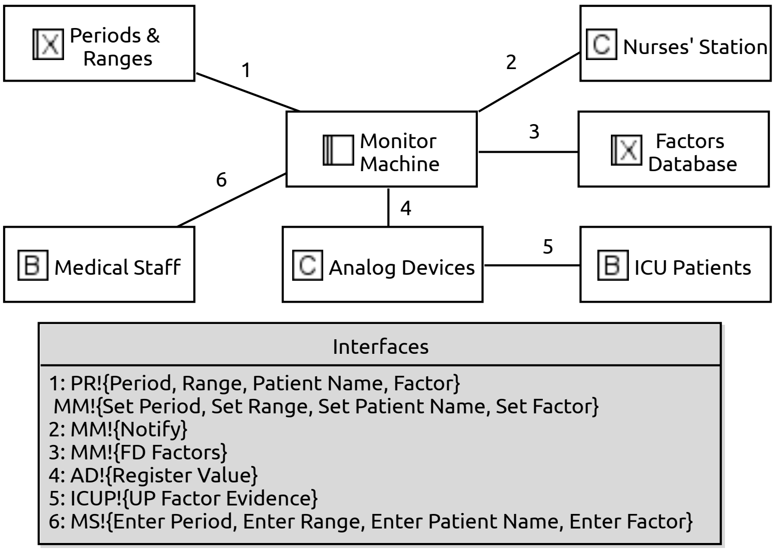
2.2. Problem Frames Approach

Together with his above introduced terminology, Jackson (2000) proposes the problem frames approach. The problem frames approach is a problem-based requirements engineering method. The goal of this method is to decompose the problem of building the software into *simple* subproblems.

2.2.1. Context Diagram

To provide an overview of the whole software development problem, Jackson proposes to create a *context diagram*. The context diagram shows the system consisting of the machine, its environmental domains, and the interfaces between them (similar to the part of Figure 2.1 with gray background). Figure 2.2 shows a context diagram for a patient monitoring system of an intensive care unit (ICU). In this case, the task is to develop a Monitor Machine (the symbol ▯ denotes that it is a machine) that receives factors, e.g, oxygen saturation, blood pressure, and heart beat frequency, from ICU Patients via Analog Devices and stores these factors in the Factors Database. Medical Staff shall be able to set periods for the monitoring and valid ranges for the patients factors using the Monitor Machine which manages these in the domain Periods & Ranges. If an ICU Patient's monitored factors are out of the specified ranges, then the Nurses' Station shall be notified.

Jackson distinguishes between *given* and *designed* domains. A given domain is an environmental domain that has to be taken as it is. In contrast, the structure and behavior of a designed domain, e.g., the machine itself, can be influenced or is also developed as part of the software project. Designed domains are highlighted using a vertical line in their icons. For example, the



Interfaces

1: PR!{Period, Range, Patient Name, Factor}
MM!{Set Period, Set Range, Set Patient Name, Set Factor}
2: MM!{Notify}
3: MM!{FD Factors}
4: AD!{Register Value}
5: ICUP!{UP Factor Evidence}
6: MS!{Enter Period, Enter Range, Enter Patient Name, Enter Factor}

Figure 2.2.: *Context diagram for an ICU system (based on (Jackson, 2000))*

domains Periods & Ranges, Factors Database, and Monitor Machine in Figure 2.2 are designed domains and the other are given domains.

Furthermore, Jackson distinguishes three types of domains the environment consists of. Causal domains (denoted by the symbol $\boxed{\text{C}}$) are technical devices (hardware, software, or both) whose behavior is predictable as it follows a known specification. Lexical domains (denoted by the symbol $\boxed{\text{X}}$) are physical representations of data, e.g., data bases and files. Biddable domains (denoted by the symbol $\boxed{\text{B}}$) are mostly humans. The characteristic of biddable domains is that we can only make assumptions about their behavior, in contrast to causal domains. In Figure 2.2, the Medical Staff and ICU Patients represent people and are hence biddable domains. The Analog Devices and the Nurses' Station are causal domains as these provide specified interfaces which can be used to receive the registered values and to notify nurses, respectively. The domains Periods & Ranges and Factors Database are both designed and lexical, i.e., they represent data and they are developed as part of the Monitor Machine.

As stated in the previous section, a phenomenon is controlled by exactly one domain. This control is expressed by the notation D!X where D is the abbreviation of the domain that controls the phenomena contained in the set X. Independent of the control, a domain can observe the phenomena contained in the set X if it is connected to the respective interface, which is denoted by a solid line between the respective domains (including the machine). The context diagram in Figure 2.2 contains six interfaces. The interface with the number 1 is between the Monitor Machine and the Periods & Ranges. It consists of the phenomena Period Range, Patient Name, and Factor controlled by the Periods & Ranges domain (with abbreviation PR) and the phenomena Set Period, Set Range, Set Patient Name, and Set Factor controlled by the Monitor Machine (with abbreviation MM). The interface specifies that the Monitor Machine can set the attributes period range, patient name, and factor (controlled phenomena of the machine) that are stored in the domain Periods & Ranges (controlled phenomena of the lexical domain). Note that the stored values controlled by the domain Periods & Ranges also belong to the interface. Hence, the machine is able to read (observe) these values.

The ICU example in Figure 2.2 shows that Jackson's problem frames approach allows to model the system with different levels of detail. For example, interfaces 1 and 6 prescribe which attributes shall be managed in the lexical domain Periods & ranges and that the Medical Staff shall be able to enter all these values. In contrast, the context diagram is not explicit about the factors that are monitored from the ICU Patients, e.g., oxygen saturation, blood pressure, and heart beat frequency.

Furthermore, it would be possible that the domain Analog Devices is ignored. That is, interface 5 could be drawn between the Monitor Machine and the ICU Patients. In this way, it would be ignored that this interface is actually realized by the causal domain Analog Devices. Domains that refine interfaces in the way the Analog Devices do are also called *connection domains.* Jackson (2000) states that connection domains may be omitted if they are not relevant for the software development problem to keep the context diagram simple and that they have to be made explicit if they introduce issues, such as unreliability. In the ICU example, the Analog Devices are made explicit as a connection domain, because they perform a transformation of the ICU Patients' factors to values the Analog Devices provided to the machine. Without the Analog Devices the Monitor Machine would not be able to observe the patients' factors.

Another issue with connection domains is that the *real world* domains could be omitted from the system description. For example, the *ICU Patients* could have been left out from Figure 2.2, because they have no direct connection to the Monitor Machine, which is the thing to be developed. In terms of Figure 2.1, this would mean that Domain_2 is omitted. Then, the requirement cannot be about the phenomena $\mathsf{p}_{\mathsf{r}_1}$ that the Stakeholder can observe, but is likely to be about the phenomena p_{d}. Consequently, the stakeholder's requirements are only sufficiently represented if the *real world domains*, e.g., the ICU Patients of Figure 2.2, are part of the context diagram. For a more detailed discussion on both discussed issues concerning connection domains and a methodology to systematically consider these issues, see the work of Ulfat-Bunyadi et al. (2016).

2.2.2. Problem Diagrams

The subproblems into which the overall problem is decomposed are represented in *problem diagrams.* Problem diagrams are (partial) projections of the software project's context diagram. That is, a problem diagram contains those domains and interfaces of the context diagram that are relevant for the considered subproblem, or in other words, relevant for the satisfaction of the considered requirement. Figure 2.3 shows a problem diagram for the following scenario from (Jackson, 2000):

> "*Lucy and John need a system to keep track of the many parties they give and the many guests they invite. They want a simple editor to maintain the information, which they call their party plan. Essentially the party plan is just a list of parties, a list of guests, and a note of who is invited to each party. The editor will accept command-line text input.*"

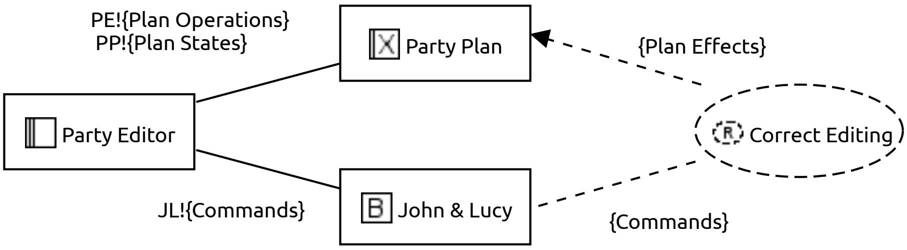


Figure 2.3.: *The party plan problem as problem diagram (based on (Jackson, 2000))*

The Party Editor is the machine to be built. It is connected to the domains Party Plan and John & Lucy. John & Lucy are a biddable domain and the Party Plan which *"[...] is just a list of parties, a list of guests, and a note of who is invited to each party[.]"* is a lexical domain (and designed). Both domains have an interface with the Party Editor.

In Figure 2.3, the interface between John & Lucy and the Party Editor consists of the phenomenon Commands controlled by John & Lucy (JL). This phenomenon represents all commands and operations that John and Lucy shall be able to issue and that the Party Editor shall react on. The interface between the Party Editor and the Party Plan consists of the Plan States controlled by the Party Plan (PP) and the Plan Operations controlled by the Party Editor (PE). The Plan States represent the data stored in the domain Party Plan that are accessable by the Party Editor. The Plan Operations are the commands that the Party Editor can issue in order to modify the Party Plan.

A dashed oval in a problem diagram represents a requirement. In Figure 2.3, the requirement is called Correct Editing. A requirement can *refer to* phenomena of domains, i.e., the requirement mentions them in the description of the situation in which the required behavior shall take place. Graphically, this relation is denoted by a dashed line without arrow head between the requirement and the domain of which phenomena are referred to. In Figure 2.3, the requirement refers to the Commands that John & Lucy may issue and that shall lead to the required behavior that the Party Plan is accordingly changed. To describe the desired behavior expressed by the requirement, the requirement *constrains* phenomena to occur or be as prescribed by the requirement. A constraint is expressed by a dashed line with an arrow head from the requirement pointing to the domain whose phenomena are constrained. In Figure 2.3 the requirement constrains the Plan Effects of the Party Plan, i.e., the effects on the Party Plan's state caused by the Commands issued by John & Lucy.

2.2.3. Problem Frames

To be more precise about what a simple subproblem is in the process of problem decomposition, Jackson proposes *problem frames* that represent known problem classes. Problem frames are a kind of pattern that can be instantiated for concrete software development problems. A problem frame is presented in the same way as a problem diagram, but all its elements are placeholders that can be instantiated. Figure 2.4 shows the simple workpieces problem frame. This problem frame generalizes the party plan problem of Figure 2.3. John & Lucy are generalized to a biddable domain called User, the Party Plan to the lexical domain Workpieces, the Party Editor to the machine Editing Tool, and the requirement Correct Editing is generalized to Command effects. The same applies to the phenomena sets annotated at the interfaces and requirement references. E1 represents the *events*[1] that the Editing Tool can issue to modify the Workpieces, *Y2* represents the *symbolic phenomena*[2] that are controlled by the Workpieces, E3 represents the events that the User can issue to instruct the Editing Tool to modify the Workpieces, and Y4 represents the symbolic phenomena whose values and states the requirement Command effects constrains.

In addition to events and symbolic phenomena, Jackson also introduces *causal phenomena*[3]. A set of causal phenomena is denoted by a C followed by a number, similar to the sets E1, Y2, E3, and Y4 in Figure 2.4

[1] Jackson defines events as: "A kind of *phenomenon*. An *individual* that is an occurrence at some point in time, regarded as atomic and instantaneous: for example, a keystroke."

[2] Jackson describes symbolic phenomena as: "*symbolic* phenomena are values, and truths and states relating only values. They are called symbolic because they are used to symbolise other phenomena and relationships among them. A symbolic state that relates values – for example, the data content of a disk record – can be changed by external causation, but we don't think of it as causal because it can neither change itself nor cause change elsewhere.

[3] Jackson describes causal phenomena as: "*causal* phenomena are events, or roles, or states relating to entities. These are causal phenomena because they are directly caused or controlled by some domain, and because they can cause other phenomena in turn. For example, a pulse event in a light unit may cause a state change in the Stop and Go lights".

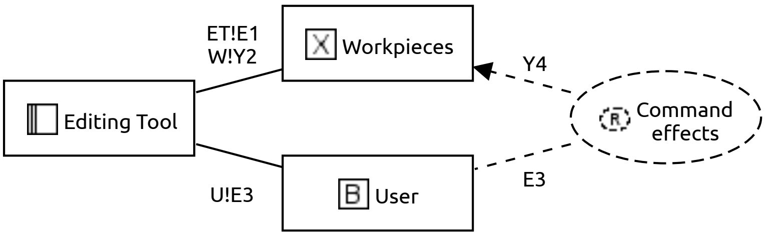


Figure 2.4.: *The simple workpieces problem frame (based on (Jackson, 2000))*

Jackson (2000) presents five basic problem frames and variations of them. Further problem frames are proposed in the works of Choppy and Heisel (2004), Wentzlaff and Specker (2006), Côté et al. (2008), and Hall and Rapanotti (2009).

2.2.4. Domain Knowledge Diagrams

In Section 2.1, I introduced the term domain knowledge, which consists of facts and assumptions. I propose in (Meis, 2014) to represent facts and assumptions in domain knowledge diagrams in a similar way as requirements are represented in problem diagrams. The main difference between a domain knowledge diagram and a problem diagram is that the domain knowledge diagram does not contain a machine that is responsible to address the contained facts and assumptions, as these are indicative statements.

Figures 2.5 and 2.6 show domain knowledge diagrams for an ICU system (cf. Section 2.2.1 and Figure 2.2). The upper assumption in Figure 2.5 constrains the Nurses' Station to notify the Nurses if the Monitor Machine instructs it to. The lower assumption constrains Nurses to recognize the notification provided by the Nurses' Station. These two assumptions are needed to reason that if the Monitor Machine issues the phenomenon Notify, then Nurses will recognize this notification.

The assumption in Figure 2.6 constrains the Analog Devices to Register Values to the Monitor Machine based on the measured UP Factor Evidences observed from the ICU Patients. This assumption is needed to reason that the phenomenon Register Value observed and later used by the Monitor Machine really corresponds to the phenomenon UP Factor Evidence that is actually relevant.

As shown in Figures 2.5 and 2.6, I use dashed ovals to represent assumptions, as also for requirements. The same applies to facts. To differentiate between all these kinds of statements, an assumption is decorated with the icon Ⓐ, a fact with the icon Ⓕ, and a requirement with the icon Ⓡ.

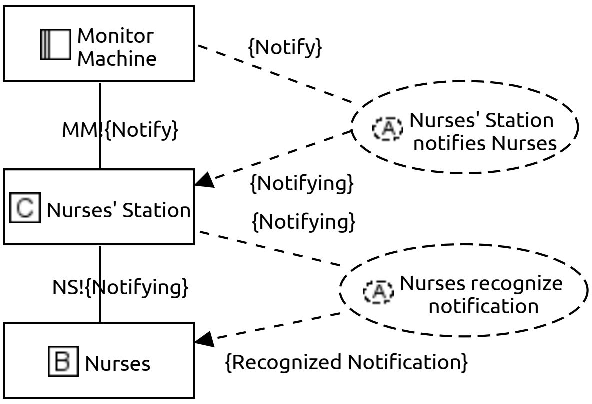


Figure 2.5.: *Assumptions concerning the Nurses' Station in the ICU system*

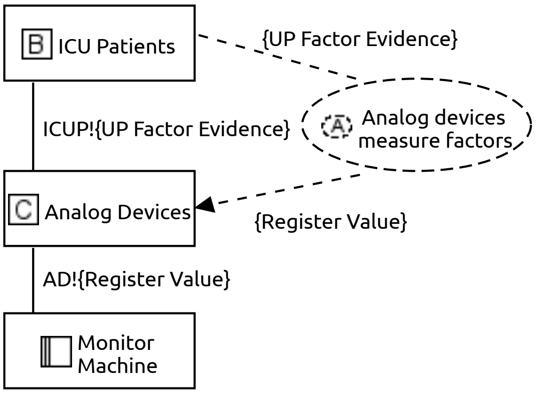


Figure 2.6.: *Assumptions concerning the Analog Devices in the ICU system*

2.3. Tool Support for the Problem Frames Approach

To assist the creation of the previously introduced problem diagrams, I developed tool support based on a formal metamodel for problem frames proposed by Hatebur et al. (2008) and the lessons learned from the UML-based problem frames tool UML4PF (Côté et al., 2011). The technology chosen for the developed tool is the Eclipse Modeling Framework (EMF) (Steinberg et al., 2008) and the Sirius Framework (Obeo and Thales, 2017). My developed tool support is called UDEPF (University Duisburg-Essen Problem Frames). UDEPF consists of an EMF metamodel and a graphical editor to create, modify, and provide views on instances of the metamodel. The ProPAn tool that assists the application of the ProPAn method (cf. Chapter 8) uses instances of the UDEPF metamodel as input.

The root element of my proposed EMF metamodel for problem frame models is called ProblemFrameModel (cf. Figure 2.7). A ProblemFrameModel contains exactly one ContextDiagram, an arbitrary number of *Statement*s, *Domain*s, and *StatementDiagram*s. In EMF, a filled diamond at an end of a relation describes that the instances of the class at which the diamond is drawn can contain instances of the class at the other end of the relation. An instance of a class can only be contained in one instance and all instances (except the root instance) should be contained in another element, otherwise they are not part of the same instance of the metamodel. The classes *Statement*, *Domain*, and *StatementDiagram* are abstract (denoted by italic font and gray background), i.e., it is not possible to create instances of these.

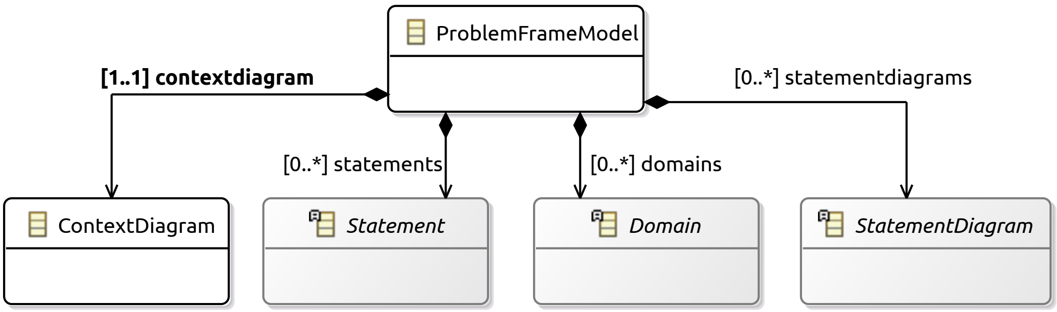


Figure 2.7.: *The root element of the EMF metamodel and its contained elements*

Most elements of a problem frame model have a name and a description for presentation and documentation purposes. This is realized using the abstract class *DocumentableElement* as shown in Figure 2.8.

Figure 2.9 shows the refinements of the abstract class *Domain*. The refinements are Jackson's

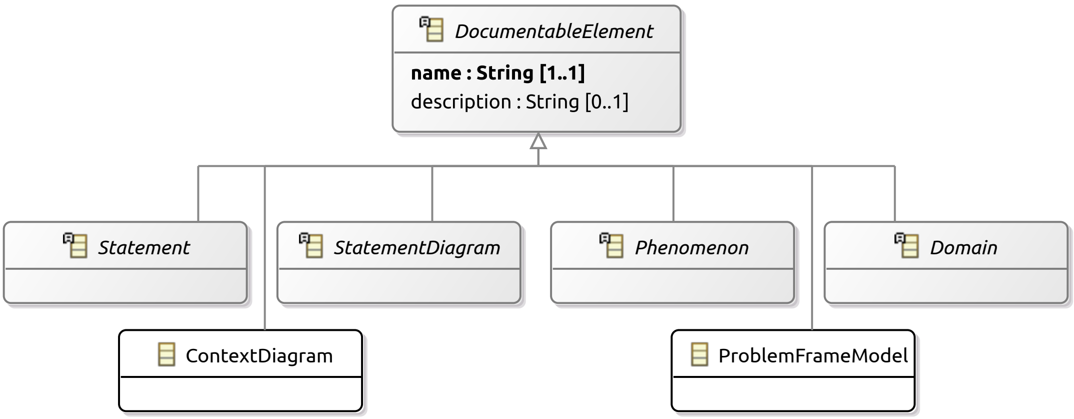


Figure 2.8.: *Abstract class* DocumentableElement *and its refinements*

domain types (cf. Section 2.2) BiddableDomain, CausalDomain, LexicalDomain, and Machine (which is considered as a refinement of the class CausalDomain). Additionally, a *Domain* has an abbreviation, is a designed domain or not, and is a connection domain or not.

As shown in Figure 2.9, a Domain contains a Phenomenon set, namely those phenomena which it controls. Additionally, there is a bi-directional relation between the classes Domain and Phenomenon that represents which phenomena a *Domain* observes and by which domains a *Phenomenon* is observed. As discussed before, Jackson distinguishes causal phenomena, symbolic phenomena, and events. Hence, we introduced the refinements CausalPhenomenon, SymbolicPhenomenon, and EventPhenomenon of the abstract class *Phenomenon*.

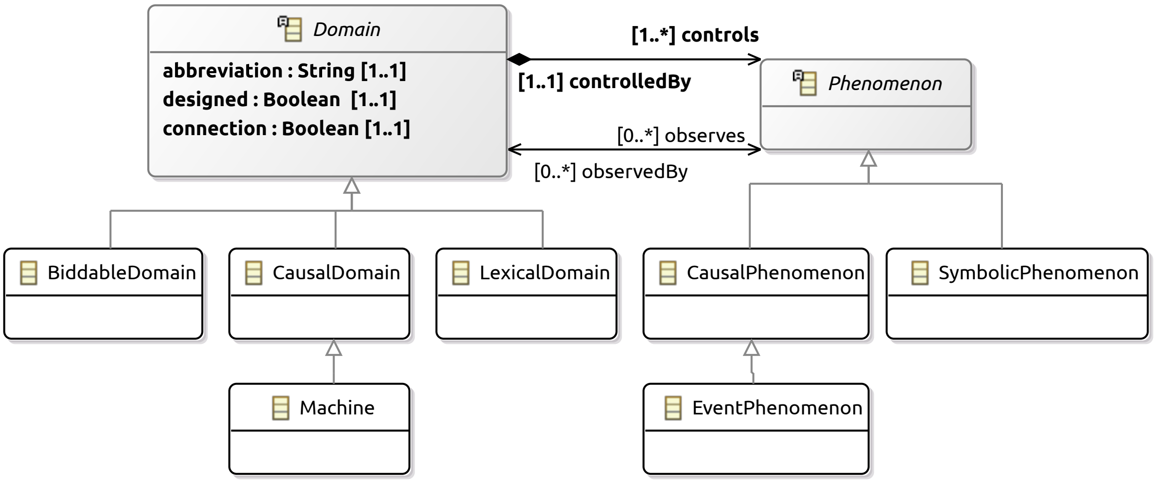


Figure 2.9.: *Focused view on the classes Domain and Phenomenon*

The refinements of the abstract class *Statement* are shown in Figure 2.10. These are Specification, SoftwareQuality, Requirement, and *DomainKnowledge*. The latter is further refined into Assumption and Fact. A Statement contains its *StatementReference*s which can be ConstrainsReferences and RefersToReferences. Each StatementReference targets a Domain and it contains a Phenomenon set of the targeted Domain that is referred to or constrained. The property that the contained phenomena belong to the respective domain (i.e., the phenomena are controlled by the domain) can be checked using OCL (Object Management Group, 2014) expressions that I introduce later.

Figure 2.11 shows that a ContextDiagram has a reference to the domains that shall be shown in it. That means, that not all domains contained in the ProblemFrameModel need to be contained

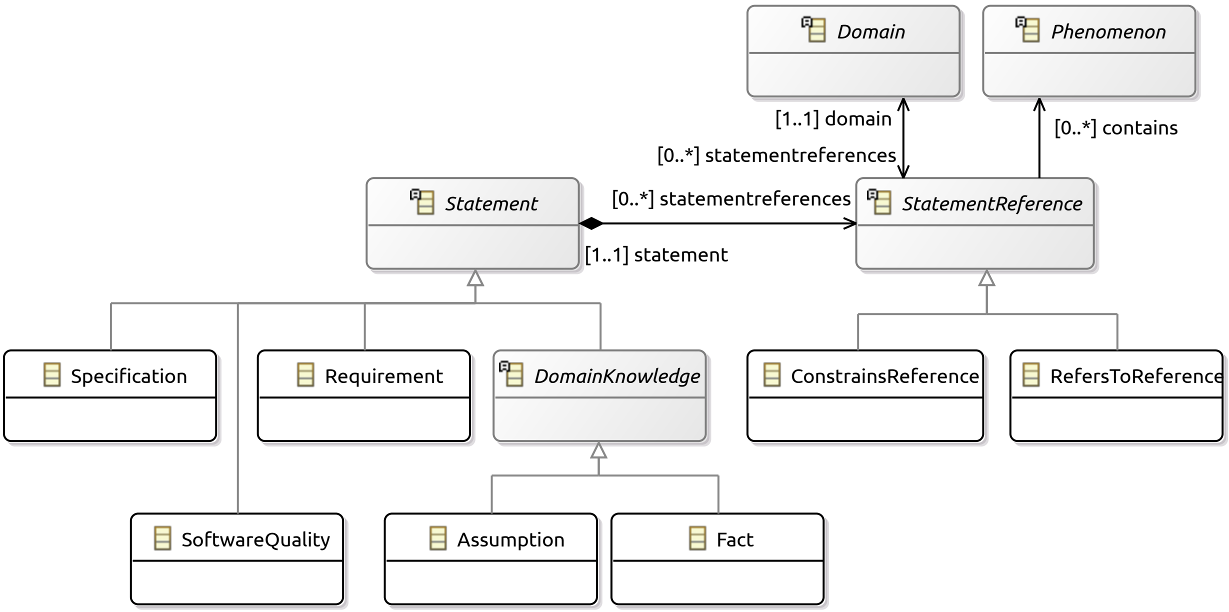


Figure 2.10.: *Focused view on the class Statement*

in the context diagram. Furthermore, a ContextDiagram contains a set of DomainInterfaces. The class DomainInterface represents the previously mentioned interfaces between domains. A DomainInterface connects at least two *Domain*s with each other and is annotated with at least one Phenomenon.

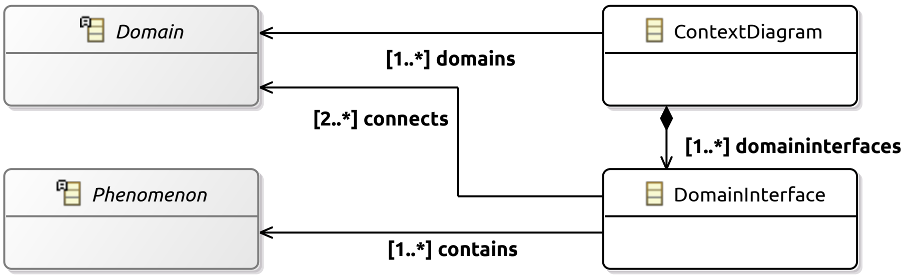


Figure 2.11.: *Focused view on the class ContextDiagram*

The abstract class *StatementDiagram* is refined into the classes ProblemDiagram and DomainKnowledgeDiagram as shown in Figure 2.12. Problem diagrams are introduced in Section 2.2 and domain knowledge diagrams are similar with the difference that they contain domain knowledge (cf. Figure 2.10) instead of a requirement and that they do not contain a machine that is responsible for the satisfaction of the domain knowledge, because a domain knowledge diagram only visualizes indicative statements. Similar to the class ContextDiagram (cf. Figure 2.11), a *StatementDiagram* has a reference to the domains it includes and the DomainInterfaces between these domains annotated with phenomena. Additionally, a StatementDiagram has a reference to the statements that it includes.

Note that *Statement*s and *Domain*s can occur in multiple *StatementDiagram*s and *Domain*s additionally in the ContextDiagram. Hence, Statements and Domains are contained in the ProblemFrameModel (cf. Figure 2.7) and are only referenced by ContextDiagrams (cf. Figure 2.11) and StatementDiagrams (cf. Figure 2.12).

There are integrity conditions for instances of the proposed problem frames metamodel that are not guaranteed by the metamodel itself. For example, for each combination of Phenomenon

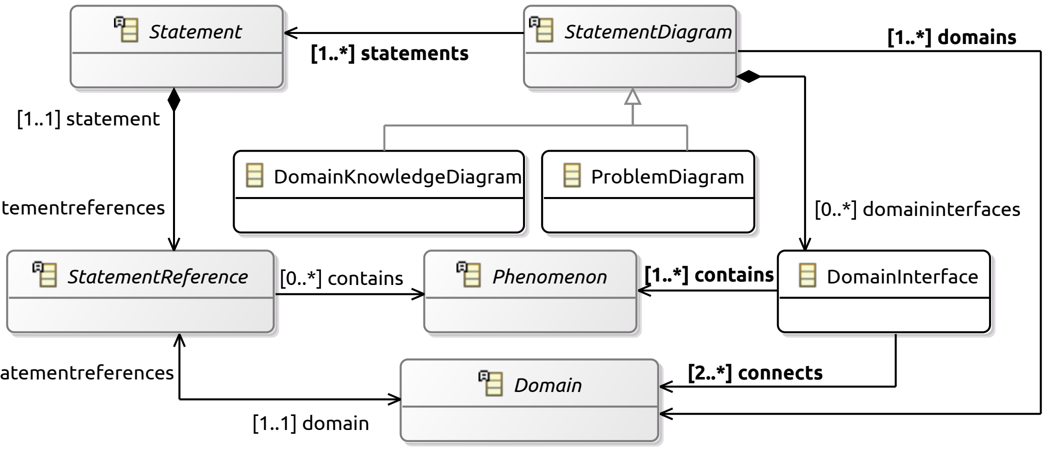


Figure 2.12.: *Focused view on the class StatementDiagram*

contained in an DomainInterface and Domain connected by the DomainInterface, the Phenomenon shall be observed or controlled by the Domain. This property is checked by the invariant for the class DomainInterface shown in Listing 2.1

Listing 2.1: *Invariant for the class DomainInterface*

```
context DomainInterface
inv: self.contains→forAll(p| self.connects→forAll(d|
       d.controls→includes(p) or d.observes→includes(p)))
```

Listings 2.2 and 2.3 show both another invariant related to the class DomainInterface. The listings only differ in their context. The context in Listing 2.2 is ContextDiagram and in Listing 2.3 *StatementDiagram*. Both check for the respective diagrams whether each contained instance of DomainInterface satisfies the condition that all domains that the interface connects are also associated to the respective diagram. Otherwise, a domain interface of a context or statement diagram could connect domains that are not all contained in the respective diagram.

Listing 2.2: *Invariant for the class ContextDiagram about the contained DomainInterfaces*

```
context ContextDiagram
inv: self.domaininterfaces→forAll(di| self.domains→includesAll(di.connects))
```

Listing 2.3: *Invariant for the class* StatementDiagram *about the contained DomainInterfaces*

```
context StatementDiagram
inv: self.domaininterfaces→forAll(di| self.domains→includesAll(di.connects))
```

For the *StatementReference*s contained in a *StatementDiagram*, I specified an invariant similar to the previous about DomainInterfaces. Listing 2.4 shows the OCL expression. It states that for each *StatementReference* of a statement contained in a statement diagram, the referenced domain shall be included in the set of domains associated to the statement diagram. Otherwise, a statement could be part of a statement diagram and refer to or constrain domains that are not contained in this diagram.

Listing 2.4: *Invariant for the class StatementDiagram about the contained StatementReferences*

```
context StatementDiagram
inv: self.statements.statementreferences→forAll(sr|
       self.domains→includes(sr.domain))
```

The invariant shown in Listing 2.5 is specified in the context of the class StatementReference. It states that the phenomena referenced by the respective instance of StatementReference are all controlled by the domain that the statement reference targets. That is, a statement reference may only refer to or constrain phenomena that are controlled by the domain which the statement refers to or constrains. If this is not the case, then the statement refers to or constrains the wrong domain. Note that Seater et al. (2007) allow that a statement also references phenomena that a domain observes. However, my stricter constraint is not a limitation, but may reveal that the actual domain of which phenomena shall be referenced is not yet contained the problem frame model.

Listing 2.5: *Invariant for the class StatementReference*

```
context StatementReference
inv: self.contains→forAll(p|self.domain.controls→includes(p))
```

The last invariants that I want to discuss are shown in Listings 2.6 and 2.7 . These invariants are concerned with the attribute connection of the class Domain. A connection domain shall only mediate between two or more domains, and shall not introduce additional functionality. This can be constraint by two necessary conditions that a connection domain has to satisfy. First, each phenomenon o observed by a connection domain (**self**) has to have a corresponding phenomenon c controlled by it (cf. Listing 2.6). Second, each phenomenon c controlled by a connection domain (**self**) has to have a corresponding phenomenon o observed by it (cf. Listing 2.7). In both cases, c has to be observed by a domain different from the domain controlling o, because otherwise the connection domain does not forward o to another domain. Note that these two constraints are not sufficient to check whether a domain is a connection domain. This is, because we may find phenomena o and c that satisfy the constraints, but that actually do not correspond to each other. To check this, we would need to add a relation between phenomena to the metamodel that allows to specify that two phenomena correspond to each other in the needed way.

Listing 2.6: *Invariant for class Domain checking that each phenomenon observed from a domain is forwarded to another domain*

```
context Domain
inv: self.connection implies
     self.observes→forAll(o|
       self.controls→exists(c|
         c.observedBy→excluding(o.controlledBy)→notEmpty()))
```

Listing 2.7: *Invariant for class Domain checking that each phenomenon forwarded to a domain corresponds to a phenomenon observed from another domain*

```
context Domain
inv: self.connection implies
     self.controls→forAll(c|
       self.observes→exists(o|
         c.observedBy→excluding(o.controlledBy)→notEmpty()))
```

Further semantic validation conditions exist for the proposed metamodel for problem frames, but these are not discussed in this thesis.

2.4. Privacy

Privacy is an interdisciplinary research field that originates from social science and legal science. Also psychologists are working in the field, especially in the perception of privacy. Starting in the late 1980s, privacy became a topic in computer science due to the forthcoming digitalization (cf. Section 1.1). The first works originate from the security research field. These works (e.g., Chaum (1992)) utilize techniques such as encryption and digital signatures to protect and enhance people's privacy in information systems.

2.4.1. Definitions of Privacy

The understanding and interpretation of privacy varies among the disciplines and also among literature of the same research field. Popular definitions are the following:

The right to be let alone This definition provided by Warren and Brandeis (1890) focuses on the freedom from intrusion. That is, an individual shall be protected against gossip, slander and other intrusions into his or her private sphere. Agre (1999) defines privacy as the freedom from unreasonable constraints on the construction of one's own identity. This definition adds to the *right to be let alone* the aspect of autonomy. This is, an individual shall be as free as possible in the construction of his or her own identity.

Information self-determination Westin (1967) defines privacy as *"the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others"*. Westin's definition emphasis that the control over personal information shall remain at the ones about whom this information is.

Privacy as confidentiality The definition of *privacy as confidentiality* (Gürses, 2010) has its roots in security research and its application to privacy. The focus of this definition is information secrecy, i.e., personal data shall be protected from disclosure to others.

In computer science, the consideration of privacy is focused on the protection of personal data processed by the systems under consideration. Hence, privacy is often also referred to using the term *data protection*. I avoid the term data protection in my thesis and use instead the term privacy to not exclude the dimensions of freedom from intrusion, self-determination, and autonomy from the scope of my research.

2.4.2. Privacy Terminology

Throughout my thesis, I mainly use the following terminology of the EU General Data Protection Regulation (European Commission, 2016). The central terms in the context of privacy are the following:

Data subject *"means an identified natural person or a natural person who can be identified, directly or indirectly, by means reasonably likely to be used by the controller or by any other natural or legal person, [...]."* (European Commission, 2016). Hence, the data subject is the individual whose data are processed by the software system. Data subjects can be users of the software-to-be, but need not to be users of it. The international standard ISO 29100 (ISO/IEC, 2011) uses the term *PII principle*, where PII is an abbreviation for personally identifiable information.

Personal data *"means any information relating to a data subject."* (European Commission, 2016) Along with the term personal data, the terms of *personal information (PI)* and *personally identifiable information (PII)* are widely used in the context of privacy and are mostly synonymous. ISO 29100 (ISO/IEC, 2011) defines PII as *"any information that (a) can be used to identify the PII principal to whom such information relates, or (b) is or might be directly or indirectly linked to a PII principal"*. To determine whether data are personal data, it has to be checked whether the data identify the data subject, or whether it is possible to link it (directly or indirectly) to the data subject.

This definition is difficult in cases where it is not clear whether the data under consideration allows to identify the person it belongs to or whether it is linkable to him or her. In 2006, AOL published pseudonymized search queries for academic research under the assumptions that these queries do not allow to identify the person that issued these. Barbaro and Zeller

(2006) revealed the identity behind one of the pseudonyms. This was possible, because the search queries allowed to infer the home location, age, and other personal data of the searcher. This case shows that all data of and about a person should be considered as personal data.

Gürses (2010) proposes instead the term *surveillance information* that she defines as follows. *"Surveillance information is data resulting from observations of the (digital or physical) world that will be collected, used, processed, distributed or deleted by the information system-to-be that is relevant for the different stakeholders privacy concerns."* (Gürses, 2010) This term is broader than personal data in the sense that it does not require a link between the data and the data subject.

In this thesis, I use the term personal data for any data that are related to a person, even if the data do not allow to identify the person to which the data are related, as long as it is not clear whether the data can be linked to an individual or allows to identify the data subject (cf. Fischer-Hübner (2001)).

Processing *"means any operation or set of operations which is performed upon personal data or sets of personal data, whether or not by automated means, such as collection, recording, organization, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, erasure or destruction."* (European Commission, 2016)

Controller *"means the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes, conditions and means of the processing of personal data; [...]."* (European Commission, 2016) This stakeholder is called *PII controller* in ISO 29100 ISO/IEC (2011). In addition to the PII controller, ISO 29100 introduces the stakeholder *PII processor*. These process personal data *"on behalf of and in accordance with the instructions of a PII controller"* ISO/IEC (2011).

Supervisory authority *"means a public authority which is established by a Member State in accordance with Article 46."* (European Commission, 2016) Article 46 states that supervisory authorities *"are responsible for monitoring the application of this Regulation and for contributing to its consistent application throughout the Union, [...]."* (European Commission, 2016)

Counterstakeholder is a person, organization, or the like against whom the data subjects' privacy shall be protected. Gürses et al. (2005) introduce the term counterstakeholder in the context of confidentiality requirements. They argue that the terms *adversary* and *attacker* that are used in most security literature are too narrow and imply malicious intentions. Counterstakeholders can be both malicious and non-malicious. That means, privacy may also be harmed if personal data are leaked to a non-malicious person or if a privacy breach is caused by a non-malicious person.

2.4.3. Privacy Principles

Privacy is not regulated homogeneously around the world. Some countries have privacy regulations applying to all kinds of software developments (e.g., EU (European Commission, 2016), Canada, and Japan), some countries have regulations affecting only specific sectors (e.g., USA), e.g., software for the domain of health or used by children, and others have no privacy regulation at all (e.g., Pakistan, Venezuela, and Iran). Due to the lack of binding privacy regulations, different standards and guidelines emerged that serve as a common ground for privacy-aware software development. The most recognized examples of these are the Fair Information Processing Principles (FIPPs) (US Federal Trade Commission, 1998; Brooks et al., 2017), the OECD guidelines (OECD, 1980), ISO 29100's privacy principles (ISO/IEC, 2011), and the Privacy-by-Design principles (Cavoukian, 2011). The EU General Data Protection Regulation (GDPR) also provides a set of principles similar to the previously mentioned.

From these sources, I extracted 15 privacy principles. These are:

Fairness The types and amount of personal data processed by a controller, the purposes they are used for, and the procedures with which the controller processes the personal data should be balanced with the actually needed types and amount of personal data for the purposes the data subjects provided their data, the expectations of the data subjects concerning the processing of their data and the protection needs for the processed personal data.

Compliance Controllers shall ensure that their processing of personal data is in compliance with applicable law, e.g., by conducting audits and privacy risk assessments.

Openness and transparency The processing of personal data shall be done in an open and transparent way. This means, that data subjects and other stakeholders (e.g., supervisory authorities) shall be informed about the procedures of the controller that concern the processing of personal data.

Access Controllers shall provide access to the personal data they process for the respective data subjects.

Individual participation Data subjects shall be able to exercise rights concerning their data, e.g., correction and deletion of their personal data, and objection to the processing of personal data.

Consent and choice Controllers should ask data subjects for their informed and explicit consent prior to processing their personal data. Data subjects shall have the choice for which purposes their data are processed.

Purpose legitimacy Controllers shall only process personal data they are allowed to process and that is legitimate for the purpose of the processing.

Purpose specification Controllers shall specify and explain the purpose for which they process personal data to the data subjects.

Use limitation Controllers shall only process personal data for the purpose for which they were originally collected, unless additional consent of the data subjects is collected.

Storage limitation Controllers shall retain personal data only as long as the personal data are necessary to be available for the purpose they were collected. After this, the personal data shall be destroyed.

Collection limitation Controllers shall only collect the personal data that were specified in the purpose specification and that are necessary for the specified purposes.

Data minimization Controllers shall ensure that the amount of personal data that is processed is minimal for the purpose it is processed for. This includes the deletion and generalization of personal data where this is possible.

Security Controllers shall implement sufficient safeguards to protect processed personal data based on security risk assessments.

Accuracy and quality Controllers shall ensure that the processed personal data are accurate, complete, up-to-date, adequate, and relevant for the specified purpose.

Accountability Controllers shall ensure that all principles are followed and shall document the actions that they have taken. This documentation shall include privacy policies, procedures, and practices.

Table 2.1 relates the principles of the different sources to each other. The first column contains the above introduced principles. The other columns represent one source of privacy principles and map the principles contained in this source to one or more of the principles contained in the first column. Note that the Privacy-by-Design (PbD) principles *Proactive not Reactive; Preventative not Remedial*, *Privacy as the Default Setting*, *Privacy Embedded into Design*, and *Full Functionality – Positive-Sum, not Zero-Sum* do not occur in the mapping, because they are to some extent related to other principles, but on a different level. Note also that I selected two versions of the FIPPs. First, the version proposed by the US Federal Trade Commission (1998) and second, a more recent version used by the National Institute of Standards and Technology (NIST) (Brooks et al., 2017).

From Table 2.1, we can see that ISO 29100 provides the most fine grained privacy principles and the best coverage of these. Because of this, I selected ISO 29100 as reference for privacy principles in Chapters 5 and 6. Note that the privacy principle fairness can be seen as a cross-cutting principle that is not explicitly listed by most sources as separate principle, but mentioned in the principles purpose legitimacy, data minimization, and use, storage and collection limitation. The fairness principle also cross-cuts the privacy principles openness and transparency, access, individual participation, consent and choice, purpose specification, security, and accuracy and quality. Fairness shall be used as a guidance for the implementation of the aforementioned principles.

2.4.4. Privacy Protection Goals

Complementary to the previously introduced privacy principles, Hansen et al. (2015) introduce six protection goals for privacy engineering. These protection goals include the three classical security goals *confidentiality*, *integrity*, and *availability* that are recognized as key elements of information security in the literature and standards, e.g., ISO 27000 (ISO/IEC, 2016).

In the context of privacy, confidentiality is about keeping personal data secret and to prevent the disclosure of personal data to counterstakeholders. Integrity is strongly related to the above mentioned principle of accuracy and quality. That is, it needs to be ensured that only correct personal data are processed and that the processing does not unintendedly modify the processed personal data. The protection goal availability targets the availability of the personal data for processing purposes the data subject agreed on and also the possibility for data subjects to have access to their personal data.

Hansen et al. complement the three security protection goals with three *new* protection goals. These are *unlinkability*, *transparency*, and *intervenability.*

Unlinkability can be seen as a kind of meta-confidentiality. That is, the relation between personal data, or the relation between data subjects and their personal data shall be kept secret. Unlinkability does not necessarily imply that the personal data themselves are kept secret, as long as they do not allow to create the undesired links. Unlinkability includes the privacy requirements *anonymity* (it shall not be possible to link personal data to their data subject), *pseudonymity* (a pseudonym is used to link the personal data to the data subject), and *undetectability* (counterstakeholders shall not be able to know about the occurrence of events or the existence of personal data) (Pfitzmann and Hansen, 2010).

Table 2.1.: *Overview of privacy principles*

Principle	ISO 29100	OECD	FTC FIPPs	NIST FIPPs	GDPR	PbD
Fairness					Lawfulness, fairness and transparency	
Compliance	Privacy compliance					
Openness and transparency	Openness, transparency and notice	Openness	Notice/ Awareness	Transparency		Visibility and Transparency
Access	Individual participation and access		Access/ Participation	Access and amendment		Respect for User Privacy
Individual participation		Individual participation		Individual participation		
Consent and choice	Consent and choice		Choice/ Consent			
Purpose legitimacy	Purpose legitimacy and specification			Authority		
Purpose specification		Purpose specification		Purpose specification and use limitation		
Use limitation	Use, retention and disclosure limitation	Use limitation			Purpose limitation	
Storage limitation					Storage limitation	
Collection limitation	Collection limitation	Collection limitation		Minimization	Data minimization	
Data minimization	Data minimization					
Security	Information security	Security safeguards	Integrity/ Security	Security	Integrity and confidentiality	End-to-End Security
Accuracy and quality	Accuracy and quality	Data quality		Quality and Integrity	Accuracy	
Accountability	Accountability	Accountability	Enforcement/ Redress	Accountability	Accountability	

Transparency is concerned with providing data subjects and supervisory authorities information about how and why their personal data are processed and the practices and procedures of the controller. The protection goal intervenability requires the controller to provide measures to empower data subjects to control whether, how, and for which purposes the controller processes them.

I investigate the privacy goals transparency and intervenability in more detail by refining these to high-level privacy requirements in Chapter 5 and Chapter 6, respectively. In Chapter 7, I present the taxonomy of privacy requirements that I use in my thesis. This taxonomy is a refinement of the six privacy protection goals proposed by Hansen et al. (2015) taking into account the privacy principles introduced in Section 2.4.3.

Chapter 3

State of the Art

In this chapter, I present the state of the art in privacy requirements engineering. I identified 40 privacy requirements engineering methods during a literature review that is explained in Section 3.1. To compare the state of the art methods, I define a high-level privacy requirements engineering method and comparison criteria in Section 3.2. The 40 privacy requirements engineering methods are introduced in Section 3.3 based on the high-level privacy requirements engineering method and the comparison criteria. I discuss the conclusions that can be drawn from the literature review and how the Problem-based Privacy Analysis (ProPAn) method, which I propose in this thesis, contributes to the state of the art in Section 3.4 Finally, I conclude this chapter in Section 3.5.

3.1. Literature Review

To collect the state of the art in privacy requirements engineering, I conducted a literature review. The inclusion criteria for documents found during the literature review are the following:

1. The document shall be concerned about privacy or data protection.
2. The document shall focus on the analysis or design phase of software development.
3. The document shall describe a method.

For each of the inclusion criteria, I defined a set of keywords. For the first inclusion criterion, I selected the keywords *privacy* and *data protection*. As the term *analysis* is too broad, i.e., it is not only used in the context of the analysis phase in software development, I decided to select instead the term *requirements* together with the term *design* for the second inclusion criterion. This should not be a limitation, because all privacy methods dedicated to the analysis phase of software development should also focus on privacy requirements. For the third inclusion criterion, I selected the terms *approach*, *methodology*, *method*, and *engineering* to find those documents that describe an approach, methodology, or engineering practice that can be followed.

At least one keyword of each inclusion criterion shall be contained in a document's title, abstract, or keywords. Otherwise, it is excluded from the literature review. Hence, the keywords of each inclusion criterion are combined with an *OR* and all disjunctions of the inclusion criteria are combined by an *AND*. The complete search term is *(privacy OR "data protection") AND (requirements OR design) AND (approach OR methodology OR method OR engineering)*.

I used Scopus[1] as search engine for an automatic search of relevant documents based on the above search term. I decided to use Scopus because: *"Scopus is the largest abstract and citation database of peer-reviewed literature: scientific journals, books and conference proceedings."*[2]. In

[1] `https://www.scopus.com` (accessed on 20 November 2017)

[2] Taken from `https://www.elsevier.com/solutions/scopus` (accessed on 29 March 2018)

comparison to other scientific search engines[3], Scopus is well accepted in the research community, covers most peer-reviewed literature, and only lists peer-reviewed literature. To further limit the search results returned by Scopus,

- I limited the subject area to computer science,
- I excluded conference reviews and editorials from the search results, because these should not propose themselves (scientific) contributions, and
- I limited the document language to English.

This search resulted in 5.178 documents[2]. To limit the number of documents that have to be assessed manually, I limited the search to documents that were published in 2015 or later. This search lead to 1.885 documents[2]. I read the title and abstract of these documents to decide whether they are really concerned with privacy (requirements) engineering methods. In this way, I obtained 77 documents that cannot be excluded without a reasonable doubt. Eight of these documents were papers of myself that I exclude from the literature research, because my thesis is based on these papers. Additionally, two papers were not accessible. Of the remaining 67 documents, I read the full text and finally decided whether they satisfy the above mentioned inclusion criteria. The final set of relevant documents contains 22 elements.

To identify relevant documents that are not indexed by SCOPUS or published earlier then 2015, I performed *backward snowballing* (Jalali and Wohlin, 2012). That means, I assessed in a second step the documents cited by the relevant documents found during the first part of the literature review. This process was iteratively repeated for the relevant documents found during the backward snowballing. During the backward snowballing I identified 65 documents that seemed to be relevant after reading title and abstract. From these one paper was not accessible and could, hence, not be considered. From the remaining 64 papers, 25 papers were identified as relevant related work after reading the full text.

I expect to have collected the most relevant documents about privacy requirements engineering methods that were published before 2015, because either these were extended or applied in the last three years, or they are cited by one of the relevant documents as related work. However, the threat to validity that I missed relevant privacy requirements engineering methods that were published before 2015 remains. This can happen if a method was not extended, applied or cited in the last three years, or if the method was not published in a peer-reviewed book, journal, or conference proceeding.

In total, I collected 47 documents as related work to my thesis. Table 3.1 shows a summary of the number of documents that I considered during the literature review. Note that during the analysis of the abstracts of the automated search result, which ended with 77 relevant documents, I did not exclude my own 8 documents and the 2 documents that were not accessible (Scopus provides the abstract of all listed documents). Furthermore, the 77 relevant documents after the *Abstract Analysis* contain also the 45 documents that are rejected during the *Full Analysis* and the 22 documents that are considered as relevant after the *Full Analysis*. The column *Total* provides the sum of the results of the *Full Analysis* and the *Snowballing*. In the next section, I introduce the comparison criteria that I use to compare the state of the art in privacy requirements engineering. A detailed comparison of the methods described in the 47 documents can be found in Section 3.3.

[3] `https://en.wikipedia.org/wiki/List_of_academic_databases_and_search_engines` (accessed on 29 March 2018)

[2] search conducted on 20 November 2017

Category	Abstract Analysis	Full Analysis	Snowballing	Total
Relevant	77	22	25	47
Not relevant	1808	45	39	84
Own	-	8	0	8
Not accessible	-	2	1	3

Table 3.1.: *Overview of the number of documents considered during the literature review*

3.2. Comparison Criteria

To compare the identified state of the art, I propose a set of comparison criteria. These criteria are based on a metamodel for privacy engineering methods proposed by Martín and del Álamo (2017) that extends the metamodel for software and systems development methodologies (SEMDM) (ISO/IEC, 2014). Additionally, I propose a high-level methodology for privacy requirements engineering that I derived from the identified state of the art and that is (partly) implemented the by state of the art privacy (requirements) engineering methods. This methodology is specified by instantiating SEMDM.

SEMDM provides a rich and extensible metamodel that allows to model software and systems development methodologies and instantiations of these, including processes, tasks, participating humans, used artifacts, and external resources. SEMDM proposes different diagrams to illustrate software and systems development methodologies of which I use three in the following three.

A *lifecycle diagram* allows to structure a software or systems development methodology into *phases* (represented as a pointing rectangle), which may contain *builds* (represented as double-pointed rectangles), *processes* (represented as rectangles with rounded corners), and *milestones* (presented as diamond shapes) (see Figure 3.1 on page 34). A build is characterized by the iterative application of the contained processes. Processes represent large-grained work units that can be refined into *tasks*. A milestone marks a point at which specific results shall be provided. The order of phases, builds, processes, milestones, and tasks is normally given by their arrangement. The reading direction is, as usual, left to right and top to bottom.

A *dependency diagram* provides the possibility to describe the dependencies between processes, *producers*, and *work products* (see Figure 3.2 on page 35). A producer is a human or tool that is responsible to perform a process. In Figure 3.2, I use only the producer kind *role* (depicted as a half ellipse) that represents a collection of responsibilities and that may be taken by different persons. Work products are artifacts that may be the input or output of processes. In Figure 3.2, I use the work product kinds *document* (represented as rectangle with dog-eared top right corner) and *model* (represented as rectangle divided into two parts by a horizontal line). In SEMDM, arrows from and to dashed boxes represent arrows pointing from all elements and to all elements inside the dashed box, respectively. Dependencies are represented as solid arrows and express that the starting end depends on the finishing end of the arrow.

A *process diagram* allows to show how processes are further refined into tasks (see Figure 3.3 on page 36). Tasks (depicted as ellipses) represent small-grained work units. The subtask relation is presented by a solid line between the process and its subtask. It can also be optional to perform a task during a process. This is depicted using a box with a tilde at the link.

3.2.1. Criteria Based on a Metamodel for Privacy Engineering Methods

Martín and del Álamo (2017) extend SEMDM with resource types that are considered by privacy engineering methods. They state that each method has an underlying conceptual model that defines what privacy means in the context of the method. For example, the privacy principles and the privacy protection goals introduced in Section 2.4.3, and Section 2.4.4, respectively, are different conceptual models. Martín and del Álamo call this resource type *Privacy Conceptual Model (PCM)*. The resource type *Privacy Normative Framework (PNF)* represents binding regulations or other obligations concerning privacy that must to be considered during the development. An example for a PNF is the EU General Data Protection Regulation (GDPR) (European Commission, 2016).

I use these two resource specializations to compare privacy (requirements) engineering methods based on their underlying taxonomy of privacy and privacy requirements (PCM), and the existence of normative frameworks that these methods explicitly target to support. Table 3.2 summarizes these criteria and provides elicitation questions and examples. The two additional criteria listed in Table 3.2 are introduced in the following section.

Table 3.2.: *Summary of evaluation criteria for privacy engineering methods*

Criterion	Elicitation Question	Examples
Privacy Conceptual Model (PCM)	On which conceptual model of privacy is the method based?	Privacy principles, privacy protection goals
Privacy Normative Framework (PNF)	Which normative frameworks for privacy are considered or supported?	GDPR
Requirements specification notation (Req. Not.)	Is a specific notation used to specify the functional requirements?	Textual, UML use case diagrams
System modeling language (Mod. Lang.)	Is a specific modeling language used to provide the system model?	Data flow diagrams, UML class diagrams

3.2.2. Criteria Based on a High-Level Privacy Requirements Engineering Methodology

To compare the different state of the art methods, I derived a high-level privacy requirements engineering methodology. The elements (i.e., processes documents, and models) of this methodology are realized differently in the state of the art. Figure 3.1 establishes the context of privacy requirements analysis in the Requirements Engineering phase of a software development process as SEMDM lifecycle diagram (ISO/IEC, 2014). The Functional Req. Analysis and the Privacy Req. Analysis are both represented as builds. Both contain two subprocesses. The milestones M0 and M1 form the input to and the output of the privacy requirements analysis, respectively.

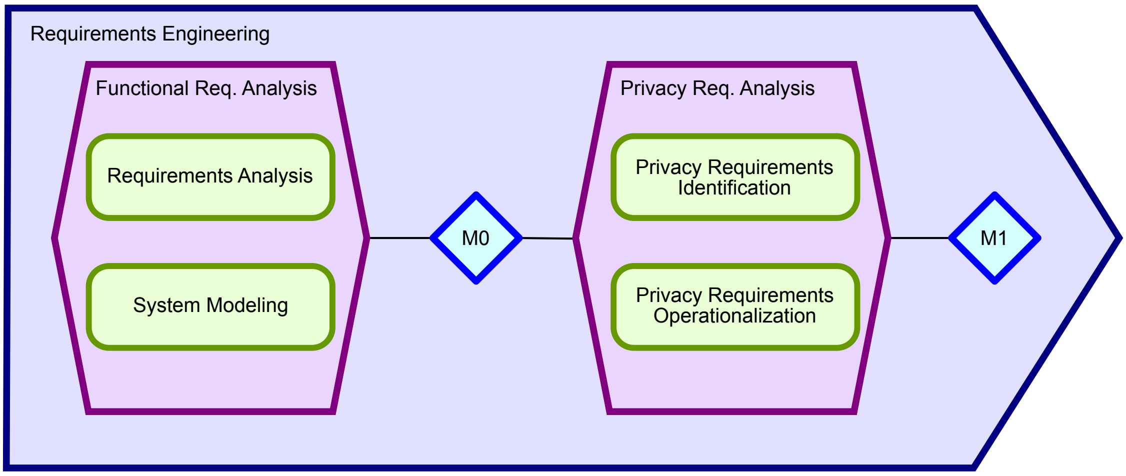


Figure 3.1.: *The Privacy Requirements Analysis inside the Requirements Engineering Phase*

The SEMDM dependency diagram in Figure 3.2 shows more information about the four processes shown in the previous lifecycle diagram. The Requirements Analysis and System Modeling processes of the Functional Req. Analysis both depend on the roles Requirements Engineer and Application Domain Expert. That means, these processes should be performed by people with these roles. An application domain expert is someone, who as special expertise in (parts of) the application domain in which the software-to-be shall be integrated into.

The Requirements Specification document depends on the Requirements Analysis, i.e., the document is created during this process. Similarly, the model System Model depends on the System Modeling process. The processes Privacy Requirements Identification and Privacy Requirements Operationalization both depend on the roles Requirements Engineer, Application Domain Expert, and Privacy Expert. Privacy experts can be, e.g., lawyers and data protection officers. End-users can be considered as both application domain and privacy experts that provide their specific (privacy) requirements. Both privacy related processes depend on the Requirements Specification and the System Model created in the first step. These two work products are produced during the functional requirements analysis resulting in milestone M0. On the other hand, these two work products also depend on the Privacy Requirements Identification and Privacy Requirements Operationalization processes, because these may extend, modify, and refine them. Additionally, the Risk Assessment document depends on the two privacy related processes. That means, that a risk assessment may be performed during these processes. The updated Requirements Specification and System Model, and the additional Risk Assessment document form the output of the Privacy Req. Analysis resulting in milestone M1.

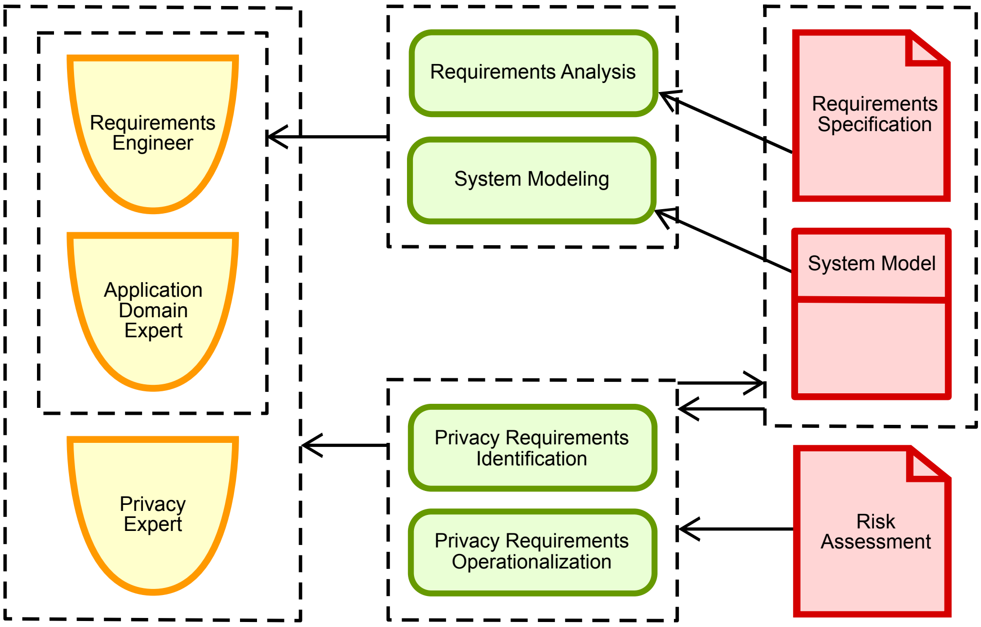

Requirements Engineer
Application Domain Expert
Privacy Expert
Requirements Analysis
System Modeling
Privacy Requirements Identification
Privacy Requirements Operationalization
Requirements Specification
System Model
Risk Assessment

Figure 3.2.: *Dependencies among the Producers, Processes, and Work Products of the high-level privacy engineering method*

Figure 3.3 shows the common tasks of the processes Privacy Requirements Identification and Privacy Requirements Operationalization in a SEMDM process diagram. The privacy requirements identification consists of three tasks. The optional task Extend req. specification and system model is concerned with an extension of the given requirements specification and system model with additional information that is needed to perform the privacy requirements analysis. During the Elicit privacy requirements task, the privacy requirements of the system-to-be are identified and documented. Privacy threats to which the system-to-be may be vulnerable to and that impose risks to the identified privacy requirements can optionally be elicited during the task Elicit privacy risks. The privacy requirements operationalization can be split into two tasks. First, privacy requirements can be refined to functional requirements during the task Refine privacy requirements. Second, privacy requirements can be operationalized by adding functional requirements that mitigate privacy risks during the optional task Treat privacy risks. These two tasks can be seen as different, but complementary, strategies to operationalize privacy requirements. Notario et al. (2015) call the first strategy *goal-oriented* and the latter *risk-based*. To avoid confusions between the goal-oriented strategy and goal-oriented requirements engineering, I call the first strategy *refinement-based*. This is, because privacy is refined into privacy requirements following this strategy. Similarly, I call the second strategy *prevention-based* to not confuse it with risk assessment methodologies and notations. If this strategy is followed, threats to privacy are tried to be prevented by identifying and treating them. Figure 3.3 also visualizes that the task Refine privacy requirements depends on the task Elicit privacy requirements (refinement-based strategy), and that the task Treat privacy risks depends on the task Elicit privacy risks (prevention-based strategy).

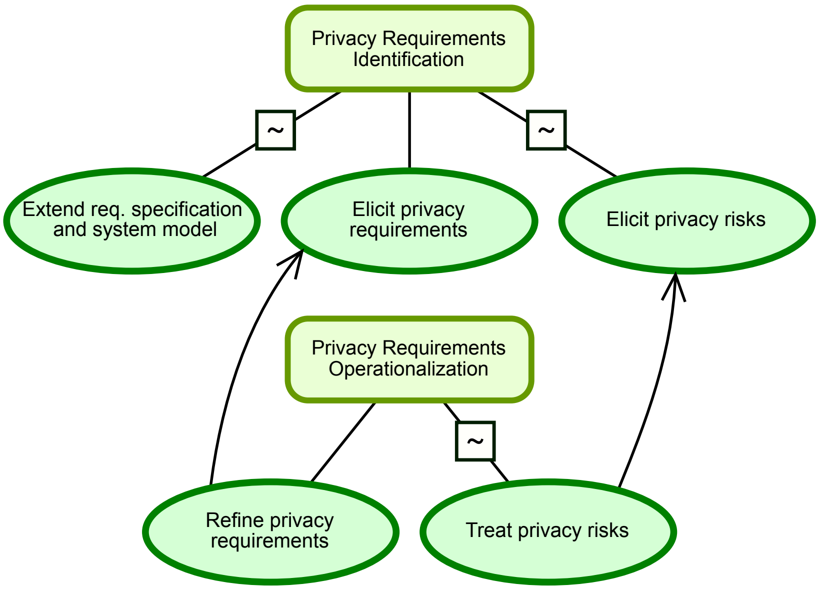

Privacy Requirements Identification
~
~
Extend req. specification and system model
Elicit privacy requirements
Elicit privacy risks
Privacy Requirements Operationalization
~
Refine privacy requirements
Treat privacy risks

Figure 3.3.: *Subtasks of the Privacy Analysis Processes*

Based on this high-level process for privacy requirements engineering, I suggest the following comparison criteria. First, the state of the art methods use different notations and languages to define the requirements specification and system model that serve as input to the method. Corresponding elicitation questions and examples are listed in Table 3.2. Second, the state of the art methods differently implement the five tasks identified for the privacy requirements analysis (cf. Figure 3.3). Hence, the methods can be compared based on whether and how they realize these tasks. More specifically, I propose to investigate for each of the five tasks, whether the task is considered, what the outcome the task is, which documents or models in which notation or language are used, how the task is supported and performed by the method, and whether tool support for this task exists. Table 3.3 lists these criteria together with elicitation questions and examples for possible answers.

Table 3.3.: *Summary of task-dependent evaluation criteria for privacy engineering methods*

Criterion for Task	Elicitation Question	Examples
Outcome	What is the result of the task, i.e., which work products are produced or updated?	Additional domain knowledge, privacy threats
Documents and Models (D & M)	Which documents and models are used, modified, or produced during the task, and which notations and languages are used?	Use cases (UML), Goal model (i*), Functional requirements (textual)
Technique	How are the tasks performed, i.e., is specific guidance provided are techniques suggested?	Questionnaires, templates
Tool	Is the task (partly) supported by a tool?	Modeling tool, automatic validation tool

3.3. State of the Art

The 47 identified documents present 40 privacy requirements engineering methods that I want to discuss in the following. To group the 40 privacy requirements engineering methods, I use two dimensions. The first dimension differentiates between methods that consider the process Privacy Requirements Operationalization and methods that do not consider this process. The second dimension differentiates between refinement-based, prevention-based, and combined privacy requirements engineering methods. A combined method contains elements of both, refinement-based and prevention-based, methods. Following these two dimensions. Table 3.4 shows how many methods were categorized to each of the six categories resulting from the two dimensions.

Table 3.4.: *Number of methods per combination of the two dimensions*

Operationalization	Refinement-based	Prevention-based	Combined
Not considered	6	2	2
Operationalization	12	10	8

In the following, I present the evaluation of the criteria presented in the previous section following the six categories of privacy requirements engineering methods. To improve the readability of the following summary tables (e.g., Table 3.5), I omit those criteria that all methods presented in the respective table do not consider. For example, Table 3.5 does not contain the lines for the criteria *PNF* and *Req. Not.*, because the listed methods do not consider any normative framework and requirements notation, respectively. If in an *Outcome* row the value *Not existing* occurs (e.g., in column **Perera et al.** in Table 3.5), then this means that the methodology given by the column does not support the task to which the *Outcome* belongs to.

3.3.1. Refinement-based Methods Not Considering Operationalization

I identified six methods that are refinement-based, but that do not consider the operationalization of privacy requirements. Tables 3.5 and 3.6 summarize the evaluated criteria for these six methods.

Antignac et al. (2016) propose an extension of data flow diagrams called privacy-aware data flow diagrams (PA-DFDs) (see column **PA-DFDs** in Table 3.5 on page 38). A DFD visualizes which entities, processes, and data stores are involved in the system-to-be and between which of these data flows. The proposed extension allows to model and make explicit elements and concepts mentioned in ISO 29100 (ISO/IEC, 2011). With this extension, existing DFDs can be annotated and extended to indicate personal data, data subjects, processors, controllers, purpose

Table 3.5.: *Overview of refinement-based methods not considering operationalization (part 1)*

Criteria	**PA-DFDs**	**Perera et al.**	**Kost et al.**
General Criteria			
PCM	Privacy principles	Privacy principles	Privacy principles
Mod. Lang.	Data flow diagrams (DFD)	Data flow (informal)	System model (e.g., UML)
Extend requirements specification and system model			
Outcome	Extended DFD with privacy annotations to indicate personal data, data subjects, processors controllers, purpose of personal data processing, and erasure of personal data	*Not existing*	System description mapped and transformed to domain-specific ontology
D & M	Privacy-aware DFD (PA-DFD)	-	Ontology (OWL)
Tool	-	-	Portégé to manually create ontology
Elicit privacy requirements			
Outcome	PA-DFD enhanced to include considerations based on ISO 29100‘s privacy principles	PbD principles relevant to the five data life cycle phases in IoT	Privacy principles and privacy constrains formalizing high-level privacy requirements, optionally a revised system model
D & M	Privacy-aware DFD (PA-DFD)	Table mapping PbD principles and life cycle phases	Ontology (OWL)
Technique	Guidelines for ISO 29100‘s privacy principles	List of 30 PbD principles for IoT	Manual formalization of privacy requirements, detection of constrain violations in the ontologies
Tool	-	-	Portégé to manually create ontology, Pellet for validation

of personal data processing, and erasure of personal data. Antignac et al. provide guidelines for all privacy principles of ISO 29100. These guidelines describe how the requirements implied by the privacy principles can be encoded using PA-DFDs.

Perera et al. (2016) propose a privacy-by-design (PbD) framework for assessing Internet of Things (IoT) applications and platforms (see column **Perera et al.** in Table 3.5 on page 38). The authors propose 30 PbD principles for the domain of IoT that they derived by combining the privacy design strategies of Hoepman (2014) with specific IoT concerns. To identify the privacy requirements of an IoT application or platform, Perera et al. propose to consider and evaluate their PbD principles for the five data life cycle phases in IoT, i.e., Consent and Data Acquisition, Data Preprocessing, Data Processing and Analysis, Data Storage, and Data Dissemination.

Kost et al. (2011) and Kost and Freytag (2012) propose the usage of ontologies for a privacy requirements analysis (see column **Kost et al.** in Table 3.5 on page 38). In their work, the authors focus on the Intelligent Transportation Systems (ITS) domain. Their approach is to create ontologies for the application domain (e.g., ITS) and for the privacy requirements that are derived from privacy principles (similar to ISO 29100's privacy principles). The ontologies are used to annotate and enhance the system model (e.g., a UML model). The annotated system model can than be validated for consistency to the ontologies. As modeling language for the ontologies, Kost et al. and Kost and Freytag (2012) use the Web Ontology Language (OWL)[4] and Portégé[5] as modeling tool. For the consistency validation, they use Pellet[6].

Table 3.6.: *Overview of refinement-based methods not considering operationalization (part 2)*

Criteria	Jutla et al.	Privacy Arguments	Bartolini et al.
General Criteria			
PCM	Privacy principles, privacy protection goals	Contextual integrity	Privacy regulation
PNF	-	-	GDPR
Req. Not.	Use cases (UML)	Problem diagrams	Process model (BPMN)
Mod. Lang.	-	Problem diagrams, event calculus	Process model (BPMN)
Elicit privacy requirements			
Outcome	Use case diagrams annotated with privacy controls	A set of privacy norms encoded in problem diagrams and event calculus	Tasks of the process model are annotated with elements of a data protection ontology
D & M	Use cases (UML)	Problem diagrams, event calculus	Ontology (OWL), Annotated process model (BPMN)
Technique	Use cases are extended with privacy controls	Add a machine responsible for addressing a privacy norm to every problem diagram with a functional requirement to which this norm is relevant	Data protection ontology
Tool	MS Visio extension for modeling	decreasoner tool to evaluate properties of event calculus expressions	Extension of Eclipse BPMN2 Modeler

Jutla et al. (2013) propose an extension to UML use cases that allows to annotate connections between actors and use cases with privacy controls, called privacy services (see column **Jutla et al.** in Table 3.6 on page 39). The ten proposed services Notice, Agreement, Consent, Access, Certification, Security, Interaction, Usage, Validation, and Enforcement are related to privacy principles and privacy protection goals. With these privacy services, privacy requirements can be expressed in UML use case diagrams. Jutla et al. implemented an extension for Microsoft Visio[7] that supports the creation of their annotations.

[4] `https://www.w3.org/OWL/` (accessed on 14 December 2017)

[5] `https://protege.stanford.edu/` (accessed on 14 December 2017)

[6] `https://github.com/stardog-union/pellet` (accessed on 13 December 2017)

[7] `http://www.microsoft.com/office/Visio` (accessed on 14 December 2017)

Tun et al. (2012) propose to express privacy arguments in the event calculus (Mueller, 2006), which is based on first-order logic (see column **Privacy Arguments** in Table 3.6 on page 39). The starting point in Tun et al.'s approach is a set of problem diagrams representing functional requirements and a behavioral description of them in event calculus. Based on this input, the authors propose to identify privacy norms following the contextual integrity framework (Barth et al., 2006). The concept of privacy as contextual integrity was originally proposed by Nissenbaum (2004). The main idea of contextual integrity is that there are positive and negative privacy norms that describe in which situations disclosure of personal data is acceptable and in which it is unacceptable. These situations are described by the context of the information disclosure (including its purpose) and the role of the actor to which information is disclosed. Tun et al. propose to encode the privacy norms as functional requirements and to add machines that address these to the problem diagrams in which they have to be considered. Additionally, the privacy norms are formalized in event calculus. Using the tool decreasoner (Mueller, 2006), it can be checked whether the functional requirements allow all positive norms, deny all negative norms, and to explore which privacy norms lead to the situation that specific information disclosure is allowed or denied.

Bartolini et al. (2017) propose a data protection ontology modeled in OWL (see column **Bartolini et al.** in Table 3.6 on page 39). This ontology represents the central elements of the EU General Data Protection Regulation (GDPR) (European Commission, 2016). This ontology can be used to annotate process models in the Business Process Model Notation (BPMN)[8]. In this way, privacy requirements can be encoded into the business process model. Bartolini et al. developed an extension of the Eclipse BPMN2 Modeler[9] that allows to annotate BPMN models with elements of their ontology.

3.3.2. Prevention-based Methods Not Considering Operationalization

I identified two methods that are prevention-based, but that do not consider the operationalization of privacy requirements. Table 3.7 summarizes the evaluated criteria for these two methods.

De and Le Métayer (2016) propose the Privacy Risk Analysis Methodology (PRIAM) (see column **PRIAM** in Table 3.7 on page 41). As input to their method they use a textual description of the functionalities of the system-to-be and a model of its interfaces and a data flow diagram (DFD). First, all relevant information about the information system, stakeholders, personal data, and already existing privacy controls is collected and added to the DFD. For the collection De and Le Métayer provide templates that describe the relevant attributes that need to be collected. Second, risk sources, privacy weaknesses, feared events, and privacy harms are collected using provided templates. This information is then connected using harm trees which are inspired from attack trees (Kordy et al., 2011). Using the harm trees the privacy risks of the system-to-be are evaluated.

Knirsch et al. (2015) propose a privacy risk assessment method for the smart grid domain (see column **Knirsch et al.** in Table 3.7 on page 41). They base their method on the data privacy taxonomy introduced by Barker et al. (2009). This taxonomy defines privacy requirements as points in a three dimensional space. The axes of this space are Granularity, Visibility, and Purpose of the personal data that are processed. Knirsch et al. start with a system model as data flow graph (DFG). The elements of this DFG are related to a smart grid specific ontology. Furthermore, the authors propose smart grid specific threat patterns that can be matched with the ontology elements and consequently to the DFG. In this way, privacy risks can be identified and evaluated. To create their models, Knirsch et al. use the European Smart Grid Architecture Model (SGAM) Toolbox (Dänekas et al., 2014).

[8] `http://www.bpmn.org/` (accessed on 14 December 2017)

[9] `https://www.eclipse.org/bpmn2-modeler/` (accessed on 14 December 2017)

Table 3.7.: *Overview of prevention-based methods not considering operationalization*

Criteria	PRIAM	Knirsch et al.
General Criteria		
PCM	Privacy threats	Data privacy taxonomy
Req. Not.	Functional specification (textual)	-
Mod. Lang.	Interfaces and data flow diagrams (DFDs)	Data flow graphs (DFGs)
Extend requirements specification and system model		
Outcome	Entities handling personal data, privacy stakeholders, privacy controls	*Not existing*
D & M	DFD, structured text	-
Technique	Attributes for stakeholders, and personal data	-
Elicit privacy risks		
Outcome	Harm trees presenting the relations between privacy harms, unwanted incidents, and vulnerabilities and there likelihood	Risk matrices containing risks identified using threat patterns
D & M	Harm trees (attack trees)	DFGs, Ontology (OWL), threat patterns (structured text)
Technique	Attributes provided for elements to be elicited, table with dependencies among the attributes, computation of likelihood	Matching of DFG elements with ontology and threat patterns
Tool	-	SGAM Toolbox for modeling

3.3.3. Combined Methods Not Considering Operationalization

I identified two methods that can be considered as both refinement-based and prevention-based methods, but that do not consider the operationalization of privacy requirements. Table 3.8 summarizes the evaluated criteria for these two methods.

P-SQUARE is proposed by Mead et al. (2011) as an extension of SQUARE (Mead and Stehney, 2005) for the consideration of privacy (see column **P-SQUARE** in Table 3.8 on page 42). To elicit privacy requirements, Mead et al. developed a questionnaire based on the privacy seals TRUSTe[10] and PrivacyMark[11], and the OECD guidelines (OECD, 1980). Based on the answers to these questions a tool called PRET (Miyazaki et al., 2008) suggests privacy requirements derived from the OECD guidelines, European, Japanese and American privacy regulation, the Common Criteria, W3C web services architecture requirements, and misuse cases. The derived privacy requirements and misuse cases are general and not tailored to the system-to-be. P-SQUARE does not require any specific requirements specification notation or system modeling language. However, the authors suggest to identify the assets, goals, and scenarios, e.g., use cases, before the questionnaire is answered.

Eddy is a formal language based on description logic to formalize and detail privacy policies proposed by Breaux and Rao (2013); Breaux et al. (2014, 2015) (see column **Eddy** in Table 3.8 on page 42). In Eddy, privacy requirements can be expressed that specify the operation mode (e.g., collection, storage, disclosure), the involved data, the source of the data, and the purpose for processing. The authors propose to use color codes to translate textual privacy policy statements

[10] `http://www.trustarc.com` (accessed on 12 December 2017)

[11] `http://www.privacymark.org` (accessed on 12 December 2017)

Table 3.8.: *Overview of combined methods not considering operationalization*

Criteria	P-SQUARE	Eddy
General Criteria		
PCM	Privacy principles, privacy regulation	Privacy principles
PNF	EU Directive, Japanese PIPA, US Laws	-
Req. Not.	Assets and goals (textual)	Privacy policy (textual)
Mod. Lang.	Scenarios	-
Elicit privacy requirements		
Outcome	List of privacy requirements based on the answered questionnaire	Privacy requirements that formalize privacy policies and specify an operation, the involved data, the source of the data, and the purpose for processing
D & M	List of privacy requirements	Privacy Requirements (Eddy/Description Logic)
Technique	Questionnaire	Translation of policy statements using color codes to Eddy and compile these to Description Logic
Tool	PRET tool	Eddy tool
Elicit privacy risks		
Outcome	List of misuse cases based on the answered questionnaire	Identify conflicting privacy requirements that lead to over-collection and repurposing
D & M	List of misuse cases	Privacy Requirements (Eddy/Description Logic)
Technique	Questionnaire	Automatic checking using the Eddy tool
Tool	PRET tool	Eddy tool

to the Eddy language. Based on the formalized privacy policies, the Eddy tool automatically identifies conflicts between these that may lead to over-collection or repurposing. The latter two are possible threats to the privacy principles collection limitation, purpose legitimacy, and purpose specification (cf. Section 2.4.3).

3.3.4. Refinement-based Methods Considering Operationalization

I identified twelve methods that are refinement-based and that consider the operationalization of privacy requirements. Tables 3.9-3.12 summarize the evaluated criteria for these twelve methods.

The Privacy Engineer's Manifesto is a book written by Dennedy et al. (2014) (see column **Manifesto** in Table 3.9 on page 43). The book provides an overview of privacy engineering practice and also describes a privacy engineering method. To model the system-to-be Dennedy et al. propose to use use cases and business activity diagrams, both presented using UML diagrams, and enhanced with additional information about the kind of data involved and the actors, called *metadata*. To identify so-called privacy use cases, the authors provide interpretation guidelines for the OECD privacy principles OECD (1980) and a list of five generic privacy use cases that can be instantiated. Based on the privacy use cases and a list of Privacy Enhancing Technologies (PETs), the business activity diagrams are enhanced by adding privacy related steps annotated

Table 3.9.: *Overview of refinement-based methods considering operationalization (part 1)*

Criteria	**Manifesto**	**Kung et al.**	**Yu and Cysneiros**
		General Criteria	
PCM	Privacy principles	Data minimization	Privacy principles
Req. Not.	Use cases (UML)	Functional requirements (textual)	Goal model (i*)
Mod. Lang.	Business activity diagram (UML)	-	Goal model (i*)
		Elicit privacy requirements	
Outcome	Privacy use cases with metadata	Refined functional requirements considering only minimal data needed	A set of softgoals refining privacy and privacy principles
D & M	Use cases (UML), Metadata (structured text)	Functional requirements (textual)	Goal model (i*)
Technique	Generic use cases, list of metadata, and interpretation guidelines	Data minimization	Catalog of soft goals refining privacy and privacy principles.
Tool	-	-	i* tool for modeling
		Refine privacy requirements	
Outcome	Enhanced business activity diagram with privacy related steps annotated with the related privacy principles and PETs	Techniques to implement data minimization	Tasks related to subgoals that operationalize these
D & M	Business activity diagram with annotations (UML)	Implementation techniques (textual)	Goal model (i*)
Technique	List of PETs and assurance checklist	-	Catalog of tasks operationalizing privacy goals, including contribution relations
Tool	-	-	i* tool for modeling

with the related privacy principles and PETs to them.

Kung et al. (2011) propose a PbD process that focuses on data minimization (see column **Kung et al.** in Table 3.9 on page 43). Their approach is to refine given functional requirements to consider only the minimal data needed to achieve the desired purpose. Then they state that implementation techniques have to be selected that implement data minimization and enforce it at run-time.

Yu and Cysneiros (2002) present a privacy requirements engineering method based on the goal modeling notation i* (see column **Yu and Cysneiros** in Table 3.9 on page 43). The authors propose a catalog of soft goals that refine the soft goal privacy. A soft goal is a goal that has no clear-cut criteria when it is satisfied or not, i.e., it needs to be refined. The catalog is based on the OECD privacy principles (OECD, 1980). To further refine these soft goals, Yu and Cysneiros provide a catalog of tasks that operationalize them and also provide contribution relations, because often mechanisms contribute (positively or negatively) to different privacy goals. Different i* modeling tools[12] are available that can be used to create the needed models.

Table 3.10.: *Overview of refinement-based methods considering operationalization (part 2)*

Criteria	**Privacy Design Strategies**	**Antón and He**	**Retro-Future**
	General Criteria		
PCM	Privacy principles, privacy regulation	Privacy regulation, privacy principles	Privacy regulations
PNF	GDPR, Privacy Shield	-	-
Req. Not.	High-level description (textual)	Goal and scenario model	-
Mod. Lang.	-	Goal and scenario model	-
	Elicit privacy requirements		
Outcome	Need to comply to GDPR or Privacy Shield is assumed	Defined purposes for processing of personal data as preconditions to tasks and specify privacy requirements as post conditions of tasks	Relevant privacy principles tailored to system-to-be
D & M	-	Goal and scenario model	Privacy principles (textual)
Technique	-	-	3 privacy principles and refinements
Tool	-	SMaRT tool for modeling goals and scenarios	-
	Refine privacy requirements		
Outcome	Suggested privacy tactics and patterns	RBAC model reflecting privacy requirements encoded in the goal model	Engineering approaches that support the implementation of relevant principles
D & M	List of privacy tactics and patterns (textual)	RBAC model (structured text)	Engineering approaches (textual)
Technique	Privacy design strategies are considered, related privacy tactics selected, and privacy patterns selected	Purpose hierarchies	Mapping of engineering approaches to privacy principles

Hoepman (2014) and Colesky et al. (2016) propose the usage of privacy design strategies extended with privacy tactics to derive the relevant privacy patterns that can be used to achieve a software architecture with a certain level of privacy protection (see column **Privacy Design Strategies** in Table 3.10 on page 44). Hoepman initially derived the privacy design strategies from ISO 29100 (ISO/IEC, 2011) and the EU Data Protection Directive (DPD) 95/46/EC (European Parliament, 1995) to translate the legal privacy requirements into a language more accessible to engineers. Colesky et al. refine the privacy design strategies by providing privacy tactics. These tactics were derived from a privacy pattern literature review and bridge the gap between the high-level strategies and more concrete privacy design patterns that help to realize

[12] http://istar.rwth-aachen.de (accessed 15 December 2017)

the strategies. Colesky et al. propose to identify the privacy protection needs from a perspective of compliance to the GDPR (European Commission, 2016) or other legal frameworks, e.g., the Privacy Shield agreement between the EU and the US (Colesky and Ghanavati, 2016). These needs are expressed as privacy design strategies and are further refined to privacy tactics. The tactics are linked to specific privacy design patterns and privacy enhancing technologies. In the collection of privacy patterns[13] on which Colesky et al. based their identification of the privacy tactics, it can be searched for patterns that address a specific privacy design strategy, but not yet based on the more fine-grained privacy tactics.

Antón and He (2003) base their method on a goal- and scenario-based approach (Antón, 1996) (see column **Antón and He** in Table 3.10 on page 44). That is, goals are operationalized by scenarios and goals may emerge from scenarios. Antón and He propose to define the purposes for the processing of personal data as preconditions to tasks in the goal model and specify privacy requirements as postconditions of tasks. These privacy requirements are then further refined into a role-based access control (RBAC) model. To create the RBAC model, the authors suggest to create purpose hierarchies that present relations between processing purposes. To model the goals and scenarios, Antón and He propose the Scenario Management and Requirements Tool (SMaRT).

Fisk et al. (2015) propose a privacy engineering approach, called Retro-Future, that is based on compliance to privacy regulations (see column **Retro-Future** in Table 3.10 on page 44). They propose three privacy principles and refinements of these. The privacy principles and their refinements need to be analyzed in the context of the system-to-be. To support the implementation of these principles, Fisk et al. provide a list of engineering approaches and technologies and relate these to the principles.

Guarda et al. (2017) and Ranise and Siswantoro (2017) provide a method to derive access control policies to comply to the EU Data Protection Directive (DPD) (European Parliament, 1995) (see column **Guarda et al.** in Table 3.11 on page 46). Their starting point is a set of message sequence charts (MSC) that describe the system-to-be. These are then annotated with read, write, and update rights for the processed data. Additionally, call-outs are added to the MSC at points where consent is needed for further processing. Guarda et al. and Ranise and Siswantoro express privacy requirements as First Order Logic (FOL) access control policies that they derived from the EU DPD. A bridge structure is used to connect the FOL access control policies with the MSC. The identified access control policies are then further refined to concrete access control policies. These can then be checked using existing access control policy tools, e.g., Armando et al. (2016) and Turkmen et al. (2015).

PriS (Kalloniatis et al., 2008; Diamantopoulou et al., 2017) considers the privacy goals authentication, authorization, identification, data protection, anonymity, pseudonymity, unlinkability, and unobservability (see column **PriS** in Table 3.11 on page 46). Kalloniatis et al. classify authentication, authorization, identification, and data protection as security related properties. Anonymity, pseudonymity, unlinkability, and unobservability form the privacy goals. The PriS method (Kalloniatis et al., 2008) starts with the business goals of the software system. Privacy goals are then related to these business goals. Based on this linkage the businesses processes related to the business goals can be enhanced using privacy process patterns. For the implementation of the privacy process patterns, a list of related implementation techniques (PETs) is provided. The PriS tool (Kalloniatis et al., 2009) can be used to create and annotate the goal model and business process model. Furthermore, it provides an overview of the relevant implementation techniques for the annotated goal and business process model.

Argyropoulos et al. (2016) combine PriS (Kalloniatis et al., 2008) and Secure Tropos (Mouratidis and Giorgini, 2007) (see column **PriS and Secure Tropos** in Table 3.11 on page 46). They enhance the PriS method by using the Secure Tropos notation for goal modeling and BPMN

[13] `http://privacypatterns.eu` (accessed on 27 October 2017)

Table 3.11.: *Overview of refinement-based methods considering operationalization (part 3)*

Criteria	**Guarda et al.**	**PriS**	**PriS and Secure Tropos**
General Criteria			
PCM	Privacy regulation	Privacy protection goals	Privacy protection goals
PNF	EU DPD	-	-
Req. Not.	-	Goal model	Goal model (Tropos)
Mod. Lang.	Message sequence charts (MSC)	Business process model	Goal model (Tropos), business process model(BPMN)
Extend requirements specification and system model			
Outcome	MSC annotated with read, write, and update rights and call-outs for consent	*Not existing*	*Not existing*
D & M	MSC	-	-
Elicit privacy requirements			
Outcome	High-level access control policies based on EU DPD and defined MSC	Refined goal model with annotated privacy goals	Refined goal model with annotated privacy goals and assigned implementation techniques
D & M	Bridge structure, access control policies (FOL)	Goal model	Goal model (Secure Tropos)
Technique	Formalization of EU DPD as FOL access control policies	Consideration of a given list of privacy goals for each organizational goal	-
Tool	-	PriS tool supporting annotation	Tropos tool for modeling
Refine privacy requirements			
Outcome	Concrete access control policies	Refined business processes with integrated privacy process patterns and suggested implementation techniques	Refined business processes with integrated privacy process patterns and assigned implementation techniques
D & M	Access control policies (FOL)	Business process model	Business process model (BPMN), Hybrid Reference Model to connect goal and process model
Technique	Policy compliance checking	Mapping of privacy process patterns to implementation techniques	Mapping of privacy process patterns to implementation techniques
Tool	Access control policy tools	PriS tool suggests implementation techniques	-

for business processes. Secure Tropos allows already to add implementation techniques as tasks to privacy goals. Hence, the authors allow this in their method in cases where implementation techniques are already prescribed or known. Furthermore, Argyropoulos et al. propose a model (called hybrid reference model) to formally connect the goal model to the business process model.

Table 3.12.: *Overview of refinement-based methods considering operationalization (part 4)*

Criteria	Bellotti and Sellen	Young	Degeling et al.
General Criteria			
PCM	Privacy as individual concept	Privacy policies	-
Req. Not.	-	Privacy policies	Process model (SeeMe)
Mod. Lang.	-	-	Process model (SeeMe)
Elicit privacy requirements			
Outcome	A list of feedback and control requirements	*Not existing*	A set of privacy related questions indicating privacy issues.
D & M	Feedback and control requirements (textual)	-	Annotations to process model (SeeMe)
Technique	Four categories concerning personal data processing with guiding questions	-	Guidelines, and predefined questions
Tool	-	-	SeeMe tool for process modeling and annotation
Refine privacy requirements			
Outcome	Mechanisms that implement feedback and control requirements	Functional requirements operationalizing privacy as specified in the given privacy policies	Enhanced process model addressing annotated privacy issues
D & M	Mechanisms (textual)	Functional requirements (textual)	Process model (SeeMe)
Technique	Eleven evaluation criteria to compare different mechanisms with each other	Systematic translation of policy text into requirements	Guidelines, workshop
Tool	-	-	SeeMe tool for process modeling and annotation

Bellotti and Sellen (1993) describe a high-level privacy engineering method for an ubiquitous computing environment (see column **Bellotti and Sellen** in Table 3.12 on page 47). They consider privacy as a concept that is perceived differently by each person and hence, has to be considered individually depending on the context and users of the system-to-be. They elicit privacy requirements in the form of feedback and control requirements, that provide transparency and intervenability to the end-users. To identify these Bellotti and Sellen (1993) propose four categories of personal data processing with questions that help to identify feedback and control requirements. Furthermore, the authors provide eleven evaluation criteria that can be used to evaluate different feedback and control mechanisms with each other to finally select appropriate mechanisms to implement the requirements.

Young (2011) proposes an approach to derive functional requirements from given privacy policies (see column **Young** in Table 3.12 on page 47). For this, she provides a systematic method to translate policy texts into functional requirements.

Degeling et al. (2016) and Lentzsch et al. (2017) propose a collaborative approach for privacy friendly system design (see column **Degeling et al.** in Table 3.12 on page 47). They suggest an iterative process for collaborative modeling that consists of a workshop phase and a reflection phase. During the first workshop, designers and users of the system-to-be create a system model. In the reflection phase, privacy experts annotate this system model with privacy related questions. The developers and users can then reflect on these questions. In the following workshop, the annotations are discussed by the users, developers, and privacy experts leading to an enhanced system model. Degeling et al. provide a collaborative web tool for the creation of SeeMee models, but it does not yet contain any privacy specific support.

3.3.5. Prevention-based Methods Considering Operationalization

I identified ten methods that are prevention-based and that consider the operationalization of privacy requirements. Tables 3.13-3.16 summarize the evaluated criteria for these ten methods.

Liu et al. (2003) propose a security and privacy requirements analysis method (see column **Liu et al.** in Table 3.13 on page 49). In this method, privacy is considered as a soft goal. The methodology, as presented in the paper, mainly considers privacy as confidentiality of personal data. Liu et al. assess privacy and security requirements based on i* models. In their method, the authors suggest to assess whether actors in the model may behave malicious and analyze their capabilities and whether these may lead to privacy vulnerabilities. Finally, countermeasures need to be integrated into the goal model that mitigate the identified vulnerabilities. A labeling algorithm provided by i* helps to assess whether vulnerabilities are already sufficiently considered or not. Different i* modeling tools[14] are available that can be used to create the needed models and to perform the labeling algorithm.

LINDDUN (Deng et al., 2011) is an adaption of the security threat analysis framework STRIDE (Howard and Lipner, 2006) to support privacy threat analyses (see column **LINDDUN** in Table 3.13 on page 49). LINDDUN differentiates between hard and soft privacy properties. As hard privacy properties they identify unlinkability, anonymity, pseudonymity, plausible deniability, undetectability, unobservability, and confidentiality. As soft privacy properties they consider consent awareness, and policy and consent compliance. Deng et al. propose to model the system-to-be using a data flow diagram (DFD). To limit the scope of the analysis, the authors suggest to add a trust boundary to the DFD. Data flows inside this trust boundary are not subject to the threat analysis. Then misuse case scenarios (Alexander, 2003) are identified. This task is based on 1) LINDDUN's privacy threats, which are negations of the before mentioned privacy properties, 2) a mapping between DFD elements and these privacy threats, and 3) threat tree patterns, which are similar to attack trees. Deng et al. provide threat tree patterns for each combination of privacy threat and DFD element. The privacy requirements of the system to be can be derived by translating the relevant privacy threats back to the privacy properties. Finally, mitigation strategies and techniques need to be selected. To support this task Deng et al. relate several PETs to LINDDUN's privacy properties.

Ahmadian and Jürjens (2016) present a model-based method to perform privacy analyses based on privacy level agreements (PLA) (see column **Privacy CARisMA** in Table 3.14 on page 50). Their method extends UMLSec (Jürjens, 2010), a security requirements analysis method based on UML models, and the CARiSMA tool [15], which supports the UMLSec method. The proposed approach starts with the definition of the customers' privacy preferences as PLAs. Ahmadian and Jürjens base their definition of the privacy preferences on the textual PLA outline proposed by the Cloud Security Alliance (2009), which is based on the EU DPD (European Parliament, 1995). Ahmadian and Jürjens formalize this textual PLA outline to a PLA metamodel and extend it by considering the General Data Protection Regulation (GDPR)

[14] http://istar.rwth-aachen.de (accessed 15 December 2017)

[15] https://rgse.uni-koblenz.de/carisma/ (accessed on 20 October 2017)

Table 3.13.: *Overview of prevention-based methods considering operationalization (part 1)*

Criteria	Liu et al.	LINDDUN
General Criteria		
PCM	High-level goal	Privacy protection goals, privacy threats
Req. Not.	Goal model (i*)	High-level description (textual)
Mod. Lang.	Goal model (i*)	Data flow diagram (DFD)
Extend requirements specification and system model		
Outcome	Attackers and their malicious intentions	Definition of a trust boundary to limit the scope of the analysis
D & M	Goal model (i*)	DFD
Technique	Consider actors as malicious	-
Tool	i* tool for modeling	-
Elicit privacy requirements		
Outcome	Privacy as softgoal associated to other goals	Privacy goals for DFD elements
D & M	Goal model (i*)	List of privacy goals (textual)
Technique	i* modeling guidelines	Mapping of LINDDUN threats to privacy goals
Tool	i* tool for modeling	-
Elicit privacy risks		
Outcome	Vulnerabilities to privacy softgoals based on the goal model	Misuse case scenarios
D & M	Goal model (i*)	Misuse cases (structured text)
Technique	Assess capabilities of attackers based on given goal model	Mapping of DFD elements to LINDDUN threats, threat tree patterns
Tool	i* tool for modeling	-
Treat privacy risks		
Outcome	Countermeasures that mitigate vulnerabilities	Suggested mitigation strategies and techniques
D & M	Goal model (i*)	Table relating strategies and techniques to misuse cases
Technique	Labeling algorithm to evaluate goal dependencies	Mapping of PETs to privacy goals
Tool	i* tool for modeling and labeling algorithm	-

(European Commission, 2016). A privacy preference concerns personal data of the customer and specifies for which *purpose* the data may be processed, in which *granularity* it may be processed, to whom the data may be *visible*, and how long the data may be *retained*, following the data privacy taxonomy of Barker et al. (2009). Then the system is described using UML models, such as sequence, activity, and class diagrams. The elements of these diagrams are annotated with their data processing properties and objectives. By matching the annotations of the system model with the defined privacy preferences, Ahmadian and Jürjens can automatically check the consistency of these and identify conflicts. Ahmadian et al. (2017) refine this approach and provide a profile to specify Attribute-Based Access Control (ABAC) (Hu et al., 2014) rules for UML elements representing the system-to-be. These annotations can also be checked for consistency

Table 3.14.: *Overview of prevention-based methods considering operationalization (part 2)*

Criteria	Privacy CARisMA	Yee
General Criteria		
PCM	Data privacy taxonomy	Privacy principles
PNF	GDPR	-
Req. Not.	-	-
Mod. Lang.	Class, sequence, activity diagrams (UML)	Data flow diagram (DFD)
Extend requirements specification and system model		
Outcome	Elements of the UML model are annotated with information concerning retention, granularity, visibility, and purpose	Personal information map (PIM) which is an extended DFD
D & M	Class, sequence, activity diagrams (UML)	PIM
Technique	UML profile defining stereotypes with attributes	-
Tool	UML tool for modeling	-
Elicit privacy requirements		
Outcome	Privacy level agreements that define the retention, granularity, visibility, and purpose of personal data	*Not existing*
D & M	Class diagrams (UML)	-
Technique	UML profile defining stereotypes with attributes	-
Tool	UML tool for modeling	-
Elicit privacy risks		
Outcome	Mismatches between system model annotations and privacy level agreements	A list of privacy risks related to elements of the PIM
D & M	List of mismatches (textual with pointers to the elements)	Table mapping risks to PIM
Technique	Matching of stereotype values	Risk questions
Tool	CARisMA tool for matching	-
Treat privacy risks		
Outcome	Updated annotated system model (including technologies)	Rated privacy risks and security measures
D & M	Class, sequence, activity diagrams (UML)	Table mapping risks to PIM, their ranking, and mitigations
Technique	UML profile for ABAC	Rate risks based on an attacker‘s risk, access, and cost to perform the attack, and on the damage the attack causes
Tool	UML tool for modeling	-

by the CARiSMA tool.

Yee (2017) presents a privacy risk assessment method for distributed software systems (see column **Yee** in Table 3.14 on page 50). First, a given DFD that describes the system-to-be is extended to a Personal Information Map (PIM), which adds information about the personal data that are processed. Then risk questions are considered to identify possible privacy risks based on the modeled PIM. Yee (2017) targets with his risk questions to elicit possible violation of the principles collection limitation, use limitation, and storage limitation (cf. Section 2.4.3) without making this explicit. The identified risks are then evaluated with respect to 1) the risk to the attackers safety (i.e., how risky is it for the attacker to perform an attack), 2) the ease of the attacker to access the software system, 3) the monetary costs for the attack, and 4) the damage caused by the attack. Finally, treatments are selected to mitigate the unacceptable risks. Yee suggests, e.g., technical measures, such as SSL, firewalls, and encryption to mitigate privacy risks.

Murukannaiah et al. (2016) propose a privacy engineering method for social applications, called Danio (see column **Danio** in Table 3.15 on page 52). They consider privacy as social expectations and norms that imply positive and negative sanctions for specific behavior. Their analysis of the system-to-be starts with a goal model described in Tropos. By assessing social relations among the actors and their expectations they identify norms and sanctions that they relate to the goal model. Then conflicting social expectations are elicited that lead to negative sanctions. These conflicts are then prioritized and addressed using an argumentation-based analysis (Murukannaiah et al., 2015).

Oetzel and Spiekermann (2014) present a systematic method for privacy impact assessments (see column **Oetzel and Spiekermann** in Table 3.15 on page 52). Their method starts with a textual project description, models that providing system, functional, and data views, and a view on the physical environment of the system-to-be. First, relevant privacy principles and privacy targets (i.e., refined privacy principles) are identified based on a given list of privacy principles and targets. Second, the privacy taxonomy of Solove (2006) is used to identify risks to the identified privacy targets based on a qualitative analysis and taking different perspectives of stakeholders into consideration. Finally, technical and non-technical privacy controls are selected to mitigate the identified privacy risks using a list of (high-level) controls related to the privacy targets.

Islam et al. (2010) propose a method to elicit security and privacy requirements from laws and regulations (see column **Islam et al.** in Table 3.15 on page 52). The authors use the goal modeling notation Secure Tropos to model the system-to-be. Privacy requirements are derived from legal texts using the privacy goal taxonomy of Antón and Earp (2004). For the assessment of risks to these privacy requirements, Islam et al. (2010) follow the Secure Tropos method (Mouratidis and Giorgini, 2007) and identify attackers, their intentions and attacks, and risks imposed by the attacks. The privacy requirements are then refined considering countermeasures to prevent the identified attacks.

Jensen et al. (2005) introduce a structured analysis framework for privacy (STRAP) (see column **STRAP** in Table 3.16 on page 53). They base their framework on goal models and provide analytic questions to identify vulnerabilities that block the satisfaction of goals in their models based on the four privacy principles Notice/Awareness, Choice/Consent, Security/Integrity, and Enforcement/Redress. To mitigate these vulnerabilities, mechanisms need to be added to the goal model or it has to be restructured to avoid the vulnerabilities. Jensen et al. provide examples for mechanisms and propose to evaluate these based on the criteria proposed by Bellotti and Sellen (1993).

Hong et al. (2004) consider privacy as an individual concept and do not consider a specific privacy taxonomy or framework (see column **Hong et al.** in Table 3.16 on page 53). The authors propose a catalog of privacy risk elicitation questions. The identified privacy risks are then evaluated based on the likelihood of their occurrence, the damage they cause, and the costs of possible countermeasures. Finally, mechanisms have to be selected to mitigate the identified privacy risks. To support this task, Hong et al. also provide a set of elicitation questions.

van Blarkom et al. (2003) present in their *Handbook of Privacy and Privacy-Enhancing Tech-*

Table 3.15.: *Overview of prevention-based methods considering operationalization (part 3)*

Criteria	Danio	Oetzel and Spiekermann	Islam et al.
General Criteria			
PCM	Social expectations and norms	Privacy principles	Privacy regulation
Req. Not.	Goal model (Tropos)	Project description	Goal model (Tropos)
Mod. Lang.	Goal model (Tropos)	System, functional, data, and physical views	Goal model (Tropos)
Elicit privacy requirements			
Outcome	Norms related to goals and tasks and sanctions for satisfying or violating norms	A list of relevant privacy principles and privacy targets (i.e., refined privacy principles)	Privacy goals derived from regulation and goal taxonomy
D & M	Norms and sanctions as quadruple	-	Goal model (Secure Tropos)
Technique	Assess social relations among actors and their expectations	Provided list of privacy principles and related privacy targets	Privacy goal taxonomy of Antón, mapping of legal text to goal model
Tool	-	-	Tropos tool for modeling
Elicit privacy risks			
Outcome	Conflicting social expectations that lead to negative sanctions	Protection demands for privacy targets and threats that could lead to a violation of these	Attackers, their intentions and attacks, and the risk imposed by the attacks
D & M	List of situations that lead to conflicts	-	Goal model (Secure Tropos)
Technique	-	Qualitative analysis, Stakeholder perspectives, Map Solove‘s privacy taxonomy to privacy targets	Goal-driven risk management and security attack scenarios
Tool	-	-	Tropos tool for modeling
Treat privacy risks			
Outcome	Prioritization of conflicting social expectations	Technical and non-technical controls that mitigate privacy risks	Refined privacy requirements considering countermeasures to prevent attacks
D & M	Norms and sanctions as quadruple	-	Goal model (Secure Tropos)
Technique	Argumentation-based analysis	A list of (high-level) technical and non-technical controls mapped to privacy targets	-
Tool	-	-	Tropos tool for modeling

nologies a prevention-based privacy engineering method (see column **Handbook of PETs** in Table 3.16 on page 53). They propose to identify the personal data processed by the system-to-be and threats to these that may lead to privacy risks. To identify threats, van Blarkom et al.

Table 3.16.: *Overview of prevention-based methods considering operationalization (part 4)*

Criteria	STRAP	Hong et al.	Handbook of PETs
General Criteria			
PCM	privacy principles	privacy as individual concept	Privacy principles, privacy regulation
PNF	-	-	EU DPD
Req. Not.	Goal model	-	-
Mod. Lang.	Goal model	-	-
Elicit privacy risks			
Outcome	Categorized and evaluated vulnerabilities blocking goals	Prioritized privacy risks	Identified personal data (assets) and threats to these that lead to risks
D & M	Goal model	privacy risks (textual)	-
Technique	Four privacy principles, analytic questions for these	Catalog of elicitation questions, evaluation of likelihood, damage, and cost of countermeasures	5 perspectives on threats: violation of regulations, created by system purpose, general solution, or chosen technology, and emanating from use situation
Treat privacy risks			
Outcome	Mechanisms mitigating vulnerabilities and/or restructured goal model	Mechanisms mitigating privacy risks	List of countermeasures reducing privacy risks
D & M	Goal model	Mechanisms (textual)	-
Technique	Examples for mechanisms, evaluation criteria for mechanisms	Catalog of elicitation questions	-

propose to consider five perspectives on threats, namely, 1) violation of privacy regulations, 2) threats emerging from the system's purpose, 3) its general solution, or 4) the chosen technology for realizing the system-to-be, and 5) privacy threats emanating from use situation. The identified risks have then to be mitigated by countermeasures.

3.3.6. Combined Methods Considering Operationalization

I identified eight methods that can be considered as both refinement-based and prevention-based methods, and that consider the operationalization of privacy requirements. Tables 3.17-3.19 summarize the evaluated criteria for these eight methods.

As a result of the EU project PRIPARE, Notario et al. (2015); Crespo et al. (2015) developed a privacy- and security-by-design methodology (see column **PRIPARE** in Table 3.17 on page 54). The starting point for their method is a textual project description. This project description is then used to identify the functional requirements, stakeholders, systems, and domains relevant for the system-to-be including their relations. Additionally, it is identified which personal data are processed by the system-to-be. Crespo et al. suggest to base this assessment on relevant legal documents and to perform a threshold analysis to decide on the scope of the privacy and security analysis. Based on the system model, a privacy principle checklist can be used to identify the relevant privacy principles that need to be considered and initial privacy controls

Table 3.17.: *Overview of combined methods considering operationalization (part 1)*

Criteria	PRIPARE	Oliver
General Criteria		
PCM	Privacy principles	Privacy regulation
Req. Not.	Project description (textual)	-
Mod. Lang.	Project description (textual)	Data flow diagrams (DFD)
Extend requirements specification and system model		
Outcome	Functional description, stakeholders, systems, domains, relations among these, personal data	Data flow diagram annotated with information about processed information, type of information, purpose, usage, and security classification
D & M	Model or textual	Data flow diagrams (DFD)
Technique	Legal assessment, threshold analysis	Ontologies for annotation types
Elicit privacy requirements		
Outcome	Relevant privacy principles and initial privacy controls	Privacy requirements defined for relevant triples consisting of the data classifier, a requirement detail level, and a requirement aspect
D & M	Privacy principles, privacy controls (textual)	Privacy requirements (textual)
Technique	Privacy principle checklist	Derive triples from annotated DFD
Elicit privacy risks		
Outcome	Misuse case scenarios	Privacy requirements mapped to risks they address and a list of unaddressed risks
D & M	Misuse cases (structured text)	Mapping (textual)
Technique	Mapping of DFD elements to LINDDUN threats, threat tree patterns	Risk Ontology
Refine privacy requirements		
Outcome	Privacy requirements that represent conformance criteria to the privacy principles	Mechanisms implementing privacy requirements
D & M	Privacy requirements (textual)	Mechanisms (textual)
Technique	Collection of conformance criteria for all privacy principles of ISO 29100	-
Treat privacy risks		
Outcome	Suggested mitigation strategies and techniques	Selection of mechanisms from risk ontology
D & M	Table relating strategies and techniques to misuse cases	Mechanisms (textual)
Technique	Mapping of PETs to privacy goals	Risk Ontology

may be selected. The identified privacy principles are then refined to privacy requirements that represent conformance criteria for these. To support this task, Crespo et al. provide a collection of conformance criteria for all privacy principles of ISO 29100 (ISO/IEC, 2011). For the risk elicitation and treatment, Crespo et al. propose to use LINDDUN (Deng et al., 2011) (see column **LINDDUN** in Table 3.13 on page 49).

Oliver (2016) proposes a privacy requirements engineering method that focuses on privacy regulation and uses data flow diagrams as system modeling language (see column **Oliver** in Table 3.17 on page 54). First, the DFDs are annotated with information about the personal data that are processed, the type of personal data, the purpose of processing and usage, and security classifications. Possible value types for these annotations are provided by ontologies. From the annotated DFD, triples consisting of a data classification, the requirement detail (policy, architecture, design, or code), and the processing purpose (called requirement aspect) are derived. Based on these triples, privacy requirements are formulated. These privacy requirements need then to be realized by specific mechanisms. Using a risk ontology, the privacy requirements are mapped to risks they address. For the unacceptable risks, mechanisms need to be selected to mitigate these.

Feth et al. (2017) propose an iterative user- and human-centered design process that focuses on security, privacy, and usability (see column **Feth et al.** in Table 3.18 on page 56). Their design process focuses especially on making the end-users aware of security and privacy protection mechanisms and to align these with the end-users' requirements and expectations. The authors consider the privacy protection goals of Hansen et al. (2015). The authors propose to express the requirements on the system in the form of use cases and scenarios. To model the end-users, they propose to make use of personas (Lidwell et al., 2010). For these, the relevant privacy protection goals are identified. Then, Feth et al. (2017) suggest to elicit privacy threats and risks that may harm the identified protection goals. Finally, privacy and security mechanisms, and corresponding awareness requirements need to be selected to refine the privacy protection goals and to treat the unacceptable privacy risks.

Gürses et al. (2011) propose an engineering approach to consider PbD (see column **Gürses et al.** in Table 3.18 on page 56). Their method focuses on end-users' privacy concerns and privacy regulations. Their method starts with a set of functional requirements that are refined to consider only the minimal data that are needed for the processing purpose (data minimization principle). Gürses et al. then propose to use a multilateral security analysis to select appropriate techniques to implement data minimization. Then attacker models are created from which threats, and risks to privacy can be derived. Mitigation techniques that address the unacceptable risks need finally to be selected.

Spiekermann and Cranor (2009) propose a privacy engineering method that is based on end-users' privacy concerns and privacy principles (see column **Spiekermann and Cranor** in Table 3.18 on page 56). The starting point is a textual project description and a differentiation between the user sphere (elements under control of the user), the joint sphere (partly controllable by the user, but mainly under the control of the service provider), and the recipient sphere (not controllable by the user). These three spheres form the three-layer privacy responsibility framework. As privacy requirements, Spiekermann and Cranor consider the principles of the Fair Information Practices (FIPs) (US Federal Trade Commission, 1998). To implement the privacy requirements, the authors suggest a list of PETs. As source of privacy risks, Spiekermann and Cranor suggest to consider user's privacy concerns and provide a list of potential concerns mapped to the user, joint, and recipient sphere. The authors provide a list of PETs and architectures to reduce the amount of personal data processed and the centralization of personal data processing. These PETs and architectures shall help to reduce the risks implied by the user's concerns.

Senarath et al. (2017) propose a user-centric privacy framework that they embed into the Unified Software Development Process (Kruchten, 2004) (see column **Senarath et al.** in Table 3.19 on page 57). The framework consists of seven steps covering the whole software development process. For each step, Senarath et al. provide guidance how privacy should be considered in a user-centric way by providing questionnaires asking questions from a user's perspective. First, a stakeholder evaluation report that summarizes privacy expectation, responsibilities, and poten-

Table 3.18.: *Overview of combined methods considering operationalization (part 2)*

Criteria	Feth et al.	Gürses et al.	Spiekermann and Cranor
General Criteria			
PCM	Privacy protection goals	Privacy concerns, privacy regulations	Privacy concerns, privacy principles
Req. Not.	Stakeholder descriptions (Personas)	Functional requirements (textual)	Project description (textual)
Mod. Lang.	-	-	Three-layer privacy responsibility framework
Elicit privacy requirements			
Outcome	Privacy goals important to stakeholders	Refined functional requirements based on minimal data (textual)	Fair information practices (FIPs) that have to be implemented
D & M	-	Functional requirements (textual)	-
Technique	Consider protection goals	Data minimization	List of FIPs
Elicit privacy risks			
Outcome	Privacy threats and risks	Attacker models, threats, and risks to privacy	User privacy concerns and threats to these
D & M	-	Attacker models, threats, and risks (textual)	-
Technique	Consider privacy protection goals	-	Mapping of privacy concerns to User, Joint, and Recipient Sphere
Refine privacy requirements			
Outcome	Privacy and security mechanisms, and corresponding awareness requirements	Techniques to implement data minimization	List of PETs to be implemented to address the relevant FIPs
D & M	-	Implementation techniques (textual)	-
Technique	-	Multilateral security analysis	Suggested PETs for FIPs
Treat privacy risks			
Outcome	Privacy and security mechanisms, and corresponding awareness requirements	Techniques mitigating privacy risks	List of PETs to reduce the amount of personal data processed and hence reducing privacy concerns
D & M	-	Implementation techniques (textual)	-
Technique	-	Multilateral security analysis	Suggested PETs and architectures

tial vulnerabilities of all stakeholders is created. Additionally, the processing of personal data is categorized following Barker et al. (2009)'s data privacy taxonomy. Using a user-survey, the users' privacy goals are identified and potential risks to the user's data are considered. Senarath

Table 3.19.: *Overview of combined methods considering operationalization (part 3)*

Criteria	Senarath et al.	SPACE	PMRM
General Criteria			
PCM	Data privacy taxonomy	Privacy protection goals, privacy threats	Privacy regulation, privacy principles
Req. Not.	-	Stakeholder requirements (textual)	Use cases (textual)
Mod. Lang.	-	-	Use cases (textual)
Extend requirements specification and system model			
Outcome	Stakeholder evaluation report, categorized personal data processing	A table listing the stakeholders‘ goals and the data involved in these	Identified personal data, their flow and relevant stakeholders
D & M	Stakeholder evaluation report (textual)	Table (structured text)	Data flows(textual)
Technique	Stepwise method and validation questions	Simple template	-
Elicit privacy requirements			
Outcome	A set of users‘ privacy goals	Security and privacy requirements added to the feared events table	Privacy controls and functionalities (called services)
D & M	-	Table (structured text)	privacy controls, and services (textual)
Technique	User-survey	-	List of 8 generic services
Elicit privacy risks			
Outcome	Privacy risks to user-data	Feared events table mapping stakeholders, threats, involved data, and feared events	Risks to privacy controls and services are elicited
D & M	-	Table (structured text)	Risks (textual)
Technique	-	Simple template	-
Refine privacy requirements			
Outcome	PETs that address the privacy goals and the privacy policy of the application	Requirements graph refining security and privacy requirements to technologies, and relating them to other requirements	Mechanisms that implement privacy functionalities (services)
D & M	-	Requirements graph	Mechanisms (textual)
Technique	For policy definition, stepwise method and validation questions	-	-
Tool	-	Requirement management tools	-

et al. propose a stepwise method to refine the user's privacy goals to privacy policies. Finally, PETs shall be selected to address the privacy goals and policies.

Vicini et al. (2016) propose the security- and privacy-by-design method SPACE (see column

SPACE in Table 3.19 on page 57). Their proposed method emphasizes *co-creation*, i.e., end-users and other stakeholders shall be involved in the design and analysis of the system-to-be. Vicini et al. (2016) base their privacy and security risk identification on LINDDUN (Deng et al., 2011) and STRIDE (Howard and Lipner, 2006), respectively. Vicini et al. provide simple templates to document stakeholders goals, which first have to be filled. Then feared events are documented together with the involved stakeholders, threats, and data in a feared events table. Security and privacy requirements that shall mitigate the feared events are then added to the respective table. Finally, technologies that realize the identified security and privacy requirements are selected and documented in a requirements graph. To model the requirements graph, Vicini et al. suggest different requirement management tools.

Drgon et al. (2016) present the Privacy Management Reference Model and Methodology (PMRM) (see column **PMRM** in Table 3.19 on page 57). Their methodology starts with textual use cases as system description. The personal data that shall be processed by the system-to-be, the implied flows of these personal data, and the relevant stakeholders are collected from these use cases. Then 8 generic privacy services (agreement, usage, validation, certification, enforcement, security, interaction, and access) are proposed that shall be considered to be integrated into the use cases. These 8 services are based on needs implied by privacy regulation and privacy principles. Then risks to the privacy services are elicited. Finally, mechanisms have to be selected that implement the relevant privacy services.

3.4. Discussion of the State of the Art

In this section, I summarize the conclusions that can be drawn from the literature review. I also discuss the shortcomings that exist in the state of the art that motivate the main contribution of this thesis, i.e., the ProPAn method (cf. Chapter 8). Tables 3.20 and 3.21 provide an overview of the considered privacy requirements engineering methods based on the previously introduced comparison criteria. In the following, I discuss the insights gained for the different comparison criteria.

3.4.1. Privacy Conceptual Models (PCMs)

Figure 3.4 shows the different identified conceptual models that I identified during the literature review and how often they are used by the different privacy requirements engineering methods (see also column **PCM** in Tables 3.20 and 3.21). Note that some methods are based on more than one conceptual model. Nearly half of the identified literature on privacy requirements engineering is based on privacy principles. One quarter considers privacy regulation taking privacy as compliance to laws, and another quarter is based on privacy goals or threats. The other conceptual models were each considered by one eighth of the methods.

The popularity of privacy principles can be explained by their purpose of making the concepts of and requirements implied by privacy more accessible to practitioners (in the computer science domain) and to define how privacy-by-design can be achieved. Additionally, privacy regulations, e.g., the GDPR (European Commission, 2016), often require that the application of privacy principles has to be demonstrated. Privacy regulations themselves are of cause considered by several privacy requirements engineering methods, because a big driving factor for considering privacy as software quality is the obligation to comply to privacy laws. Privacy goals or threats are often used by security related privacy engineering methods. These goals and threats are closer related to PETs that allow to implement and mitigate them, respectively. Privacy as a personal notation means that the individual data subjects' privacy understanding has to be captured and considered. This is the most open approach to privacy, but hardly manageable especially for a large and diverse set of data subjects. For example, data subjects' privacy

Table 3.20.: *Overview of all considered privacy requirements engineering methods (part 1)*

Privacy Method	PCM	Req. Not. & Mod. Lang.	Techniques	Strat.	Op.	Ext.	Meth.	Tool
Privacy CARisMA	DPT	System model	Catalogs	P	+	+	+	+
Liu et al.	PG	Goal model	Guidelines	P	+	+	+	+
SPACE	PG, PT	Informal	Templates	C	+	+	+	O
PRIPARE	PP	Informal	Guidelines	C	+	+	+	-
Senarath et al.	DPT	Informal	User-Suvey	C	+	+	+	-
Oliver	PR	DFD	Ontologies	C	+	+	-	-
Guarda et al.	PR	Process model	Logic	R	+	O	+	O
PMRM	PR, PP	Use cases	Guidelines	C	+	O	+	-
Yee	PP	DFD	Catalogs	P	+	O	+	-
LINDDUN	PG, PT	DFD	Guidelines, Attack trees	P	+	O	+	-
PriS	PG	Process and goal model	Catalogs	R	+	-	+	+
Islam et al.	PR	Goal model	-	P	+	-	+	O
Yu and Cysneiros	PP	Goal model	Catalogs	R	+	-	+	O
Pris and Secure Tropos	PG	Process and goal model	Catalogs	R	+	-	+	O
Antón and He	PP, PR	Scenarios, Goal model	-	R	+	-	+	O
Feth et al.	PG	Informal	-	C	+	-	+	-
Oetzel and Spiekermann	PP	Informal	Catalogs	P	+	-	+	-
Danio	IC	Goal model	-	P	+	-	+	-
Manifesto	PP	Use cases, Process model	Guidelines	R	+	-	+	-
Young	Other	Privacy policy	Translation	R	+	-	+	-

DPT:	Data protection taxonomy	PG:	Protection goals	PT:	Privacy threats
PP:	Privacy principles	PR:	Privacy regulations	IC:	Individual concept
P:	Prevention-based	R:	Refinement-based	C:	Combined
+:	Existing	O:	Partially existing	-:	Not existing

perception may change due to a growing awareness of privacy concerns. The conceptual model of the data privacy taxonomy and the other models are related to the privacy principles and privacy regulations.

Hence, we can differentiate between approaches that more focus on privacy in terms of a *real*

Table 3.21.: *Overview of all considered privacy requirements engineering methods (part 2)*

Privacy Method	PCM	Req. Not. & Mod. Lang.	Techniques	Strat.	Op.	Ext.	Meth.	Tool
Degeling et al.	Other	Process model	Guidelines	R	+	-	O	O
Handbook of PET	PP, PR	Informal	Catalogs	P	+	-	O	-
STRAP	PP	Goal model	Guidelines	P	+	-	O	-
Bellotti and Sellen	IC	Informal	Guidelines	R	+	-	O	-
Kung et al.	PP	Informal	-	R	+	-	O	-
Spiekermann and Cranor	IC, PP	Informal	Catalogs	C	+	-	-	-
Gürses et al.	IC, PR	Informal	Guidelines	C	+	-	-	-
Hong et al.	IC	Informal	Catalogs	P	+	-	-	-
Privacy Design Strategies	PP, PR	Informal	Catalogs	R	+	-	-	-
Retro-Future	PR	Informal	-	R	+	-	-	-
PRIAM	PT	DFD	Attack trees	P	-	+	+	-
PA-DFD‘s	PP	DFD	Guidelines	R	-	O	O	-
Kost et al.	PP	System model	Ontologies	R	-	O	O	-
Eddy	PP	Privacy policy	Logic	C	-	-	+	+
P-SQUARE	PP, PR	Scenarios	Questionnaire	C	-	-	+	+
Perera et al.	PP	Informal	Guidelines	R	-	-	+	-
Knirsch et al.	DPT	DFD	Ontologies, Patterns	P	-	-	O	O
Privacy Arguments	Other	Problem diagrams, Logic	-	R	-	-	-	+
Bartolini et al.	PR	Process model	Ontologies	R	-	-	-	O
Jutla et al.	PP, PG	Use cases	-	R	-	-	-	O

DPT:	Data protection taxonomy	PG:	Protection goals	PT:	Privacy threats
PP:	Privacy principles	PR:	Privacy regulations	IC:	Individual concept
P:	Prevention-based	R:	Refinement-based	C:	Combined
+:	Existing	O:	Partially existing	-:	Not existing

world requirement (cf. Section 2.2), i.e., privacy regulation and privacy as individual concept, and approaches that consider privacy goals and threats that are closer to the *machine* and PETs. Privacy principles serve as a bridge between these two by providing guidelines that may be followed to reach compliance to legislation and to satisfy data subjects' privacy requirements.

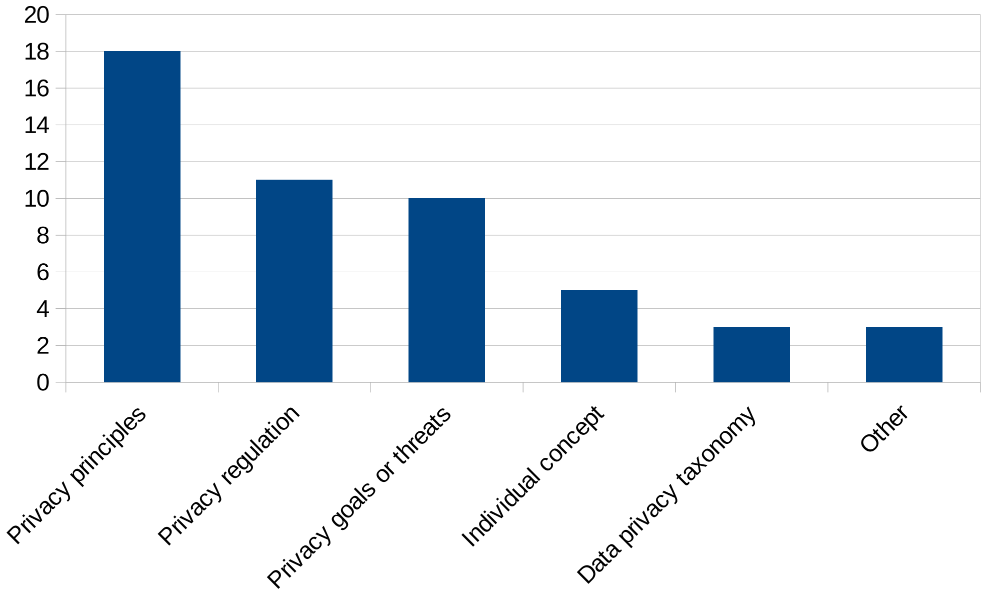


Figure 3.4.: *Privacy conceptual models identified during the literature review*

Privacy principles also formulate requirements that can be expressed in terms of data protection goals. These relations are discussed in more detail in Chapter 7.

In my thesis, I mainly consider the privacy protection goals proposed by Hansen et al. (2015) (cf. Section 2.4.4) as conceptual model for privacy. I also consider privacy principles and privacy regulation to refine the privacy goals transparency (see Chapter 5) and intervenability (see Chapter 6). Different privacy principles are also considered as guidelines during the proposed ProPAn method (see Chapter 8). In this way, the ProPAn method takes into account relevant privacy regulations and principles, but is not limited to the considered regulation and principles. Additionally, the consideration of the privacy protection goals has the advantage of being more tailored to the operationalization of privacy requirements and the selection of PETs.

3.4.2. Privacy Normative Frameworks (PNFs)

Only six of the 40 privacy requirements engineering methods (Bartolini et al., 2017; Mead et al., 2011; Hoepman, 2014; Colesky et al., 2016; Guarda et al., 2017; Ahmadian and Jürjens, 2016; van Blarkom et al., 2003) explicitly consider a specific privacy normative framework. All of them consider either the GDPR or the former EU DPD. Mead et al. (2011) additionally consider the Japanese PIPA and US laws, and Colesky et al. (2016) the Privacy Shield agreement between the United States and the European Union.

In this thesis, I also focus on the GDPR as source for privacy requirements. However, my privacy requirements taxonomy presented in Chapter 7 and the ProPAn method introduced in Chapter 8 are expected to be adaptable to other legislations as well.

3.4.3. Notations and Languages (Req. Not. & Mod. Lang.)

The different requirements specification notations and system modeling languages used by the methods to describe requirements, the system, domain knowledge, etc., are summarized in Figure 3.5 including the number of usages (see also column **Req. Not. & Mod. Lang.** in Tables 3.20 and 3.21). Note that some methods use more than one of these notations and languages. More than one third of the methods does not use any modeling approach or formal notation, but relies on textual documentation. The text-based approaches have the advantage that it is easier to adapt them in practice, because the users of the method do not have to learn a specific modeling notation or formal language. The disadvantage is that automation and consistency checks are hardly possible on informal notations.

The most popular "formal" notations and languages are goal models, DFDs, and process models. However, it could be argued that DFDs are also a kind of process model, what would make process models the most used notation in the identified literature. Goal models describe why the system-to-be shall be built and which properties it shall have. These models also allow to model how these goals can be achieved by specific tasks. DFDs and process models provide a behavioral view on the system-to-be, still abstracting from the concrete and technical realization of processes.

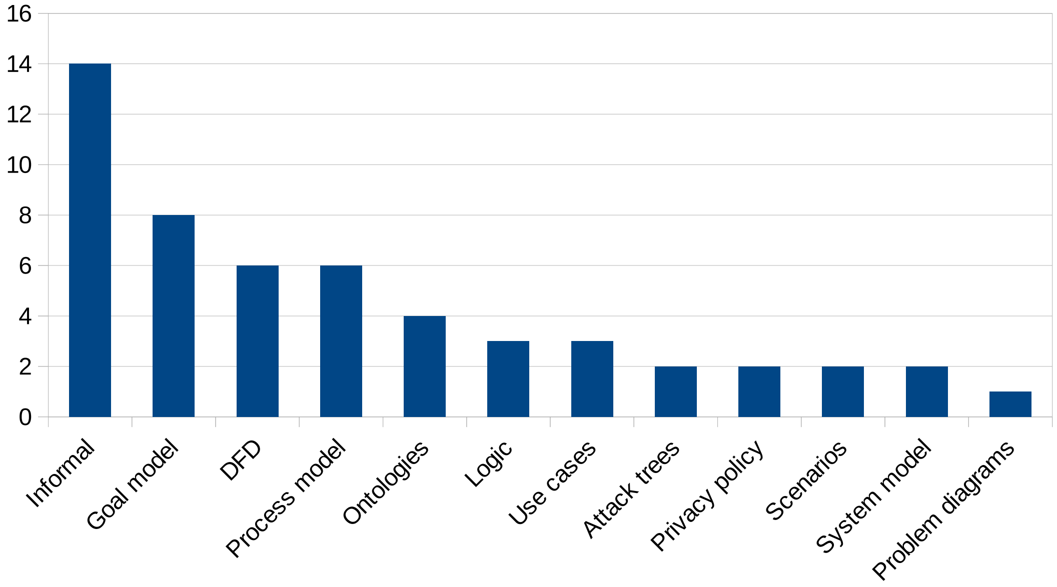


Figure 3.5.: *System modeling languages identified during the literature review*

We can conclude from the literature review, that a privacy requirements engineering method needs to capture and consider both, the representation of the requirements on the system-to-be, and a description of the entities in the environment of the machine and their interplay. The problem frames approach proposed by Jackson (2000) (cf. Section 2.2) allows to capture and relate the requirements on the system-to-be with the entities in the environment of the machine. While goal modeling supports the task of identifying functional requirements, the problem frame approach is more focused on deriving the specification of the machine based on the functional requirements by focusing on single functional requirements and taking explicitly into account the involved entities. However, goal and problem oriented approaches are complementary and can be combined (Liu and Jin, 2006; Beckers et al., 2013b; Mohammadi et al., 2013).

I selected the problem frames requirements engineering method as foundation for the privacy requirements engineering method proposed in this thesis, because I focus on a privacy analysis of a given set of functional requirements. Additionally, I decided to take the problem frames approach because problem diagrams provide means to connect requirements with the environment of the machine and provide a more concrete picture of how the machine is connected to its environment. This additional connection is unique to the problem frames approach and does not exist as such in goal, use-case, or DFD models. A behavioral view similar to a DFD is systematically derived from the given problem frame model (cf. Section 2.3) during the ProPAn method (cf. Chapter 11). That means, that I provide support for the creation of a behavioral model that illustrates the flow of personal data in a system based on its functional requirements that are represented in problem diagrams.

3.4.4. Operationalization Strategy (Strat.)

From the literature review, it can be recognized that both, refinement-based and prevention-based privacy engineering methods can be followed to elicit and operationalize privacy requirements and that these can also be combined (see also column **Strat.** in Tables 3.20 and 3.21). I think that it is valuable to combine both approaches for the operationalization of privacy requirements to functional requirements, because privacy as a software quality has to be refined to functionalities, and potential risks to the system-to-be or imposed by it have to be elicited and treated. Hence, the ProPAn method combines both approaches.

Only one quarter of the methods does not support the operationalization of privacy requirements (see also column **Op.** in Tables 3.20 and 3.21). Most methods support the operationalization by providing lists of PETs that could be used to address specific privacy goals or threats. With the ProPAn method, I want to provide more guidance and support for the operationalization of privacy by providing a method to systematically integrate PETs as *cross-cutting* functional requirements into the system-to-be (see Part IV).

3.4.5. Detail of Methodology and Methodological Support (Meth. and Techniques)

The provided methodological support is quite diverse in the considered privacy requirements engineering methods (see also column **Meth.** in Tables 3.20 and 3.21). I differentiate between fine-grained, coarse-grained, and implicitly mentioned methods indicated by a "+", "-", and "O", respectively, in Tables 3.20 and 3.21. A fine-grained method provides a detailed description of its steps and the involved artifacts. A coarse-grained method provides high-level steps that need to be performed and possibly sketches what kind of artifacts may be involved. An implicit method does not explicitly describe a methodology, but proposes, e.g., guidelines or artifacts to be used for privacy requirements engineering.

The kind of methodological support also varies among the privacy engineering methodologies (see also column **Techniques** in Tables 3.20 and 3.21). Most methods that consider the operationalization of privacy requirements provide exemplary catalogs of PETs that can be used during the privacy requirements operationalization process. Other support provided by methods includes questionnaires, templates, patterns, and ontologies. Only eight methods suggest no specific techniques that can be used to perform their method steps.

The ProPAn method is a systematic and fine-grained method that provides for all its steps supporting notations, models, templates, and questionnaires that support the performance and documentation of these steps. For each step of the ProPAn method, the inputs, the process of the step, the outputs, and validation conditions (that can be used to check whether the created artifacts contain errors) are defined.

3.4.6. Extend requirements specification and system model (Ext.)

Only few methods support the high-level task *Extend requirements specification and system model* (see also column **Ext.** in Tables 3.20 and 3.21). This means, only few methods explicitly consider the elicitation of additional (domain) knowledge that is necessary to perform a privacy analysis. This knowledge may include information about implicit flows outside the machine's control or availability of personal data at domains independent from the machine. Furthermore, only few methods consider how to elicit the information flows imposed by the system-to-be based on its functional requirements and the privacy requirements implied by these. The ProPAn method aims at closing these gaps by providing systematic methods to elicit and document the domain knowledge and information flows.

3.4.7. Tool Support

Half of the privacy requirements engineering methods do not provide tool support. One quarter provides partial support, e.g., tool support for modeling goal models (see also column **Tool** in Tables 3.20 and 3.21). Only 6 methods provide more comprehensible tool support that allows, e.g., automatic reasoning about the created models. It can be argued that tool support is, especially for model driven methods, essential to support the application of the method and to utilize the documented knowledge about the system-to-be and its requirements for the privacy analysis. The results of the latter can then be used to generate documentation needed to show compliance to specific regulations or standards. For example, a report for a privacy impact assessment (PIA) could partially be generated from artifacts created during a privacy requirements and risk analysis.

In this thesis, I present a computer-aided method for privacy requirements engineering. All steps of the ProPAn method are supported by the provided tool support and the elicited knowledge is documented in a structured and computer-readable way using EMF models (cf. Section 2.3). Methodological support is provided to systematically create the needed models, steps of the method are automated, and validation of the created models is performed automatically to check the consistency of these whenever this is possible.

3.5. Conclusions

In this chapter, I provided an overview of the state of the art in privacy requirements engineering. For this, I conducted a literature review combining automatic search based on the search engine Scopus and backward snowballing. To compare the 40 identified privacy requirements engineering methods with each other, I proposed comparison criteria based on existing work of Martín and del Álamo (2017) and a high-level privacy requirements engineering methodology that I derived based on the state of the art methods. I presented all 40 privacy requirements engineering methods based on the identified criteria. Finally, I discussed the state of the art methods and identified weaknesses and gaps in these methods that shall be addressed by the ProPAn method that is proposed in this thesis. I also sketched how I propose to address the weaknesses and gaps. In Chapter 20, I evaluate the ProPAn method based on the criteria presented in this chapter and discuss to which extend the ProPAn method addresses the identified weaknesses and gaps.

Chapter 4

Electronic Health System (Running Example)

In this chapter, I introduce a subsystem of an electronic health system (EHS) that I use as running example in Chapters 9–17 to illustrate the application of the ProPAn method. The EHS scenario is based on documents (Cubo et al., 2011) provided by the industrial partners of the EU project *Network of Excellence (NoE) on Engineering Secure Future Internet Software Services and Systems* (*NESSoS*)[1].

In Section 4.1, I describe the considered part of the EHS scenario and the selected functional requirements. The context diagram for the EHS scenario is explained in Section 4.2, and the problem diagrams in Section 4.3.

4.1. Scenario Description

An electronic health system shall be developed to allow the management of electronic health records (EHR). Doctors shall be able to browse, access, create, and modify EHRs. The selected EHS scenario is based on the German health care system which uses health insurance schemes for the accounting of treatments.

The EHS shall support doctors to bill their patients based on the information contained in the EHRs and an insurance application which forms the connection to the patients' insurances to perform the accounting of treatments. The insurance application also provides the information whether specific treatments are covered by the patients' insurance contracts or not. In the case that treatments are not covered by a patient's insurance contract, a respective invoice shall be created. In a billing process the EHS shall make use of a financial application to bill a patient if an invoice was created for him or her.

The EHS shall support mobile devices to measure and record patients' vital signs to add these to their EHRs. It shall also be possible to send instructions, appointments, and alarms to patients using these mobile devices.

Finally, the EHS shall support researchers to access anonymized medical data for clinical research via a research database application. The medical data shall be based on the EHRs.

From the EHS description, I derived the following 7 functional requirements.

R1 Doctors shall be able to modify and create EHRs.

R2 Doctors shall be able to browse and access EHRs.

R3 Doctors shall be able to initiate the accounting of their performed treatments. This accounting shall be performed using an insurance application that returns which treatments are

[1] `http://www.nessos-project.eu/`

covered by the patient's insurance contract and which not. In the case that a treatment is not beared by a patient's insurance, an invoice shall be created.

R4 Regularly, unpaid invoices shall be billed using a financial application.

R5 Alarms, appointments, and instructions shall be shown to patients using their mobile devices based on the patients' health records.

R6 Vital signs of patients measured by their mobile device shall be added to their EHR.

R7 Researchers shall be able to access anonymized medical data for clinical research based on the EHRs via a research database application.

4.2. Context Diagram

The context diagram for the EHS (see Figure 4.1) shows the machine EHS in its environment. The lexical domains Invoice and EHR are considered as designed domains and hence as part of the EHS. All other domains are given domains.

The Financial Application and Insurance Application are both causal domains that the EHS can use to perform the billing and accounting, respectively. Both applications return then a corresponding result.

The biddable domain Doctor is able to modify, create, and browse (including access) EHRs. Furthermore, a Doctor and can initiate the accounting of patients. In return to the browse request, the EHS returns the requestedEHR, which the Doctor can observe.

A Patient is also a biddable domain and connected to the EHS via the causal domain Mobile Device. A Mobile Device can observe a Patient's vital signs and send these to the EHS. Furthermore, the EHS can send appointments, instructions, and alarms to a Mobile Device, which in consequence shows these to the respective Patient.

The Research Database Application is a causal domain that allows a Researcher to request medical data from the EHS. If this request can be granted, the EHS shall provide the requested data to the Researcher via the Research Database Application.

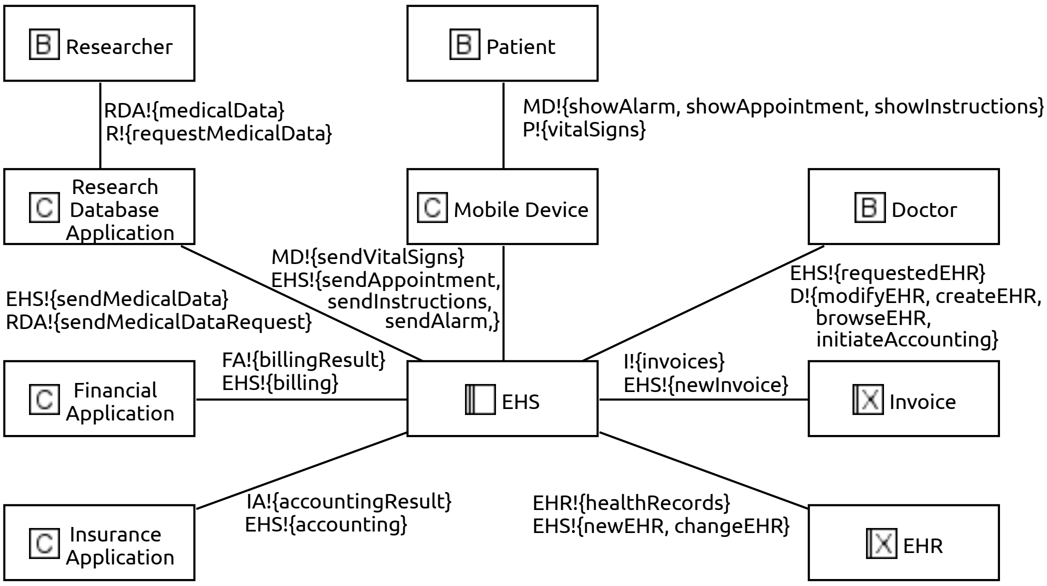


Figure 4.1.: *Context diagram for the EHS scenario*

4.3. Problem Diagrams

4.3.1. R1 – Manage EHRs

Figure 4.2 shows the problem diagram for requirement R1 ("*Doctors shall be able to modify and create EHRs.*"). R1 refers to the commands modifyEHR and createEHR that a Doctor can issue and that are observed by the EHS. R1 constrains the healthRecords contained in the EHR to be modified in accordance with the referred to commands. The EHS can operate on the health records using the commands newEHR and changeEHR to create new health records and to change existing health records, respectively.

4.3.2. R2 – Browse EHRs

Requirement R2 ("*Doctors shall be able to browse and access EHRs.*") is visualized in the problem diagram shown in Figure 4.3. R2 refers to the command browseEHR that a Doctor can issue and that is observed by the EHS. Additionally, R1 refers to the healthRecords contained in the EHR that the Doctor wants to access and browse. The healthRecords can be observed by the EHS and the EHS can provide requestedEHR to the requesting Doctor. R2 constrains that a Doctor gets knowledge about the provided EHRs. That means, R2 requires not only that the requested health records are sent, but also that the doctor receives and perceives these. In order to satisfy this requirement, we have to assume that doctors are able and willing to understand the observed phenomenon requestedEHR. This assumption can be phrased as follows: If a doctor receives the requestedEHR, he or she receives and perceives these, such that he or she knows the requested (knowsEHR).

4.3.3. R3 – Accounting

The problem diagram in Figure 4.4 presents requirement R3 (*"Doctors shall be able to initiate the accounting of their performed treatments. This accounting shall be performed using an insurance application that returns which treatments are covered by the patient's insurance contract and which not. In the case that a treatment is not beared by a patient's insurance, an invoice shall*

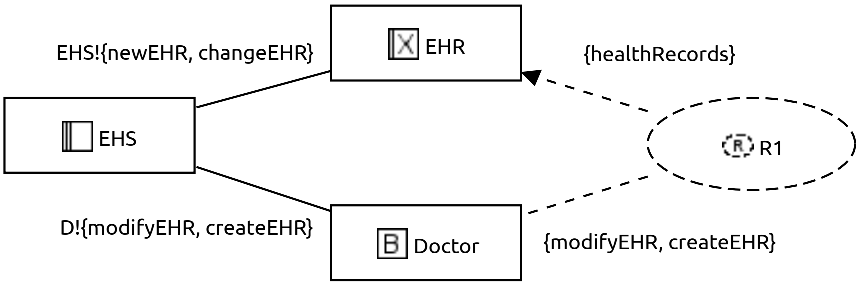


Figure 4.2.: *Problem diagram for requirement R1 in the EHS scenario*

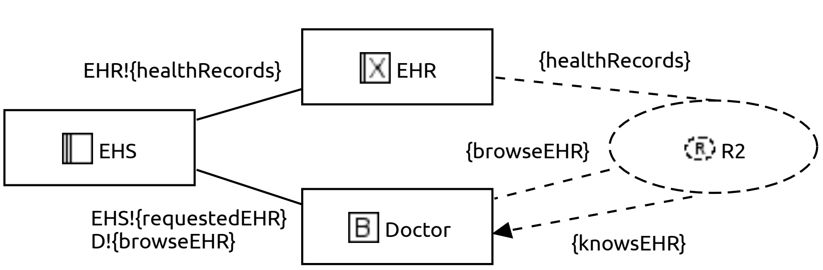


Figure 4.3.: *Problem diagram for requirement R2 in the EHS scenario*

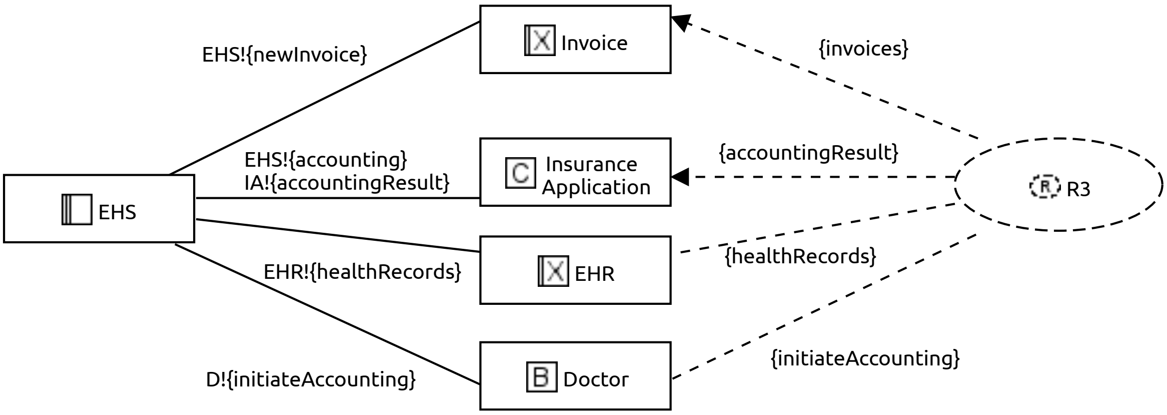


Figure 4.4.: *Problem diagram for requirement R3 in the EHS scenario*

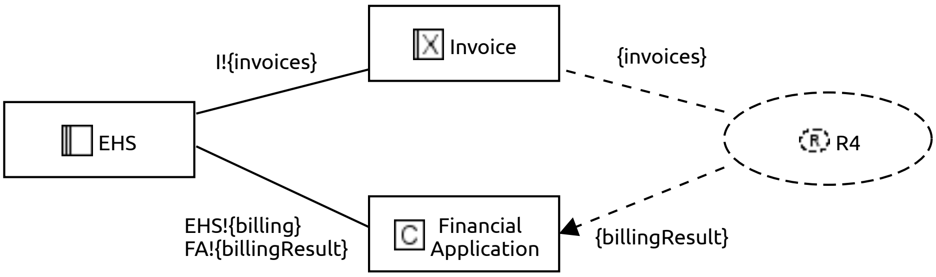


Figure 4.5.: *Problem diagram for requirement R4 in the EHS scenario*

be created."). Note that R3 is not a *simple* problem according to Jackson, because it does not fit to a problem frame. It could be decomposed into the problems of initiating the accounting, performing the accounting, and creating an invoice. However, I decided not to decompose this requirement, to assess the impact of the functional requirements' granularity on the privacy analysis.

R3 refers to the command initiateAccounting controlled by the domain Doctor and the healthRecords (containing the performed treatments) controlled by the EHR. The EHS is able to observe both referred to phenomena. R3 constrains the accountingResult returned by the Insurance Application as reaction to the command accounting that the EHS can issue. Finally, R3 constrains the creation of respective invoices based on the accountingResult. To create invoices, the EHS can issue the command newInvoice that is observed by the lexical domain Invoice.

4.3.4. R4 – Billing

Figure 4.5 shows the problem diagram for R4 (*"Regularly, unpaid invoices shall be billed using a financial application.*). R4 refers to the invoices that are open. These invoices are also observable by the EHS. R4 constrains the billingResult provided by the Financial Application. That is, R4 requires the Financial Application to successfully handle the billing. This billing is initiated by the EHS using the command billing. For the sake of simplicity, we assume that the Financial Application is able to successfully handle the billing requests and that we do not have to handle the case of an unsuccessful billing.

4.3.5. R5 – Send Instructions

Requirement R5 (*"Alarms, appointments, and instructions shall be shown to patients using their mobile devices based on the patients' health records.*) is visualized in the problem diagram shown

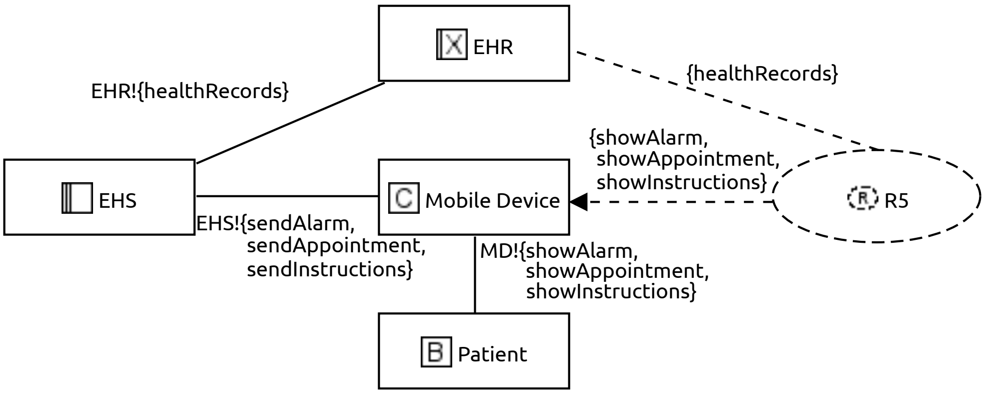

EHR
EHR!{healthRecords}
{healthRecords}
EHS
Mobile Device
{showAlarm, showAppointment, showInstructions}
R5
EHS!{sendAlarm, sendAppointment, sendInstructions}
MD!{showAlarm, showAppointment, showInstructions}
Patient

Figure 4.6.: *Problem diagram for requirement R5 in the EHS scenario*

in Figure 4.6. R5 refers to the healthRecords which shall be used to create alarms, appointments, and instructions. R5 constrains that these alarms, appointments, and instructions are shown by the respective Mobile Device. The Patient can observe the shown alarms, appointments, and instructions shown by his or her Mobile Device. The EHS is able to observe the healthRecords and to send alarms, appointments, and instructions to the Mobile Device.

When we compare the problem diagram in Figure 4.6 with the problem diagram for requirement R2 in Figure 4.3, then we see two different approaches to model feedback to biddable domains. R2 constrains the biddable domain to really perceive the received information, while R5 only constrains the mobile device to present the information to patients. I decided to model these requirements following the different approaches, which leads to an inhomogeneous modeling of the requirements, to assess the impact of the way how functional requirements are modeled on the privacy analysis. The impact of how problem diagrams are modeled on the ProPAn method becomes visible in Sections 9.2 and 9.4 where I describe that the domain knowledge has to be documented that patients get information from their mobile devices, and that there is a connection domain refining the interface between the domains EHS and Doctor.

4.3.6. R6 – Record Vital Signs

The problem diagram in Figure 4.7 presents requirement R6 (*"Vital signs of patients measured by their mobile device shall be added to their EHR."*). R6 refers to the Patient's vitalSigns that are observed by the Mobile Device. Furthermore, R6 refers to the event sendVitalSigns issued by the Mobile Device, which is observed by the EHS. R6 constrains the healthRecords to contain the measured vital signs. The EHS is able to issue the command changeEHR to add the received vital signs to the respective health record.

4.3.7. R7 – Clinical Research

Figure 4.8 shows the problem diagram for R7 (*"Researchers shall be able to access anonymized medical data for clinical research based on the EHRs via a research database application.*). R7 refers to the event requestMedicalData that a Researcher issues to request medical data from the Research Database Application. Furthermore, R7 refers to the healthRecords which serve as the basis to generate the requested medical data. Finally, R7 constrains the Research Database Application to provide the medicalData to the requesting Researcher. The Research Database Application serves as connection domain between the Researcher and the EHS, i.e., it forwards requests of the Researcher to the EHS, and also the generated medical data from the EHS to the Researcher.

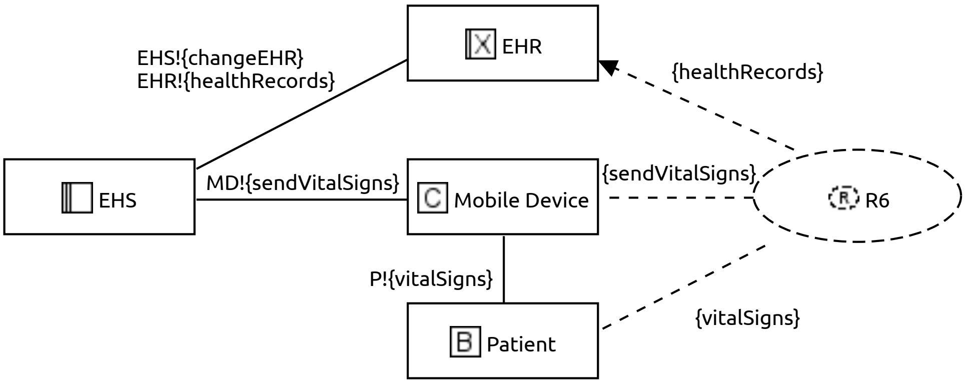


Figure 4.7.: *Problem diagram for requirement R6 in the EHS scenario*

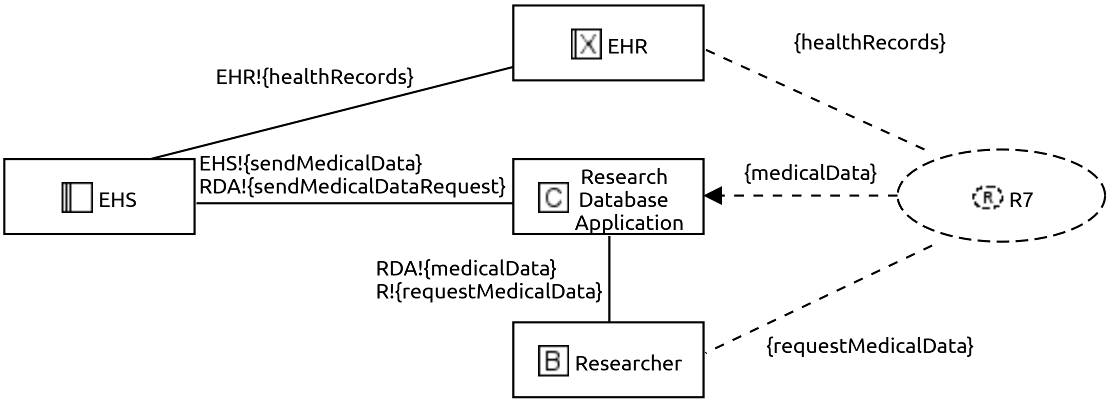


Figure 4.8.: *Problem diagram for requirement R7 in the EHS scenario*

Part II.

Privacy Requirements

Chapter 5

Refining the Privacy Goal Transparency

In this chapter, I present a taxonomy that refines the privacy goal transparency into more fine-grained, still non-functional, privacy requirements. The work presented in this chapter is already published in (Meis et al., 2015). Co-authors of this paper are Maritta Heisel, who provided helpful comments to improve the research, and Roman Wirtz, who proposed an initial version of the presented taxonomy in his bachelor thesis (Wirtz, 2014). The presented refinement process, final taxonomy, and the literature review are my own contributions.

This chapter is structured as follows. My transparency requirements taxonomy is derived and presented in Section 5.2. Then it is validated using related work identified using a systematic literature review in Section 5.3. Section 5.4 concludes this chapter.

5.1. Introduction

The awareness for privacy concerns is growing in the public. With this awareness comes a call for more transparency on what, why and how software-systems collect, use, and process personal information. Hansen (2012) identifies transparency as one of three privacy protection goals ensuring *"that all privacy-relevant data processing including the legal, technical and organizational setting can be understood and reconstructed"* (Probst and Hansen, 2013). Hence, it is not sufficient to increase user's privacy awareness, it is also necessary to provide to users the information that they need to understand how their personal data are processed. Transparency, as all software qualities, is a complex property. It leads to requirements for the representation of static information about the software's intended purpose, but also to requirements on informing users about run-time events, e.g., malfunctions. In addition to the requirements about informing *what* happens, there are also requirements on *how* the information is shown to users to ensure that mechanisms to improve the software's transparency have an impact on the user's privacy-awareness. Especially concerning legal compliance, requirements engineers have to provide an as complete set of requirements as possible to ensure that the software that is built based on these requirements is compliant. That means, the software requirements have to bridge the gap between the legal requirements and the technical mechanisms to realize them. To empower requirements engineers to identify all transparency requirements relevant for the software-to-be, we have to refine the high-level privacy goal transparency into more concrete transparency requirements that assist requirements engineers in the elicitation process.

To obtain an as complete taxonomy of transparency requirements as possible, we consider different sources that requirements engineers also should consider. To be compliant with legislation, requirements engineers have to consider privacy and data protection laws relevant to them. Depending on the application domain of the software-to-be, also standards have to be considered. To increase user acceptance, the user's needs have to be considered. I use as sources for the creation of my taxonomy the ISO 29100 standard (ISO/IEC, 2011) and the draft of the EU Data Protection Regulation (European Commission, 2012)[1]. I then consider relevant research in the field of privacy, transparency, and awareness including empirical research on user's privacy concerns to validate the completeness of the proposed taxonomy.

5.2. Deriving and Structuring Requirements on Transparency

In Section 5.2.1, I systematically analyze the privacy principles described in ISO 29100 (ISO/IEC, 2011) and the draft of the EU data protection regulation (European Commission, 2012) complemented by the guidelines given by the Article 29 Data Protection Working Party (2018) to derive the transparency requirements they contain. To derive the requirements, I analyzed the description of the privacy principles and the formulations of the regulation. I looked for verbs like *inform*, *notify*, *document*, *present*, *provide*, *explain*, *communicate* and related nouns. I keep the formulation of the identified transparency requirements close to the original documents from which I identified them. In Section 5.2.1, I enumerate these derived requirements using the notation Tn. As the ISO principles and EU articles partly overlap, I identified several refinements of identified requirements. I relate those requirements using a *refines* relation. If a transparency requirement Tn_1 refines a part of another requirement Tn_2, this means that Tn_1 adds further details on how or what information has to be made transparent. The *refines* relation is visualized in form of an initial ontology of transparency requirements in Figure 5.1. In Section 5.2.2, I structure the transparency requirements identified in Section 5.2.1 into a taxonomy of transparency requirements. This taxonomy is presented as an extensible metamodel.

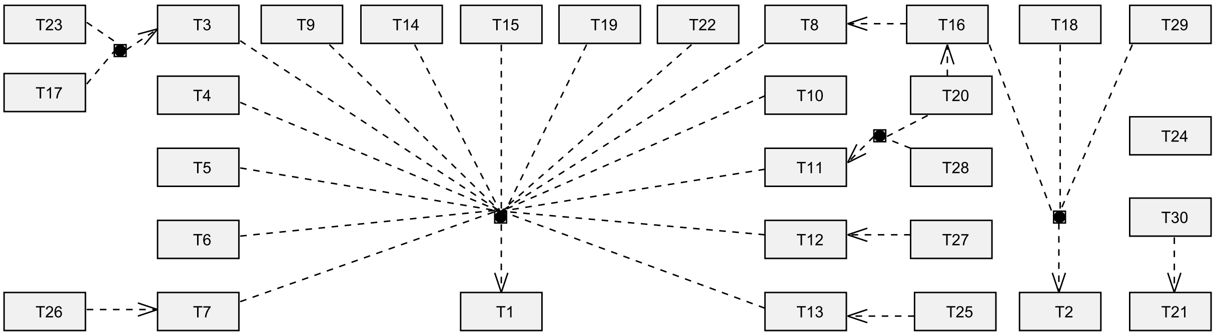


Figure 5.1.: *Initial ontology of transparency requirements*

ISO 29100 and the draft of the EU data protection regulation do not use the same terminology. To avoid ambiguities, I use the terms introduced in Section 2.4.2 on page 25.

5.2.1. Requirements Identification from Privacy Principles and Legislation

In the following two subsections, I derive from the international standard ISO 29100 (ISO/IEC, 2011) (see Section 5.2.1.1) and the draft of the EU Data Protection Regulation (European Commission, 2012) complemented by the guidelines given by the Article 29 Data Protection Working Party (2018) (see Section 5.2.1.2) initial privacy requirements.

[1] The draft of the EU data protection regulation was adopted with some changes on 27 April 2016 and entered into force on 24 May 2016. Note that my analysis is based on the draft and not on the final version of the regulation. However, my results also apply for the final general data protection regulation (GDPR).

5.2.1.1. ISO 29100 Privacy Principles

ISO 29100 (ISO/IEC, 2011) defines 11 privacy principles which are a superset of the OECD principles (OECD, 1980) and the US fair information practices (FIPs) (US Federal Trade Commission, 1998) (cf. Section 2.4.3).

I start my analysis of the privacy principles with the *openness, transparency and notice principle*, which is obviously concerned with transparency. From this principle, I obtain the following transparency requirements.

T1 Inform data subjects about the controller's policies, procedures and practices with respect to the processing of personal data.

T2 The information about the management of personal data has to be clear and easily accessible for data subjects (and the public).

T3 Explain the purpose of data processing to data subjects, i.e., why do the personal data need to be processed.

T4 Specify the persons[2] to whom the personal data might be disclosed.

T5 Provide the identity of the controller including contact information to data subjects.

T6 Provide information about the choices to limit the processing of personal data to data subjects.

T7 Provide information about the means to access, correct and remove personal data to data subjects.

T8 Provide information in the case that a decision that a data subject can make has an impact on the data subject.

T9 Document and communicate all contractual obligations that impact personal data processing externally to the extent those obligations are not confidential.

T10 Provide information about the personal data required for the specified purpose to data subjects.

T11 Provide information about how and what personal data are collected to data subjects.

T12 Provide information about how, what and to whom personal data are communicated to data subjects.

T13 Provide information about how and what personal data are stored to data subjects.

T14 Provide information about authorized natural persons who will access personal data to data subjects.

T15 Provide information about data retention and disposal requirements.

T1 and T2 are the most general requirements in my initial ontology. Hence, they form the root elements (cf. Figure 5.1). T1 is considered with *what* information has to be presented and is refined by T3-T15 that are all also concerned with about what data subjects have to be informed. In contrast, T2 is concerned with *how* that information has to be presented to data subjects.

[2] Using the term *person* I refer to natural, legal, as well as artificial persons, e.g., organizations or authorities.

The *consent and choice principle* strengthens that data subjects have to give their consent on a *"knowledgeable basis"* and hence, they have to be informed before obtaining consent. This information has also to contain information about *"the implications of granting or withholding consent"*. I identify the following requirement.

T16 Before data subjects are asked to give consent to use their data, provide all information necessary to make this decision to them, including the implications of granting or withholding consent.

This requirement refines T2 in the sense that the point in time when the information has to be provided is specified. Additionally, T16 refines T8 by describing which data has to be provided to data subjects when they make the decision to give consent.

The principle *purpose legitimacy and specification* stresses that data subjects have to be informed about the purpose of data collection and use before the data are used for the first time or for a new purpose. This information has to be presented using language *"which is both clear and appropriately adapted to the circumstances.* In the case that sensitive data are processed, sufficient explanations have to be provided to the data subject. Hence, I obtain following requirements.

T17 Inform data subjects about the purpose of data collection and use before the data are collected or used for the first time for this purpose.

T18 The language used for providing information to data subjects has to be clear and appropriately adapted to the circumstances.

T19 Provide sufficient explanations whenever sensitive data are used to data subjects.

Requirement T17 complements T3 with the information when data subjects have to be informed. T18 is a refinement of T2 by adding the notice that the presentation has to be adapted to the circumstances in which this information is shown. T19 places emphasis on providing explanations whenever sensitive data are used and hence refines the top-level requirement T1.

The principle *collection limitation* is concerned with limiting the collected personal data to the minimum needed. I obtain the following additional requirement.

T20 Provide information to data subjects about if it is optional to provide personal data.

This requirement complements T11 and T16, because it is important to inform data subjects before data collection and giving consent whether it is optional to provide the questioned personal data.

The principle *accountability* contains the following transparency requirements that are concerned with the occurrence of privacy breaches, which is not yet covered by other transparency requirements, because the other requirements are concerned with the normal behavior of the system under consideration.

T21 Inform data subjects and other relevant stakeholders (as required in some jurisdictions) about privacy breaches that can lead to substantial damage to data subjects as well as the measures taken for resolution.

The principle *information security* implies the following transparency requirement that refines the transparency requirement T1.

T22 Inform data subjects about the (security) mechanisms to protect their personal data.

5.2.1.2. Draft of the EU Data Protection Regulation

To identify further transparency requirements and to refine the already identified requirements, I analyze the draft of the EU Data Protection Regulation (European Commission, 2012).

Article 5 (b) adds the need that the purpose has to be legitimate to requirement T3. Hence, I obtain the following refined requirement.

T23 Explain data subjects why the purpose of data collection is legitimate.

Article 12 prescribes the implementation of procedures and mechanisms for exercising the rights of data subjects and says that *"If the controller refuses to take action on the request of the data subject, the controller shall inform the data subject of the reasons for the refusal and on the possibilities of lodging a complaint to the supervisory authority and seeking a judicial remedy."*. Hence, I identify a transparency requirement that, similar to T21, is not concerned with the normal system behavior.

T24 If requests of data subjects for exercising their rights are rejected, then the reasons for the refusal have to be provided and the possibilities of lodging complaints.

From Article 14, I can derive following transparency requirements that refine previously identified requirements.

T25 Provide the period for which the personal data will be stored to data subjects.

T26 Provide information about *"the existence of the right to request from the controller access to and rectification or erasure of the personal data concerning the data subject or to object to the processing of such personal data"*

T27 Provide information about data transfer *"to a third country or international organisation and the level of protection afforded by that third country or international organization"*.

T28 Inform the data subject about the source the personal data used originate from.

T29 Provide information to data subjects *"at the time when the personal data are obtained from the data subject; or where the personal data are not collected from the data subject, at the time of the recording or within a reasonable period after the collection, having regard to the specific circumstances in which the data are collected or otherwise processed, or, if a disclosure to another recipient is envisaged, and at the latest when the data are first disclosed."*

T25 refines T13 by adding the need for specifying the duration of data storage. T26 adds a legal need to T7. T27 refines T12 by requiring special treatment when data are transferred to third countries or international organizations. T28 refines T11 by adding the need to provide information of the source of the personal data used. T29 refines T2 with information about when to provide information to data subjects.

Article 31 is concerned with the notification of personal data breaches and refines T21 by adding a duration after which the supervisory authorities have to be informed.

T30 Notify supervisory authorities (and data subjects) about the occurrence of a personal data breach not later than 24 hours after having become aware of it.

5.2.2. Setting up a Transparency Requirements Taxonomy

In this section, I structure the identified preliminary transparency requirements into a transparency requirements taxonomy. Figure 5.2 shows my taxonomy in the form of a metamodel as a UML class diagram. I structured the transparency requirements into a hierarchy, which is derived from the initial ontology shown in Figure 5.1. I describe my taxonomy in the following from the top to the bottom. An overview of the mapping between the transparency requirements taxonomy to the initial transparency requirements is given in Table 5.1.

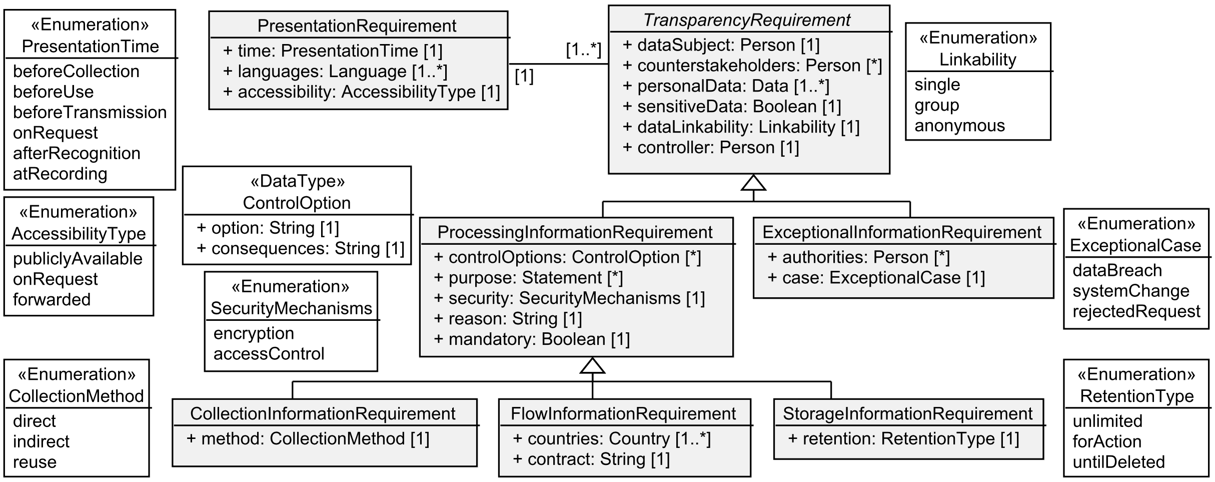


Figure 5.2.: *My proposed taxonomy of transparency requirements.*

Table 5.1.: *Mapping of transparency requirements to preliminary requirements*

Requirement	Attribute	Tn
TransparencyRequirement	data subject, personal data	T1
	controller	T5
	counterstakeholder	T4, T14
	linkability	T16
	sensitiveData	T19
PresentationRequirement	accessibility	T2
	language	T18
	time	T16, T29, T30
ExceptionalInformationRequirement	case	T17, T21, T24, T30
	authorities	T21
ProcessingInformationRequirement	controlOptions	T6, T7, T8, T26
	mandatory	T10, T20
	purpose, reason	T3, T17, T23
	security	T22
CollectionInformationRequirement	method	T11, T28
StorageInformationRequirement	retention	T13, T15, T25
FlowInformationRequirement	contract, country	T9, T12, T27

5.2.2.1. Transparency Requirement

The top-level element of my hierarchy is the general *TransparencyRequirement* which corresponds to the initial requirement T1. In my metamodel I declared this requirement as *abstract*, i.e., it is not possible to instantiate it, only its specializations can be instantiated. It has six attributes. First, the dataSubject who has to be informed. Second, a set of counterstakeholders who are involved in the processing of the data subject's data and the data subject has to be informed about them. For example, T4 and T14 prescribe to specify the (authorized) persons to whom personal data might be disclosed. This is the case for many requirements in my taxonomy and hence, I put this attribute to the top-level requirement. If there is no need to specify persons who are somehow involved in the data processing, the attribute counterstakeholder is left empty. My taxonomy suggests to consider data subjects and counterstakeholders as persons. The data subject should be a natural person, whereas the counterstakeholders can be natural, legal, or artificial persons, e.g., organizations or authorities. Third, the set of personal data of the data subject for which the transparency requirement is relevant. Almost all transparency requirements that I identified previously refer to the data subject and his or her personal data. Hence, all transparency requirements in my taxonomy have the data subject and his or her personal data as attribute. Fourth, I document whether the specified personal data represent senstiveData, because T19 states that sensitive data need special consideration. Fifth, the attribute linkability documents whether the personal data are linkable to a single data subject, a group of possible data subjects, or is anonymous. This attribute is not explicitly motivated from the requirements, but T16 mentions that in the case of giving consent all information necessary to make this decision has to be provided to data subject and I think that the linkability of the personal data to the data subject is such an information. Sixth, in accordance with T5 the data subject has to be informed about who the controller is.

5.2.2.2. Presentation Requirement

The initial transparency requirements T2, T16, T18, T29, and T30 are in contrast to the other requirements not mainly concerned with *what* information shall be provided to the data subject, but with *how* this information has to be presented. To decouple the *how* from the *what* in my taxonomy, I introduce PresentationRequirements. Every TransparencyRequirement has exactly one PresentationRequirement assigned, which describes how the information has to be provided to the data subject. On the other side, the same PresentationRequirement can be related to multiple TransparencyRequirements. The attribute time reflects T16, T29, and T30 that prescribe the time when information has to be provided. The possible values for this attribute are summarized in the enumeration PresentationTime (cf. Figure 5.2). I derived these values from T16, T29 and T30. Nevertheless, I do not consider this enumeration, such as all other enumerations presented in my taxonomy, as complete; whenever necessary they can be extended. The attribute languages is not explicitly mentioned in a transparency requirement, but to provide information clearly and adapted to the circumstances to data subjects (in accordance with T18) one should present this information using at best the first language of each possible data subject. The attribute accessibility serves to document the requirements on how a data subject shall be able to access the information, indicated by T2. An information may have to be publiclyAvailable, onRequest of the data subject, or the information is forwarded to the user when needed.

5.2.2.3. ExceptionalInformationRequirement

Most transparency requirements are concerned with providing information about the normal behavior of the considered system. This information can be considered as rather static. In contrast, T21, T24, and T30 require to inform data subjects in cases where unexpected events occur. For this purpose, I refine the general TransparencyRequirement into the requirement ExceptionalInformationRequirement. The attribute case stores the kind of unintended event the data subject has to be informed about. This can be a dataBreach as mentioned in T21 and T30, a systemChange that, e.g., changes the purpose of data processing (cf. T17), or a rejectedRequest of a data subject as described in T24. In addition to the data subject that has to be informed, T21 also states that authorities may have to be informed. The attribute authorities is used to document the natural, legal, or artificial persons that have to be informed if the respective exceptional case occurs.

5.2.2.4. ProcessingInformationRequirement

The requirement ProcessingInformationRequirement refines TransparencyRequirement and contains the properties that all transparency requirements, which refine the initial requirement T1 (cf. Figure 5.1), have in common. The attribute controlOptions summarizes (using the data type ControlOption) the options the data subject has to limit the processing of personal data (T6), means to access, correct and remove personal data (T7 and T26), and the consequences implied by these options (T8). T3, T17, and T23 require that the purpose of data processing is explained to data subjects. The attribute purpose is used to provide a set of Statements that could consist of functional requirements and knowledge about the software environment (cf. Section 2.1) for whose fulfillment the personal data of the data subject are needed. Furthermore, the attribute reason is used to provide information about why the personal data are needed for the purpose and why it is legitimate to use them. Due to T10 and T20, data subjects have to be clearly informed whether the provision of personal data is optional and whether the personal data are needed for the specified purpose. The attribute mandatory is used to capture this information. The attribute security is used to represent how the personal data are protected as required by T22. Possible protection mechanisms are e.g., encryption and accessControl.

5.2.2.5. CollectionInformationRequirement

Requirement T11 prescribes that data subjects have to be informed about how and what data are collected from them. For this purpose, I refined the ProcessingInformationRequirement into the CollectionInformationRequirement. In addition to the information that is already inherited from TransparencyRequirement and ProcessingInformationRequirement, I derived from T28, which is a refinement of T11 (cf. Figure 5.1), the attribute method that reflects whether the data collection is direct, indirect, or whether existing data of the subjects are reused.

5.2.2.6. FlowInformationRequirement

Requirement T12 implies a further refinement of ProcessingInformationRequirement that I call FlowInformationRequirement. This requirement prescribes to inform data subjects about the flow of their data. From T9 and T27, I derived that for each information flow, it is important to inform the data subject about the contractual obligations and policies the data receiver is bound to. This information is represented in the attribute contract. Furthermore, T27 puts an emphasis on taking care of data transfer to *third countries* and international organizations. Hence, I added the attribute countries to capture the geographical destination of the data flow.

5.2.2.7. StorageInformationRequirement

From T13, I derive the requirement StorageInformationRequirement that is also a refinement of ProcessingInformationRequirement. This requirement is used to represent the information that is needed to inform the data subject about the storage of his or her personal data. In addition to the attributes inherited from TransparencyRequirement and ProcessingInformationRequirement, T15 and T25 require that the data subject is informed about the duration of storage and the data retention and disposal requirements. To reflect this information, I use the attribute retention. The possible values of this attribute can indicate that personal data are stored for an unlimited time, as long as they are needed for the purpose they were collected for (forAction), or until they are deleted (untilDeleted) after there is no reason to keep the data anymore, but not necessarily directly after the data are no longer needed.

The complete taxonomy is shown in Figure 5.2. Note that the taxonomy is easily extensible by further refinements of requirements, adding further attributes and relations, and adapting the suggested enumerations to the needs implied by the application domain and relevant legislation of the software to be developed. Table 5.1 provides an overview of how the initial requirements Tn that I derived from ISO 29100 and the draft of the EU Data Protection Regulation are reflected by the proposed taxonomy.

5.3. Validation of the Taxonomy Using Related Literature

In this section, I give an overview of existing research that also contains considerations about the privacy goal of transparency. To validate my proposed taxonomy, I map the notions and concepts used in the related literature to my taxonomy to check whether it is suitable to reflect the shapes of transparency used in the literature.

To identify the relevant related work, I performed a systematic literature review using backward snowballing (Jalali and Wohlin, 2012). To obtain the starting set of papers for my review, I manually searched the proceedings and issues of the last 10 years of computer science conferences and journals that are mainly concerned with at least one of the topics privacy, requirements, and software engineering and ranked at least as *B-level* in the CORE2014 ranking[3]. In this way, I selected 15 conferences and 19 journals. First, I checked whether title or abstract of a paper indicated that the paper is concerned with privacy (requirements), transparency, or awareness. If this was the case, I analyzed the full text of the paper. Due to the manual search process, I have to deal with the threat of validity that my starting set of papers does not contain all relevant literature, because it was published in a source that I did not consider or was published earlier than in the last 10 years, To mitigate this threat, I applied backward snowballing. That means, I also considered the papers referenced in the papers that I identified as relevant until no new candidates were found. In total, I identified 403 papers that seemed to be relevant after reading title and abstract. After the analysis of the full text, I finally identified 39 papers as related work.

I was able to map each explicitly mentioned transparency related concept in the literature to an element of my taxonomy. I categorized the identified literature into the four categories *Empirical Research on Privacy Awareness* (see Table 5.2), *Privacy (Requirements) Engineering* (see Table 5.3), *Privacy from the Legal Perspective* (see Table 5.4), and *Privacy Policies and Obligations* (see Table 5.5).

The papers in the category *Empirical Research on Privacy Awareness* (see Table 5.2) mainly investigate the users' awareness of data processing. The papers did not give recommendations on how data subjects shall be informed about exceptional cases.

From Table 5.3, we can see that almost all papers in the category *Privacy (Requirements) Engineering* have considered *what* information has to be provided to data subjects, but only the half of these papers mentioned that it is important *how* this information is provided. Only three contained aspects related to notification of data subjects in exceptional cases, e.g., data breaches. Note that none of the papers in this category covered all elements of my taxonomy.

The category *Privacy from the Legal Perspective* (see Table 5.4) contains papers that consider single laws or aspects that can be reflected by single elements of my taxonomy, and papers that consider a larger legal framework or privacy impact assessments and hence, cover (almost) all elements of my taxonomy.

[3] `http://www.core.edu.au/conference-portal` (accessed on 15 January 2018)

Table 5.2.: *Mapping of transparency notions from empirical research on privacy awareness to my proposed taxonomy*

Source	PR	EIR	PIR	SIR	FIR	CIR
Reinfelder et al. (2014)			X	X		X
Sheth et al. (2014)	X		X	X	X	X
Zviran (2008), Sheehan and Hoy (2000)			X			X

PR: PresentationRequirement **EIR:** ExceptionalInformationRequirement
SIR: StorageInformationRequirement **PIR:** ProcessingInformationRequirement
FIR: FlowInformationRequirement **CIR:** CollectionInformationRequirement

Table 5.3.: *Mapping of transparency notions from the privacy (requirements) engineering literature to my proposed taxonomy*

Source	PR	EIR	PIR	SIR	FIR	CIR
Breaux (2014)	X					
Deng et al. (2011)		X	X	X	X	X
Feigenbaum et al. (2002)	X		X			X
Fhom and Bayarou (2011)			X	X	X	
Hedbom (2009)			X	X	X	X
Hoepman (2014)		X	X	X	X	
Kung et al. (2011)			X	X	X	X
Langheinrich (2001)	X					X
Masiello (2009)	X		X	X	X	X
Miyazaki et al. (2008)		X	X			
Mouratidis et al. (2013); Kalloniatis et al. (2014)	X		X			
Pötzsch (2009)	X		X		X	
Rost and Pfitzmann (2009), Hansen (2012), Bier (2013)			X	X	X	X
Spiekermann and Cranor (2009)	X		X		X	X
Wicker and Schrader (2011)	X		X	X	X	X

PR: PresentationRequirement **EIR:** ExceptionalInformationRequirement
SIR: StorageInformationRequirement **PIR:** ProcessingInformationRequirement
FIR: FlowInformationRequirement **CIR:** CollectionInformationRequirement

The papers in the category *Privacy Policies and Obligations* (see Table 5.5) provide the most structured, detailed, and complete concepts related to transparency requirements. Nevertheless, I did not find any literature that provides an as structured, detailed, and complete overview of transparency requirements as my proposed taxonomy shown in Section 5.2.

Table 5.4.: *Mapping of transparency notions from literature with a focus on privacy from the legal perspective to my proposed taxonomy*

Source	PR	EIR	PIR	SIR	FIR	CIR
Breaux and Gordon (2013), Tomaszewski (2006)		X				
Jones and Tahri (2010)			X			
Mulligan (2014), Wright (2012)	X			X	X	X
Otto et al. (2007)	X	X	X	X	X	X
Solove (2006)		X	X		X	
Sype and Seigneur (2014)	X	X	X	X	X	X
Wright and Raab (2014)	X					X

PR: PresentationRequirement **EIR:** ExceptionalInformationRequirement
SIR: StorageInformationRequirement **PIR:** ProcessingInformationRequirement
FIR: FlowInformationRequirement **CIR:** CollectionInformationRequirement

Table 5.5.: *Mapping of transparency notions from privacy policies and obligations literature to my proposed taxonomy*

Source	PR	EIR	PIR	SIR	FIR	CIR
Alcalde Bagüés et al. (2008)		X	X	X	X	
Antón et al. (2002); Antón and Earp (2004); Antón et al. (2007)	X	X	X	X	X	X
Casassa Mont (2004)	X	X	X	X	X	X
Kelley et al. (2009, 2010)	X		X		X	
Lobato et al. (2009)	X	X	X	X	X	X

PR: PresentationRequirement **EIR:** ExceptionalInformationRequirement
SIR: StorageInformationRequirement **PIR:** ProcessingInformationRequirement
FIR: FlowInformationRequirement **CIR:** CollectionInformationRequirement

5.4. Conclusions

In this chapter,

1. I systematically derived requirements for the privacy goal transparency from the ISO 29100 standard (ISO/IEC, 2011) and the draft of the EU Data Protection Regulation (European Commission, 2012). These two documents belong to the most relevant sources for privacy requirements that have to be considered by software developers.

2. I then structured these requirements in a metamodel for transparency requirements. This metamodel provides an overview of the identified kinds of transparency requirements and shall help requirements engineers to identify and document the transparency requirements relevant for them and the information needed to address the transparency requirements.

3. I performed a systematic literature review and provided an overview of the relevant research related to transparency requirements.

4. I validated that my taxonomy contains all necessary aspects mentioned in the identified literature.

The literature review showed that all aspects of the privacy goal transparency mentioned in the literature are reflected in the proposed taxonomy, i.e., I did not find a privacy-related transparency aspect that could not be mapped to one of my proposed transparency requirements. Furthermore, I did not find any literature that presents transparency requirements in an as structured, detailed, and complete manner. My proposed metamodel of the taxonomy can easily be adopted and extended.

In the next Chapter, I extend the metamodel for transparency requirements with intervenability requirements that are tightly related to the transparency requirements, because intervenability requirements refine the abstract controlOptions associated to ProcessingInformationRequirements (cf. Figure 5.2). I introduce human-readable representations of instantiated transparency requirements based on text templates in Chapter 7. Furthermore, I present a systematic and computer-aided method to derive the relevant transparency requirements for a software-to-be based on its functional requirements in Chapter 12.

Chapter 6

Refining the Privacy Goal Intervenability

In this chapter, I present a taxonomy that refines the privacy goal intervenability into more fine-grained, still non-functional, privacy requirements. The work presented in this chapter is already published in (Meis and Heisel, 2016c) and (Meis and Heisel, 2017b). Co-author of these paper is Maritta Heisel, who provided helpful comments to improve the research. The presented refinement process, final taxonomy, and the literature review are my own contributions.

This chapter is structured as follows. My intervenability requirements taxonomy is derived and presented in Section 6.2. Then it is validated using related work identified using a systematic literature review in Section 6.3. Section 6.4 concludes this chapter.

6.1. Introduction

A central concern of end-users with regard to privacy is that they have almost no control over their personal data once these are put into an information system (GSMA, 2014; Symantec, 2015; Quah and Röhm, 2013; Reagle et al., 1999). End-users wish for more empowerment, i.e., they want to keep the control over their personal data and how their data are processed by information systems. Hansen (2012) summarizes this and other privacy needs into the privacy goal *intervenability*. Hansen states *"Intervenability aims at the possibility for parties involved in any privacy-relevant data processing to interfere with the ongoing or planned data processing. The objective of intervenability is the application of corrective measures and counterbalances where necessary."* (Hansen, 2012)

Intervenability is a complex software quality that is strongly coupled with other privacy-related goals. For example, end-users have to be sufficiently aware of how and what personal data are processed and which options exist to intervene in order to be able to exercise these options. Hence, the privacy goal transparency can be seen as prerequisite for intervenability.

As a first step to assist requirements engineers to deal with the complex privacy goal intervenability, I propose a requirements taxonomy that further refines intervenability into subrequirements enriched with attributes and associated to transparency requirements that I identified in Chapter 5. The taxonomy shall help requirements engineers to understand which intervenability and transparency requirements have to be considered for the system they analyze.

6.2. Deriving and Structuring Requirements on Intervenability

In Section 6.2.1, I systematically analyze the privacy principles described by the international standard ISO 29100 (ISO/IEC, 2011) and the EU data protection regulation (European Commission, 2016) to derive the intervenability requirements they contain and the transparency requirements related to them. To derive the requirements, I analyze the description of the privacy principles and the formulations of the regulation. I keep the formulation of the identified intervenability and transparency requirements close to the original documents from which I identified them. In Section 6.2.1, I enumerate these derived requirements using the notation In for intervenability requirements and Tn for the related transparency requirements. Note that the numbering of the transparency requirements introduced in this chapter does not correspond to the numbering in Chapter 5, because the transparency requirements elicited in this chapter focus more on intervenability. However, the transparency requirements identified in this chapter and those from Chapter 5 are related, which becomes visible in the combined taxonomy of transparency and intervenability requirements (see Section 6.2.2).

As the ISO principles and EU articles partly overlap, I identified several refinements of identified requirements. I relate those requirements using a *refines* relation. If an intervenability requirement In_1 refines a part of another requirement In_2, this means that In_1 adds further details on how or which possibilities have to exist to intervene in the processing of personal data. Furthermore, I identified that there are transparency requirements that are closely related to intervenability requirements. This is, because in order to be able to make use of intervenability mechanisms, data subjects have to be aware of them. Hence, I use a *relatedTo* relation to make the relations between transparency and intervenability requirements explicit. The *refines* (directed dashed edges) and *relatedTo* (solid edges) relation are visualized as an initial overview of intervenability requirements in Figure 6.1. In Section 6.2.2, I structure the intervenability requirements identified in Section 6.2.1 into a taxonomy of intervenability requirements and integrate this taxonomy into the taxonomy of transparency requirements introduced in Chapter 5. The taxonomy is presented as an extensible metamodel as a UML class diagram.

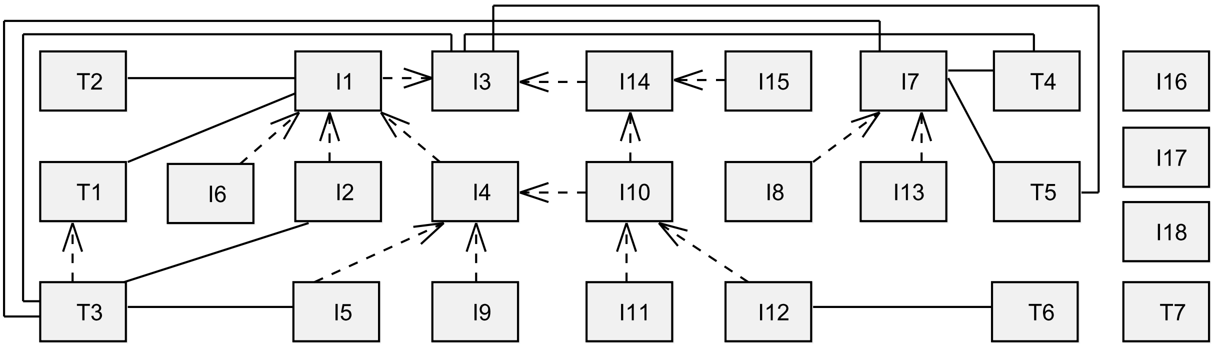


Figure 6.1.: *Initial overview of intervenability requirements*

ISO 29100 and the EU data protection regulation do not use the same terminology. To avoid ambiguities, I use the terms introduced in Section 2.4.2 on page 25.

6.2.1. Requirements Identification from Privacy Principles and Legislation

In the following two subsections, I derive from the international standard ISO 29100 (ISO/IEC, 2011) (see Section 6.2.1.1) and the EU Data Protection Regulation (European Commission, 2016) (see Section 6.2.1.2) initial transparency and intervenability requirements.

6.2.1.1. ISO 29100 Privacy Principles

The international standard ISO 29100 (ISO/IEC, 2011) defines 11 privacy principles which are a superset of the OECD principles (OECD, 1980) and the US fair information practices (FIPs) (US Federal Trade Commission, 1998) (cf. Section 2.4.3).

I start my analysis with the *consent and choice principle*, which is obviously concerned with providing data subjects the power to decide how their data are processed. From this principle, I obtain the following intervenability and transparency requirements.

I1 Present to the data subjects the choice whether or not to allow the processing of their personal data.

I2 Obtain the opt-in consent of the data subject for collecting or otherwise processing sensitive personal data.

T1 Inform data subjects before obtaining consent about their rights to access their personal data and to influence the processing of these.

I3 Provide data subjects with the opportunity to choose how their personal data are handled.

I4 Allow data subjects to withdraw consent easily and free of charge.

T2 Where the personal data processing is not based on consent, but instead on another legal basis, the data subject should be notified wherever possible.

I5 Where the data subject has the ability to withdraw consent and has chosen to do so, these personal data should be exempted from processing for any purpose not legally mandated.

I6 Provide data subjects with clear, prominent, easily understandable, accessible and affordable mechanisms to exercise choice and to give consent in relation to the processing of their personal data at the time of collection, first use or as soon as practicable thereafter.

Requirement I3 states that data subjects shall have the opportunity to choose how their data are handled and is the most general intervenability requirement. It is refined by I1 (cf. Figure 6.1) that states that data subjects shall have the choice whether their data are processed or not. I1 is further refined by I2 that requires opt-in consent for processing of sensitive personal data, I4 that requires the possibility to withdraw consent, and I6 that describes requirements for the mechanisms to realize I1. I5 refines I4 by describing the effects of withdrawing consent. Both transparency requirements T1 and T2 are related to I1 (cf. Figure 6.1). T1 requires that data subjects have to be informed about their rights before consent is obtained. T2 requires to inform data subjects if their data are processed without their explicit consent.

From the *openness, transparency and notice principle* I identify an additional transparency requirement that is related to all intervenability requirements that describe the choices and means for data subjects to influence how their data are processed (cf. Figure 6.1).

T3 Disclose the choices and means offered by the controller to data subjects for the purposes of limiting the processing of, and for accessing, correcting and removing their personal data.

The following two intervenability requirements are derived from the *individual participation and access principle.*

I7 Give data subjects the ability to access and review their personal data, provided their identity is first authenticated with an appropriate level of assurance and such access is not prohibited by applicable law.

I8 Allow data subjects to challenge the accuracy and completeness of their personal data and have them amended, corrected or removed as appropriate and possible in the specific context.

I7 and I8 are not refinements of the already identified intervenability requirements, because they are not concerned with how data subjects can influence how or if their personal data are processed. But I consider I8 as a kind of refinement of I7, because I8 depends on I7. Note that I7 prescribes that data subjects shall be empowered with the ability to access and review their personal data. Hence, I7 is considered as an intervenability requirement. Nevertheless, allowing data subjects to access and review their personal data also contributes to transparency.

The other principles presented in ISO 29100 do not contain further statements from which I can derive intervenability requirements.

6.2.1.2. EU Data Protection Regulation

To identify further intervenability and transparency requirements and to refine the already identified requirements, I analyze the EU data protection regulation (European Commission, 2016). I selected this regulation as a representative data protection regulation.

Article 7 describes the conditions for consent and I derive from it the following intervenability requirement that refines I4.

I9 The data subject shall have the right to withdraw his or her consent at any time.

Article 12 specifies requirements on mechanisms for exercising the rights of data subjects. I identified the following two transparency requirements that are related to all intervenability requirements that describe the choices and means for data subjects to influence how their data are processed.

T4 The controller shall inform the data subject without delay and, at the latest within one month of receipt of the request, whether or not any action has been taken if a data subject requested information and shall provide the requested information.

T5 If the controller refuses to take action on the request of the data subject, the controller shall inform the data subject of the reasons for the refusal and on the possibilities of lodging a complaint to the supervisory authority and seeking a judicial remedy.

Article 15 describes the right of access by the data subject. I do not derive further requirements from Article 15, but it supports the intervenability requirement I7 and transparency requirement T3.

The right to rectification is defined in Article 16. This right is already covered by the intervenability requirement I8.

Article 17 is about the right to be forgotten and to erasure. From this article I derive the following requirements.

I10 The data subject shall have the right to obtain from the controller the erasure of personal data relating to them and the abstention from further dissemination of such data if the data subject withdraws consent or objects to the processing of personal data.

I11 The controller shall carry out the erasure without delay, except to the extent that the retention of the personal data is necessary.

I12 Where erasure is not possible, the controller shall instead restrict processing of personal data.

T6 The controller shall inform the data subject before lifting the restriction on processing.

I10, I11, and I12 refine the consequence of withdrawing consent (I4) and objecting to processing (I14 see below). T6 requires that data subjects are informed about the restrictions on processing implied by I12 before these are lifted.

The right to data portability is introduced by Article 20. It implies the following intervenability requirement that refines I7.

I13 The data subject shall have the right, where personal data are processed by electronic means and in a structured and commonly used format, to obtain from the controller a copy of data undergoing processing in an electronic and structured format which is commonly used and allows for further use by the data subject.

Article 18 is about the right to restriction of processing. This right overlaps with the right to object described in Article 21. From these Articles, I derived the following two intervenability requirements that refine I3.

I14 The data subject shall have the right to object, on grounds relating to their particular situation, at any time to the processing of personal data, unless the controller demonstrates compelling legitimate grounds for the processing.

I15 If the objection is valid, the controller shall no longer use or otherwise process the personal data concerned.

Article 58 describes the powers of supervisory authorities. In contrast to the previously identified requirements, the following requirements do not describe intervention possibilities for data subjects or needs to provide information to data subjects, but for and to supervisory authorities, respectively.

T7 Supervisory authorities may order the controller to provide any information relevant for the performance of their duties to them.

I16 Supervisory authorities may order the rectification or erasure of all data when they have been processed in breach of the provisions of a regulation.

I17 Supervisory authorities may impose a temporary or definitive ban on processing.

I18 Supervisory authorities may order to suspend data flows to a recipient in a third country or to an international organization.

Table 6.1 summarizes from which ISO 29100 principles and articles of the draft of the EU data protection regulation which initial intervenability and transparency requirements were derived. Additionally, it allows to associate the elements of my intervenability requirements taxonomy (introduced in the next section) with the principles and articles from which these were identified.

6.2.2. Setting up an Intervenability Requirements Taxonomy

In this section, I structure the identified preliminary intervenability requirements into an intervenability requirements taxonomy. I integrate this taxonomy into the transparency requirements taxonomy presented in Chapter 5 using the related preliminary transparency requirements. Figure 6.2 shows my taxonomy in the form of a metamodel as a UML class diagram. Note that I only show the attributes and enumerations of the transparency taxonomy that are relevant for the introduction of the intervenability requirements taxonomy. All elements that have bold font and thick lines are newly added to the transparency taxonomy. The requirements with dark gray background represent the newly identified transparency and intervenability requirements.

Table 6.2 provides an overview of how the initial requirements are reflected in my proposed taxonomy. In the following, I explain the new elements of my taxonomy and how they are related to the requirements introduced in Chapter 5.

Table 6.1.: *Mapping of ISO principles and data protection articles to the requirements*

Principle/Article	**I*n*/T*n***	**IR**	**DIR**	**AIR**	**PIR**	**EIR**	**IIR**
Consent and choice	I1-I6, T1, T2	X	X		X		
Openness, transparency and notice	T3		X		X		
Individual participation and access	I7, I8	X	X				
Article 7	I9		X				
Article 12	T4, T5						X
Article 15	I7, T3	X	X		X		
Article 16	I8	X	X				
Article 17	I10-I12, T6	X	X				X
Article 20	I13	X	X				
Article 18 and 21	I14, I15	X	X				
Article 58	I16-I18, T7	X		X		X	

IR: IntervenabilityRequirement **DIR**: DataSubjectInterventionRequirement
AIR: AuthorityInterventionRequirement **PIR**: ProcessingInformationRequirement
EIR: ExceptionalInformationRequirement **IIR**: InterventionInformationRequirement

Table 6.2.: *Mapping between taxonomy and preliminary requirements*

Requirement	**Attribute**	**I*n*/T*n***
IntervenabilityRequirement	effect	I1, I3, I5, I7, I8, I10-I13, I15-I18
DataSubjectInterventionRequirement	type	I1-I5, I7, I8, I10-I15
	time	I6, I9, I14
	consequences	T1, T3, I6
AuthorityInterventionRequirement	type	I16, I17, I18
ProcessingInformationRequirement	controlOptions	T1, T3, I6
	grounds	T2
ExceptionalInformationRequirement	exceptionalCase	I16, I17, I18, T7
InterventionInformationRequirement		T4, T5, T6

6.2.2.1. Intervenability Requirement

The root element of my intervenability requirements taxonomy is the *IntervenabilityRequirement*. I modeled it as an abstract class because only its specializations shall be instantiated. It contains the attribute effect that describes the consequences of an intervenability requirement. The possible effects are derived from the preliminary requirements I1, I3, I5, I7, I8, I10-I13, and I15-I18, and are summarized in the enumeration InterventionEffect (cf. Figure 6.2). The effects are that data subjects get *access* to their personal data, that their personal data are *not processed*, that the *processing is restricted*, that their personal data are *amended*, *corrected*, or *erased*, that they *receive a copy* of their data, and that *data flows are suspended*. In addition to the effect that an intervenability requirement shall have, it has a type describing how data subjects or supervisory authorities can cause the wanted effects. As these types differ for data subjects and authorities, I added the attribute type to the intervenability requirements DataSubjectInterventionRequirement (representing intervention possibilities for data subjects) and AuthorityInterventionRequirement (representing intervention possibilities for supervisory authorities).

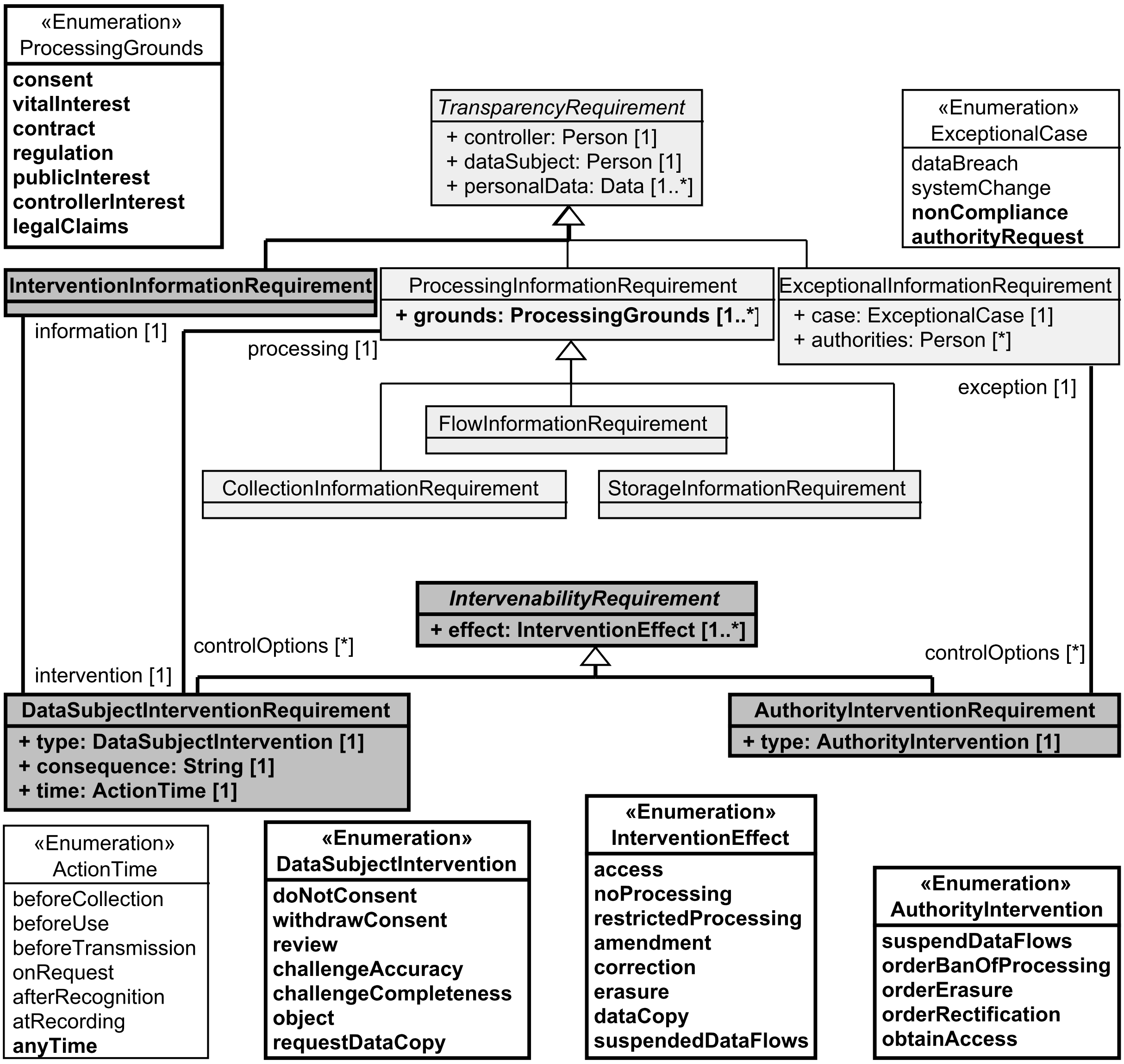


Figure 6.2.: *My combined taxonomy of transparency and intervenability requirements.*

6.2.2.2. AuthorityInterventionRequirement

Almost all initial requirements describe rights of data subjects to influence how their personal data are processed. Only I16, I17, I18, and T7 present possibilities for supervisory authorities to intervene in the processing of personal data. The intervention types for authorities are summarized in the enumeration AuthorityIntervention (cf. Figure 6.2). Supervisory authorities may order to *suspend data flows*, order a *ban of processing* of personal data, and order the *erasure* or *rectification* of personal data. The initial requirements I16, I17, and I18 also describe which type of intervention shall lead to which kind of intervention effect. Hence, there are limitations for the combination of intervention types and effects when an AuthorityInterventionRequirement is instantiated. Table 6.3 presents the valid combinations of intervention types and effects.

T7 indicates that supervisory authorities have to be informed about the processing in order to exercise their rights to intervention properly. Hence, each AuthorityInterventionRequirement has an ExceptionalInformationRequirement assigned that describes which supervisory authorities may intervene. I newly introduced into the enumeration ExceptionalCase the literals nonCompliance and authorityRequest to reflect that authorities have to be informed in the case of processing

Table 6.3.: *Mapping between authority intervention types and intervention effects*

Intervention Type	Possible Intervention Effects	Source
suspendDataFlows	suspendedDataFlows	I18
orderBanOfProcessing	noProcessing, restrictedProcessing	I17
orderErasure	erasure	I16
orderRectification	correction, amendment	I16
obtainAccess	access	T7

of personal data in a way that does *not comply* with the regulations and that authorities then have the possibility to intervene in this processing. Additionally, authorities have the right to *request* information concerning the processing of personal data from the controller. This may also lead to situations where authorities need to *obtain access* to the personal data processed by the controller to decide whether the controller processes the data subjects' personal data in compliance with the applicable regulations.

6.2.2.3. DataSubjectInterventionRequirement

The DataSubjectInterventionRequirement presents the possibilities for data subjects to intervene in the processing of their personal data. These possibilities are summarized in the enumeration DataSubjectIntervention (cf. Figure 6.2) that I derived from the preliminary requirements I1-I5, I7, I8, and I10-I15. These initial requirements additionally describe which combinations of intervention types and effects are allowed for DataSubjectInterventionRequirements. The valid combinations are shown in Table 6.4.

Table 6.4.: *Mapping between data subject intervention types and intervention effects*

Intervention Type	Possible Intervention Effects	Source
doNotConsent	noProcessing	I2
withDrawConsent	noProcessing, restrictedProcessing, erasure	I4, I5, I10, I12
review	access	I7
challengeAccuracy	correction, amendment, erasure	I8
challengeCompleteness	amendment, erasure	I8
object	noProcessing, restrictedProcessing, erasure	I10, I12, I15
requestDataCopy	dataCopy	I13

T1, T3, I6, and I9 require that data subjects have to be informed about how they can intervene in the processing of their personal data. To reflect this, I introduced the association controlOptions between DataSubjectInterventionRequirement and ProcessingInformationRequirement (cf. Figure 6.2). From the perspective of the ProcessingInformationRequirement, the association describes which options exist for data subjects to intervene in the processing of their personal data. The two attributes consequence and time of DataSubjectInterventionRequirement are used to describe further details on the control option described by the DataSubjectInterventionRequirement. The attribute consequences allows to provide a textual description of the consequences that the utilization of the corresponding intervenability option has. The attribute time describes when data subjects can exercise the corresponding option. Hence, the association controlOptions in Figure 6.2 refines the attribute controlOptions in Figure 5.2 on page 78, and the class DataSubjectInterventionRequirement in Figure 6.2 replaces the data type ControlOption in Figure 5.2 on page 78.

6.2.2.4. InterventionInformationRequirement

From the preliminary requirements T4-T6, I identified that an additional transparency requirement should be added to the taxonomy. This requirement states the need to inform data subjects about the progress or rejection of interventions requested by them. For this purpose, I introduce the InterventionInformationRequirement. Each DataSubjectInterventionRequirement is associated to an InterventionInformationRequirement and vice versa that presents the need to inform data subjects about the progress or rejection of their intervention.

Furthermore, I identified from T2 that the ProcessingInformationRequirement should also inform data subjects about the legal grounds on which their data are processed. For this, I enriched this requirement with an attribute grounds that reflects the possible grounds for processing personal data by the controller. These are derived from ISO 29100 and the EU data protection regulation. They are *consent* of the data subject, the *vital interest* of the data subject, an existing *contract*, a *regulation* that allows the processing, and *public interest*.

6.3. Validation of the Taxonomy Using Related Literature

In this section, I give an overview of existing research that also contains considerations about the privacy goal of intervenability. To validate my proposed taxonomy, I map the notions and concepts used in the related literature to my taxonomy to check whether it is suitable to reflect the intervenability concepts used in the literature.

To identify the relevant related work, I performed a systematic literature review using backward snowballing (Jalali and Wohlin, 2012). To obtain the starting set of papers for my review, I manually searched the proceedings and issues of the last 10 years of computer science conferences and journals that are mainly concerned with at least one of the topics privacy, requirements, and software engineering and ranked at least as *B-level* in the CORE2014[1] ranking. In this way, I selected 15 conferences and 19 journals. First, I checked whether title or abstract of a paper indicates that the paper is concerned with privacy (requirements), intervenability, empowerment, user's controls, or user's choices. In this way, I obtained 219 papers. I then analyzed the full texts of these papers. Doing this, I reduced the number of relevant papers to 21. Due to the manual search process, I have to deal with the threat to validity that my starting set of papers does not contain all relevant literature, because it was published in a source that I did not consider or was published earlier than in the last 10 years. To mitigate this threat, I applied backward snowballing. That is, I also considered the papers referenced in the papers that I identified as relevant until no new candidates were found. During the snowballing, I identified 79 possibly relevant papers from which 12 were finally considered as relevant. In total, I identified 298 papers that seemed to be relevant after reading title and abstract. After the analysis of the full text, I finally identified 33 papers as related work.

The most important finding is that I was able to map each explicitly mentioned intervenability-related concept in the literature to an element of my taxonomy and that none of the papers provides such a structured overview of intervenability requirements and relates these explicitly to transparency requirements. Table 6.5 shows to which degree the papers identified during the literature review address the intervenability requirements that I identified in this work. For each paper, I investigated to which degree aspects of the DataSubjectInterventionRequirement (column **DIR**), the AuthorityInterventionRequirement (column **AIR**), and the relations between intervenability and transparency requirements (column **RIT**) are mentioned in it. I distinguish in Table 6.5 three cases. If all aspects are addressed, I denote this with a "+". If the aspects are only partially considered, then I denote this with a "o". If no aspects are addressed, I denote this with a "-".

[1] `http://www.core.edu.au/conference-portal` (accessed on 15 January 2018)

Table 6.5.: *Mapping of intervenability notions from the literature to my taxonomy*

Source	DIR	AIR	RIT
Bier (2013), Hansen (2012)	+	+	o
Hoepman (2014)	+	o	o
Mouratidis et al. (2013)	o	o	o
Miyazaki et al. (2008)	+	+	-
Kalloniatis et al. (2014); Kalloniatis (2015), Spiekermann and Cranor (2009)	o	o	-
Makri and Lambrinoudakis (2015), Acquisti et al. (2013), Masiello (2009), Krol and Preibusch (2015), Deng et al. (2011), Komanduri et al. (2011), Cranor (2012), Wicker and Schrader (2011)	o	-	o
Strickland and Hunt (2005), Sheth et al. (2014), Fhom and Bayarou (2011), Antón et al. (2002); Antón and Earp (2004), Sype and Seigneur (2014), Basso et al. (2015)	+	-	-
Lobato et al. (2009), Caron et al. (2016), Borgesius (2015), Breaux (2014), Langheinrich (2001), Feigenbaum et al. (2002), Wright and Raab (2014), Guarda and Zannone (2009), Hedbom (2009), Smith et al. (2011)	o	-	-

DIR: DataSubjectInterventionRequirement, **AIR**: AuthorityInterventionRequirement
RIT: Relation between intervenability and transparency requirements

From Table 6.5, we can see that no paper discusses all aspects concerning the relation between intervenability and transparency requirements. Several papers mention that transparency is a prerequisite for intervenability or that data subjects have to be aware of their options to intervene in the processing of their personal data, but none of the papers mentions that data subjects have to be informed about the progress of the intervention requests they have triggered. Few of the papers consider the intervention options of supervisory authorities. Only three papers covered all of the aspects and 5 identified the need to be able to answer requests of supervisory authorities in order to prove compliance with regulations or standards. All papers discuss at least partially options for the data subject to intervene in the processing of their personal data. The most often discussed intervenability option is to consent or withdraw consent. Another interesting observation that I made is that only Hoepman (2014) discusses the right to data portability. This right, its implementation, and consequences have not yet been discussed deeply in the literature.

6.4. Conclusions

In this chapter,

1. I systematically derived requirements for the privacy goal intervenability and related transparency requirements from the ISO 29100 standard (ISO/IEC, 2011) and the EU data protection regulation (European Commission, 2016).

2. I then integrated these requirements into the metamodel for transparency requirements presented in Chapter 5. The new metamodel provides an overview of the identified kinds of transparency and intervenability requirements and how these are related to each other. The metamodel shall furthermore help requirements engineers to identify and document the transparency and intervenability requirements relevant for them and the information needed to address the transparency and intervenability requirements.

3. I performed a systematic literature review and provided an overview of the relevant research related to intervenability requirements.

4. I validated that my taxonomy contains all necessary aspects mentioned in the identified literature. The literature review showed that all aspects of the privacy goal intervenability mentioned in the literature are reflected in the proposed taxonomy, i.e., I did not find a privacy-related intervenability aspect that could not be mapped to one of my intervenability requirements. Furthermore, I did not find any literature that presents intervenability requirements and their relation to transparency requirements in such a structured, detailed, and complete manner.

I believe that my taxonomy is flexible enough to also represent intervenability and transparency requirements from other regulations and standards, because my proposed metamodel of the taxonomy can easily be adopted and extended. In these cases, my metamodel can be enhanced with, e.g., further intervention types and effects. These can easily be added to the corresponding enumerations (cf. Figure 6.2).

I introduce human-readable representations of instantiated intervenability and transparency requirements based on text templates in Chapter 7. Furthermore, I present a systematic and computer-aided method to derive the relevant intervenability and transparency requirements for a software-to-be based on its functional requirements in Chapter 12.

Chapter 7

A Taxonomy of Privacy Requirements

This chapter provides an overview of the privacy requirements that form the privacy taxonomy of the problem-based privacy analysis (ProPAn) method (see Chapter 8). This taxonomy refines the privacy protection goals proposed by Hansen et al. (2015) (see Chapter 2.4.4). The protection goals *transparency* and *intervenability* are refined by the taxonomies introduced and derived in the previous two chapters, the security goals *confidentiality*, *integrity*, and *availability* are not further refined and taken as privacy requirements, and the protection goal *unlinkability* is refined based on the work of Pfitzmann and Hansen (2010).

This chapter has four contributions. First, it shows how the privacy requirements with their properties and relations can be represented in an EMF metamodel that allows to integrate the requirements into the tool support of the ProPAn method. Second, I propose for each privacy requirement a text template that defines the meaning of an instantiated privacy requirement. Third, I provide validation conditions in natural language and as OCL constraints that specify properties that valid instances have to have and consistency conditions between different privacy requirements. Earlier versions of these templates and validation conditions are published in (Meis and Heisel, 2016b) and (Meis and Heisel, 2017b). I am the main author of both papers and Maritta Heisel gave valuable feedback that helped to improve the papers. The first three contributions are presented in Section 7.2. Fourth, I discuss other privacy goals and privacy taxonomies identified during the literature review shown in Chapter 3 and their relation to ProPAn's privacy requirements taxonomy in Section 7.3. I give an introduction to this chapter in Section 7.1 and it is concluded in Section 7.4.

7.1. Introduction

As shown in my literature review on the state of the art in privacy requirements engineering (see Chapter 3) and in the background of my thesis (see Section 2.4), there are different conceptual models of privacy, e.g., privacy principles and privacy protection goals. In this chapter, I introduce the privacy conceptual model of my thesis and the ProPAn method in the form of a (non-functional) privacy requirements taxonomy. This taxonomy refines the privacy protection goals proposed by Hansen (2012) and hence, represents more technical privacy requirements than privacy principles that describe guidelines to reach compliance with data protection legislations. I discuss in Section 7.3 how my privacy requirements taxonomy is related to other privacy conceptual models and how they may complement each other.

I describe my privacy requirements taxonomy as an EMF model. This allows me to arrange the privacy requirements (represented as classes) into a hierarchy, to enrich them with attributes and to specify relations between them. Furthermore, I present validation conditions that allow to detect errors in the specification of privacy requirements and inconsistencies between them.

Listing 7.1: *Concrete syntax for a conditional expression in the text templates*

```
<if condition="booleanexpr">
  bodyexpr
</if>
```

Listing 7.2: *Concrete syntax for a case distinction over enumeration literals in the text templates*

```
<switch enumeration="enumexpr">
  <case literal="literalexpr1">
    caseexpr1
  </case>
  <case literal="literalexpr2">
    caseexpr2
  </case>
  ...
</switch>
```

Listing 7.3: *Concrete syntax for a loop expression in the text templates*

```
<foreach collection="collectionexpr" iterator="name">
  bodyexpr
</foreach>
```

These validation conditions are formalized as OCL invariants on the respective EMF classes.

To describe the meaning of instantiatable privacy requirements (represented by non-abstract classes in the EMF model), I use a simple template language that allows to embed OCL expressions into the text of the template. The OCL expressions can, e.g., be used to access attributes of the EMF class for which the template is evaluated. Additionally, conditional expressions (keyword **if**), case distinction over enumeration values (keywords **switch** and **case**) , and iteration over collections (keyword **foreach**). can be used in the template language following an XML-like notation. The concrete syntax for a conditional expression is shown in Listing 7.1. The body expression *bodyexpr* inside the opening and closing **if**-tags is only evaluated if its condition *booleanexpr* is evaluated to *true*. Listing 7.2 shows the concrete syntax for a case distinction over enumeration literals. The enumeration is specified by the expression *enumexpr* inside the **switch**-tag. For each possible literal of the enumeration, an expression *caseexpr* can be specified surrounded by **case**-tags that specify the literal with the expression *literalexpr*. Only the case expression *caseexpr* that is surrounded by **case**-tags with the respective literal value of the *enumexpr* is evaluated. The concrete syntax for the iteration over collection is shown in Listing 7.3. The expression *collectionexpr* specifies the collection that shall be iterated. For each element of the collection the body expression *bodyexpr* is evaluated once. To refer to the currently considered element in the body expression the iterator name *name* can be used.

7.2. Privacy Requirements Taxonomy

In this section, I introduce my taxonomy. The root element of my taxonomy is introduced in Section 7.2.1. Section 7.2.2 introduces the security-related privacy requirements. The requirements refining the protection goal unlinkability are introduced in Section 7.2.3. The refinements of the protection goals transparency and intervenability that I already introduced in Chapter 5 and Chapter 6 are presented in Section 7.2.4 and Section 7.2.5, respectively. Each subsection first introduces the metamodel for the respective privacy requirements. Then the meaning of the instantiatable privacy requirements is specified using text templates. Finally, validation conditions for the privacy requirements are introduced, both in natural language and as OCL constraints in Appendix A.

7.2.1. Top-Level Privacy Requirement

7.2.1.1. Metamodel for the Top-level Privacy Requirement

In (Beckers et al., 2014b), I derived from the common criteria (ISO and IEC, 2009) that all privacy requirements mentioned there have in common that they state which *personal data* of which *data subject* shall be protected against which *counterstakeholders* (for the terminology see Section 2.4.2). Hence, I propose as root for my privacy taxonomy the abstract privacy requirement shown in Figure 7.1 as EMF model. All other privacy requirements are subclasses of *PrivacyRequirement*, which itself is a subclass of SoftwareQuality (see also Figure 2.10 on page 22), because the privacy requirements proposed in my taxonomy are not functional requirements, but software qualities. Each privacy requirement is for exactly one person[1], called dataSubject, and a non-empty set of personalData belonging to the data subject. Optionally, a collection of counterstakeholders can be assigned to a privacy requirement.

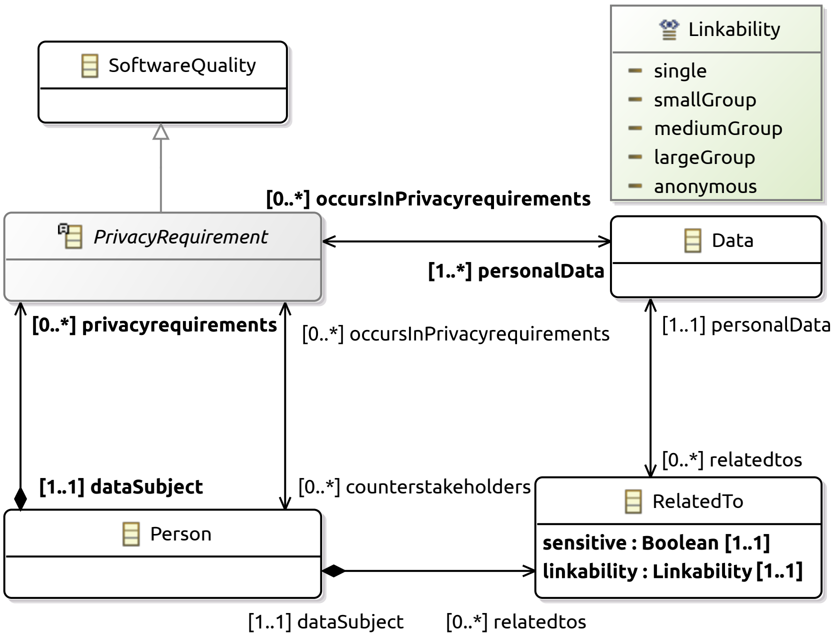


Figure 7.1.: *Abstract privacy requirement*

Figure 7.1 additionally shows the class RelatedTo. Instances of the class RelatedTo define which data are related to which person. I introduce the *related to* relation to make explicit that data can be personal data of different persons with different sensitivity and linkability to the data subject. The meaning of a related to instance is defined by the text template shown in Listing 7.4. Note that the size of small, medium, and large groups needs to be specified by the analysis team for a concrete privacy analysis.

[1] the class Person actually represents a group of persons and is later mapped to biddable domains (see Section 2.3)

Listing 7.4: *Text template defining the meaning of an instance of the class RelatedTo*

```
personalData are
<if condition="sensitive">
  sensitive
</if>
personal data of the dataSubject.
<switch enumeration="linkability">
  <case literal="Linkability::single">
    The personal data allow to identify data subject they are related to.
  </case>
  <case literal="Linkability::smallGroup">
    The personal data allow to narrow down the number of data subjects to whom the data belong to a small group.
  </case>
  <case literal="Linkability::mediumGroup">
    The personal data allow to narrow down the number of data subjects to whom the data belong to a medium group.
  </case>
  <case literal="Linkability::largeGroup">
    The personal data allow to narrow down the number of data subjects to whom the data belong to a large group.
  </case>
  <case literal="Linkability::anonymous">
    The personal data does not allow to narrow down the number of data subjects to whom the data belong to.
  </case>
</switch>
```

7.2.1.2. Validation Condition for the Top-level Privacy Requirement

To ensure that privacy requirements are consistent to the related to relations, the following validation condition needs to be checked.

VP1 The data referenced by a privacy requirement shall be personal data of the mentioned data subject.

7.2.2. Security-related Privacy Requirements

7.2.2.1. Metamodel for Security-related Requirements

ProPAn's privacy taxonomy includes the basic security requirements confidentiality, integrity, and availability, which I interpret in the context of privacy. Figure 7.2 shows the EMF model that defines the three security requirements *ConfidentialityRequirement*, IntegrityRequirement, and AvailabilityRequirement as subclasses of the abstract class *PrivacyRequirement*. The abstract class *ConfidentialityRequirement* is further refined into the DataConfidentialityRequirement. Note that the attributes dataSubject, counterstakeholders, and personalData of the class *PrivacyRequirement* correspond to relations as shown in Figure 7.1. To improve the readability of the diagrams, I present relations as attributes in all cases where this improves the readability of the diagram and the relations would not significantly improve the understandability of the diagram.

7.2.2.1.1. Confidentiality Requirement Class ISO 27000 (ISO/IEC, 2016) defines confidentiality as *"property that information is not made available or disclosed to unauthorized individuals, entities, or processes"*. In the context of privacy, the term *information* can be interpreted

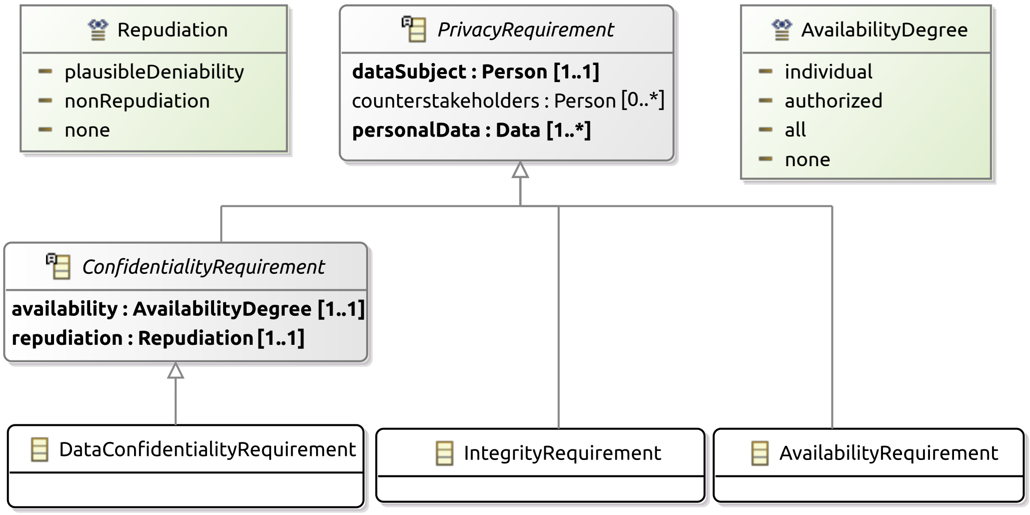

Repudiation
plausibleDeniability
nonRepudiation
none
PrivacyRequirement
dataSubject : Person [1..1]
counterstakeholders : Person [0..*]
personalData : Data [1..*]
AvailabilityDegree
individual
authorized
all
none
ConfidentialityRequirement
availability : AvailabilityDegree [1..1]
repudiation : Repudiation [1..1]
DataConfidentialityRequirement
IntegrityRequirement
AvailabilityRequirement

Figure 7.2.: *Used taxonomy of security requirements*

in different ways. First, the information can be personal data (data confidentiality). Second, the information about the relation of personal data to the respective data subject or to other personal data shall not be made available (unlinkability). Third, the information that personal data exists or not shall not be available to counterstakeholders (undetectability). For details on the last two, see Section 7.2.3.

All these confidentiality requirements are subclasses of the abstract class *ConfidentialityRequirement*. A confidentiality requirement has the two additional attributes availability and repudiation. The attribute availability allows to specify that the information shall be available to specific counterstakeholders. The attribute repudiation can be used to specify whether counterstakeholders to which the information is available shall be able to prove the correctness of the information, whether data subjects shall be able to deny this, or none of these two properties is needed.

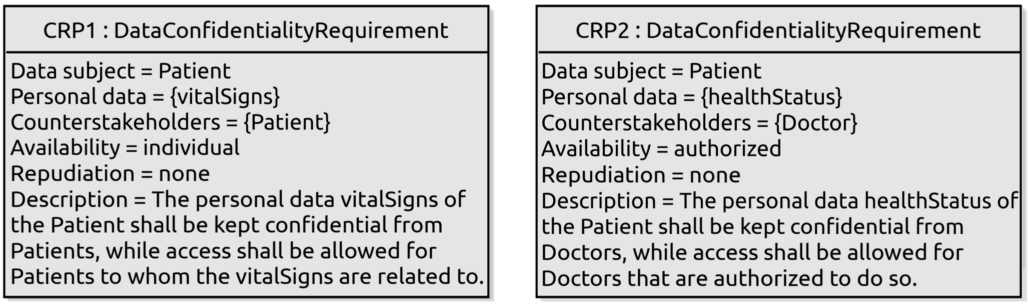
I introduce the attribute availability, because the type Person is later mapped to domains of the problem frame model, more specifically to biddable domains (see also Section 2.3). That means, the type Person does not represent an individual, but a group of individuals with similar characteristics, e.g., patients, doctors, and researchers in the EHS example (cf. Chapter 4). The attribute availability allows to formulate data confidentiality requirements such as, patient's vital signs shall be kept secret from other patients, but not from the individual patient him- or herself, or the patient's health status shall only be accessible to authorized doctors. Hence, the attribute availability allows to formulate more precisely for which individuals of the group of counterstakeholders the confidentiality requirement applies. The data confidentiality requirement instances for the above given examples are shown in Figure 7.3. Note that the attribute description is inherited from the abstract class *Statement* (cf. Figure 2.10 on page 22). This attribute can automatically set by the ProPAn tool based on the later introduced text templates.

The attribute repudiation is motivated from the security goal *non-repudiation* and the privacy goal *plausible deniability* that oppose each other. Deng et al. (2011) define these goals as follows:

Plausible deniability *"For privacy, plausible deniability refers to the ability to deny having performed an action that other parties can neither confirm nor contradict. Plausible deniability from an attackers perspective means that an attacker cannot prove a user knows, has done or has said something."*

Non-repudiation "Non-repudiation allows an attacker to gather evidence to counter the claims of the repudiating party and to prove that a user knows, has done or has said something."

The attribute repudiation can be used to specify whether data subjects shall be able to *plausibly deny* that the information available to the counterstakeholders is valid, counterstakeholders to whom the information is available can prove that the validity of it (*non-repudiation*), or *none* of both properties is needed. In this way, I encode the goals plausible deniability and non-repudiation into the requirements that state which information is allowed to be available to counterstakeholders.



CRP1 : DataConfidentialityRequirement

Data subject = Patient
Personal data = {vitalSigns}
Counterstakeholders = {Patient}
Availability = individual
Repudiation = none
Description = The personal data vitalSigns of the Patient shall be kept confidential from Patients, while access shall be allowed for Patients to whom the vitalSigns are related to.

CRP2 : DataConfidentialityRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {Doctor}
Availability = authorized
Repudiation = none
Description = The personal data healthStatus of the Patient shall be kept confidential from Doctors, while access shall be allowed for Doctors that are authorized to do so.

Figure 7.3.: *Examples of confidentiality requirement instances*

The enumeration AvailabilityDegree defines the possible values for the attribute availability. These are individual, i.e., all counterstakeholders to whom the personal data are related may have access to the information under consideration, authorized, i.e., specific counterstakeholders who are authorized to access the information under consideration may do this, and none, i.e., no counterstakeholder is allowed to access the respective information. An availability degree of all means that to all counterstakeholders the information may be available. This seems to contradict the intention of a confidentiality requirement. However, I allow to define such confidentiality requirements to make explicit that access to the respective information is intended and to specify the kind of (non-)repudiation for the counterstakeholders and data subjects.

7.2.2.1.2. Data Confidentiality Requirement Class DataConfidentialityRequirement is a non-abstract subclass of *ConfidentialityRequirement*. The information that shall not be available to counterstakeholders due to a data confidentiality requirement is the personal data specified by the attribute personalData.

7.2.2.1.3. Integrity and Availability Requirement Classes The classes IntegrityRequirement and AvailabilityRequirement are subclasses of *PrivacyRequirement* and have no additional attributes.

7.2.2.2. Text Templates for Security-related Requirements

7.2.2.2.1. Data Confidentiality Requirement Template The text template that defines the meaning of a DataConfidentialityRequirement is shown in Listing 7.5.

7.2.2.2.2. Integrity Requirement Template The meaning of an integrity requirement is described by the text template shown in Listing 7.6. Note that my definition of the security requirement integrity (and the following requirement availability) also include a dimension that is classically assigned to safety requirements, namely random faults that may cause harm to the system. For privacy, these random faults are also of relevance, because privacy issues not only arise from attacks a counterstakeholder might perform or unwanted incidents he or she incidentally causes, but also because of random faults that delete or change the content of the personal data of the data subject, or that do not allow access to information in the case of an availability requirement.

Listing 7.5: *Text template defining the meaning of a DataConfidentialityRequirement*

```
<switch enumeration="availability">
  <case literal="AvailabilityDegree::none">
    The personal data personalData of the dataSubject shall be kept confidential from counterstakeholders.
  </case>
  <case literal="AvailabilityDegree::individual">
    The personal data personalData of the dataSubject shall be kept confidential from counterstakeholders, while access shall be allowed for counterstakeholders to whom the personalData are related to.
  </case>
  <case literal="AvailabilityDegree::authorized">
    The personal data personalData of the dataSubject shall be kept confidential from counterstakeholders, while access shall be allowed for counterstakeholders that are authorized to do so.
  </case>
  <case literal="AvailabilityDegree::all">
    All counterstakeholders shall be allowed to access the personal data personalData of the dataSubject.
  </case>
</switch>
<switch enumeration="repudiation">
  <case literal="Repudiation::plausibleDeniability">
    dataSubject shall be able to plausibly deny that the personal data available to counterstakeholders are related to them.
  </case>
  <case literal="Repudiation::nonRepudiation">
    counterstakeholders shall be able to prove that the personal data available to them are related to the dataSubject.
  </case>
</switch>
```

Listing 7.6: *Text template defining the meaning of an IntegrityRequirement*

```
Random faults of the system and counterstakeholders shall not be able to negatively influence the consistency and correctness of the personal data personalData of the dataSubject.
```

Listing 7.7: *Text template defining the meaning of an AvailabilityRequirement*

```
Random faults of the system and counterstakeholders shall not be able to hinder the corresponding dataSubject to access his or her personal data personalData.
```

7.2.2.2.3. Availability Requirement Template The meaning of an availability requirement is described by the text template shown in Listing 7.7.

7.2.2.3. Validation Conditions for Security-related Requirements

7.2.2.3.1. Confidentiality Requirement Constraints For all subclasses of the abstract class *ConfidentialityRequirement*, I identify the following validation conditions.

In the case that the attribute availability is set to none, the attribute repudiation shall also be none. This is, when no counterstakeholder shall be able to access the respective information, it is unnecessary to specify a kind of repudiation for this information. This introduces the following validation condition.

VSC1 A confidentiality requirement with availability none shall not have a repudiation type different from none.

To avoid the definition of inconsistent confidentiality requirements, I define additional validation conditions concerning the availability degree and repudiation type of confidentiality requirements with the same linkability, data subject, personal data, and counterstakeholder. Note that the linkability only applies for subclasses of *UnlinkabilityRequirement* (see Section 7.2.3).

VSC2 For each instance of a subclass of confidentiality requirement and each combination of (linkability,) data subject, personal data, and counterstakeholder, there shall be at most one instance for each availability degree.

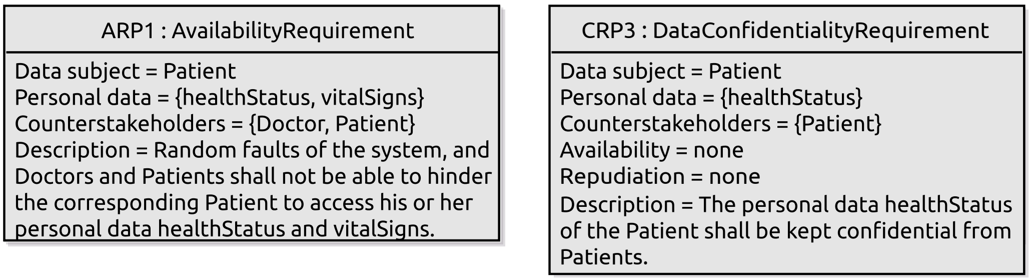
If a confidentiality requirement specifies that no counterstakeholders shall be able to access the respective information, then there should not be another confidentiality requirement of the same type and about the same data subject, personal data, and counterstakeholders.

VSC3 For each confidentiality requirement with availability none, there shall not be another confidentiality requirement of the same type concerning the same personal data, data subject, and counterstakeholder.

Note that this validation condition still allows that there are for the same data subject, personal data, and counterstakeholder three confidentiality requirements of the same type with the availability degrees individual, authorized, and all. This allows to specify different repudiation properties for the different subgroups of the counterstakeholder. To ensure that the repudiation types do not contradict each other in such cases, I introduce the following validation condition.

VSC4 For each confidentiality requirement with availability all and repudiation nonRepudiation, there shall not be another confidentiality requirement of the same type concerning the same personal data, data subject, and counterstakeholder.

7.2.2.3.2. Consistency between Data Confidentiality and Availability Requirements The goals of availability and data confidentiality requirements possibly contradict each other. For example, a confidentiality requirement could state that the data subject him- or herself is not allowed to access his or her personal data by adding the data subject to the list of counterstakeholders and setting the availability to none (see CRP3 in Figure 7.4). An availability requirement stating that the personal data mentioned in the confidentiality requirement shall be accessible to the data subject then contradicts this confidentiality requirement (see ARP1 in Figure 7.4).



ARP1 : AvailabilityRequirement

Data subject = Patient
Personal data = {healthStatus, vitalSigns}
Counterstakeholders = {Doctor, Patient}
Description = Random faults of the system, and Doctors and Patients shall not be able to hinder the corresponding Patient to access his or her personal data healthStatus and vitalSigns.

CRP3 : DataConfidentialityRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {Patient}
Availability = none
Repudiation = none
Description = The personal data healthStatus of the Patient shall be kept confidential from Patients.

Figure 7.4.: *Example of contradicting availability and confidentiality requirements*

To avoid these kinds of contradictions, I introduce the following validation condition.

VSA1 For each availability requirement, there shall be data confidentiality requirements that permit the access for data subjects to the personal data listed by the availability requirement.

Figure 7.4 shows an availability requirement that specifies that patients shall not be hindered to access their health status and vital signs. That requirement does not satisfy the above invariant, because the data confidentiality requirement in the same figure states that no patient shall be allowed to access his or her health status. The definition of a data confidentiality requirement that allows the access to the personal data healthStatus for patients is not allowed due to validation condition VSC3. This is, because the value of the attribute availability of the data confidentiality requirement CRP3 is set to none. Consequently, VSC3 forbids the existence of data confidentiality requirements about the patients' healthStatus with an availability different from none.

7.2.3. Unlinkability-related Privacy Requirements

To derive the privacy requirements related to the protection goal of unlinkability, I use the well-recognized terminology of Pfitzmann and Hansen (2010). Pfitzmann and Hansen define the privacy properties anonymity, unlinkability, undetectability, unobservability, and pseudonymity. Hansen (2012) summarizes all these properties under the protection goal unlinkability.

Pfitzmann and Hansen (2010) define the following privacy properties:

Undetectability *"Undetectability of an item of interest (IOI) from an attacker's perspective means that the attacker cannot sufficiently distinguish whether it exists or not."*

Anonymity *"Anonymity of a subject from an attacker's perspective means that the attacker cannot sufficiently identify the subject within a set of subjects, the anonymity set."*

Unlinkability *"Unlinkability of two or more items of interest (IOIs, e.g., subjects, messages, actions, ...) from an attacker's perspective means that within the system (comprising these and possibly other items), the attacker cannot sufficiently distinguish whether these IOIs are related or not."*

Unobservability *"Unobservability of an item of interest (IOI) means*

- *undetectability of the IOI against all subjects uninvolved in it and*
- *anonymity of the subject(s) involved in the IOI even against the other subject(s) involved in that IOI."*

Pseudonymity *"A pseudonym is an identifier of a subject other than one of the subject's real names." "Pseudonymity is the use of pseudonyms as identifiers."*

7.2.3.1. Metamodel for Unlinkability-related Requirements

Based on this terminology, I created the EMF model shown in Figure 7.5. For the protection goal unlinkability, I derived the class PseudonymityRequirement, the abstract class *UnlinkabilityRequirement*, and the class UndetectabilityRequirement. These are all subclasses of the abstract class *ConfidentialityRequirement*, because they are all concerned with making or not making information available to counterstakeholders. For a pseudonymity requirement, the information is the relation between the pseudonym and the data subject. For an unlinkability requirement, the information is the relation between personal data and the data subject or other personal data. The abstract *UnlinkabilityRequirement* is further refined into the DataUnlinkabilityRequirement (concerning links between personal data) and the AnonymityRequirement (concerning links between personal data and data subjects). Note that the privacy property unobservability is not represented as a separate requirement, but it can be expressed by instantiating respective anonymity and undetectability requirements (cf. the above definition).

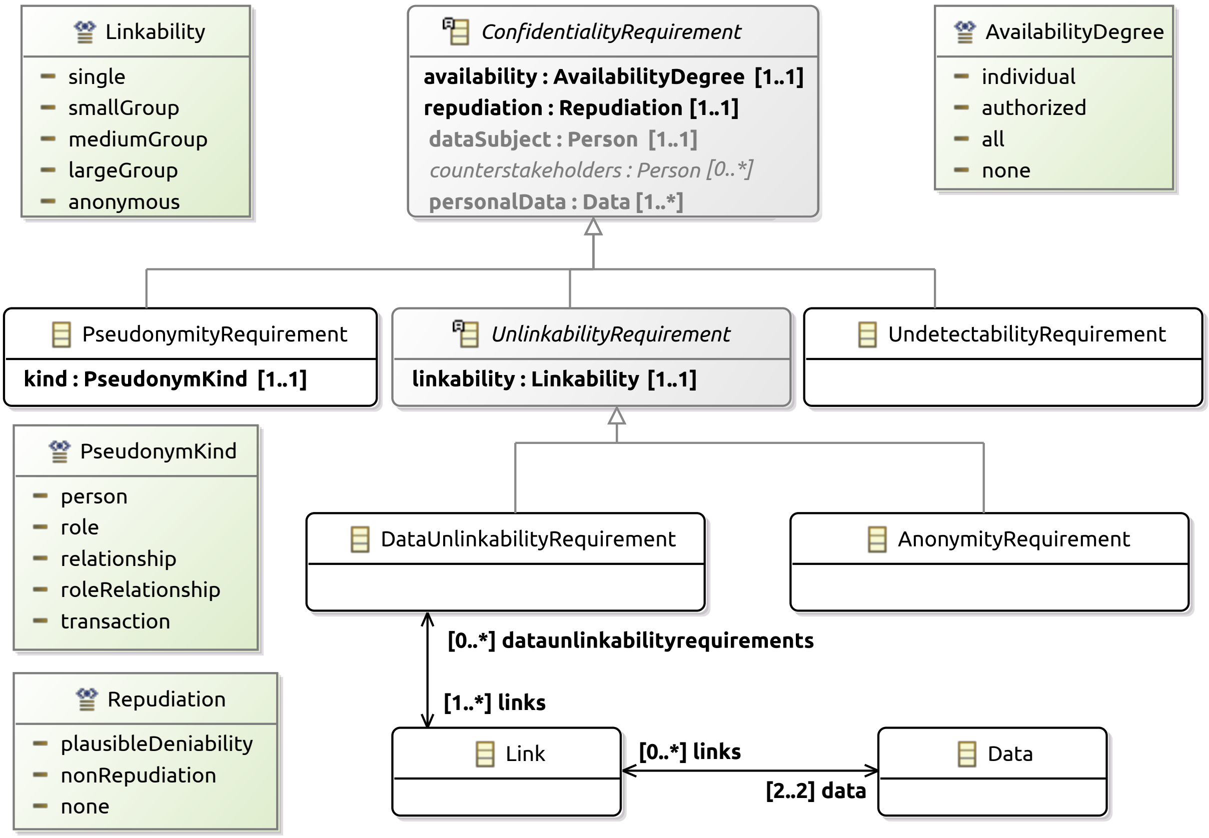

Linkability
single
smallGroup
mediumGroup
largeGroup
anonymous
ConfidentialityRequirement
availability : AvailabilityDegree [1..1]
repudiation : Repudiation [1..1]
dataSubject : Person [1..1]
counterstakeholders : Person [0..*]
personalData : Data [1..*]
AvailabilityDegree
individual
authorized
all
none
PseudonymityRequirement
kind : PseudonymKind [1..1]
UnlinkabilityRequirement
linkability : Linkability [1..1]
UndetectabilityRequirement
PseudonymKind
person
role
relationship
roleRelationship
transaction
DataUnlinkabilityRequirement
AnonymityRequirement
[0..*] dataunlinkabilityrequirements
[1..*] links
Repudiation
plausibleDeniability
nonRepudiation
none
Link
[0..*] links
[2..2] data
Data

Figure 7.5.: *Used Taxonomy of Unlinkability Requirements*

7.2.3.1.1. Undetectability Requirement Class UndetectabilityRequirement is a non-abstract subclass of *ConfidentialityRequirement*. For an undetectability requirement, the information that shall not be available to counterstakeholders is the knowledge about the existence of certain personal data of the data subject. As items of interest (IOI), I consider only personal data in my taxonomy. However, I do not consider this as a limitation, because respective personal data can be introduced that represents the occurrence of a message or action.

7.2.3.1.2. Unlinkability Requirement Class The abstract class *UnlinkabilityRequirement* serves as superclass for the non-abstract classes DataUnlinkabilityRequirement and AnonymityRequirement. These two have in common that the information to be protected is the *linkability* between personal data and the data subject or other personal data. The linkability attribute reflects the allowed degree of linkability with which a counterstakeholder may link a pair of personal data or a piece of personal data and a data subject to each other. Possible values are defined by the enumeration Linkability (cf. Figure 7.5). These are, single, i.e., counterstakeholders are allowed to know which instances of the personal data are related to which other personal data or data subject, smallGroup, mediumGroup, largeGroup, i.e., counterstakeholders are only allowed to know that instances of the personal data are possibly related to a small, medium, or large group of personal data or data subjects, and anonymous, i.e., the counterstakeholder is not allowed to know which instances of personal data are related to each other or to a data subject. Note that an unlinkability requirement with linkability single does not constrain the system-to-be to ensure that counterstakeholders are not able to link personal data to each other or the data subject, but nevertheless, I allow to make this information explicit to document that this linkability is intended.

7.2.3.1.3. Data Unlinkability Requirement Class My DataUnlinkabilityRequirement allows to specify with which degree of linkability counterstakeholders may relate personal data of the data subject to each other. To represent a pair of personal data, I use the class Link that relates two instances of the class Data to each other (see Figure 7.5). A DataUnlinkabilityRequirement instance is associated to a non-empty collection of links. Note that for a data unlinkability requirement the attribute personalData is not relevant. However, it shall be consistent to the associated links (see validation conditions).

7.2.3.1.4. Anonymity Requirement AnonymityRequirement is a non-abstract subclass of the abstract class UnlinkabilityRequirement. An instance of AnonymityRequirement represents that a counterstakeholder is only allowed to link the personal data with the given degree of linkability to the data subject.

7.2.3.1.5. Pseudonymity Requirement Pfitzmann and Hansen (2010) differentiate five kinds of pseudonyms (cf. enumeration PseudonymKind in Figure 7.5). These are:

Person pseudonym *"A person pseudonym is a substitute for the holder's name which is regarded as representation for the holder's civil identity. It may be used in many different contexts, e.g., a number of an identity card, the social security number, DNA, a nickname, the pseudonym of an actor, or a mobile phone number."*

Role pseudonym *"The use of role pseudonyms is limited to specific roles, e.g., a customer pseudonym or an Internet account used for many instantiations of the same role "Internet user". The same role pseudonym may be used with different communication partners. Roles might be assigned by other parties, e.g., a company, but they might be chosen by the subject himself/herself as well."*

Relationship pseudonym *"For each communication partner, a different relationship pseudonym is used. The same relationship pseudonym may be used in different roles for communicating with the same partner. Examples are distinct nicknames for each communication partner"*

Role-relationship pseudonym *"For each role and for each communication partner, a different role-relationship pseudonym is used. This means that the communication partner does not necessarily know, whether two pseudonyms used in different roles belong to the same holder. On the other hand, two different communication partners who interact with a user in the same role, do not know from the pseudonym alone whether it is the same user."*

Transaction pseudonym *"For each transaction, a transaction pseudonym unlinkable to any other transaction pseudonyms and at least initially unlinkable to any other IOI is used, e.g., randomly generated transaction numbers for online-banking. Therefore, transaction pseudonyms can be used to realize as strong anonymity as possible."*

The information that shall not be available to counterstakeholders due to a pseudonymity requirement is the link between the pseudonym and the corresponding data subject, while counterstakeholders shall be able to link the personal data of the data subject to a data subject's pseudonym. The availability attribute allows to further narrow down for which counterstakeholders it shall be possible to link the personal data to a pseudonym of the data subject.

Listing 7.8: *Text template defining the meaning of an* UndetectabilityRequirement

```
<switch enumeration="availability">
  <case literal="AvailabilityDegree::none">
    The counterstakeholders shall not be able to sufficiently distinguish whether the personal data personalData of the dataSubject exist or not.
  </case>
  <case literal="AvailabilityDegree::individual">
    The counterstakeholders shall not be able to sufficiently distinguish whether the personal data personalData of the dataSubject exist or not, while counterstakeholders to whom the personalData are related to shall be allowed to know whether the personal data exist or not.
  </case>
  <case literal="AvailabilityDegree::authorized">
    The counterstakeholders shall not be able to sufficiently distinguish whether the personal data personalData of the dataSubject exist or not, while authorized counterstakeholders shall be allowed to know whether the personal data exist or not.
  </case>
  <case literal="AvailabilityDegree::all">
    All counterstakeholders shall be allowed to know whether the personal data personalData of the dataSubject exist or not.
  </case>
</switch>
<switch enumeration="repudiation">
  <case literal="Repudiation::plausibleDeniability">
  dataSubject shall be able to plausibly deny that the personal data exist or not.
  </case>
  <case literal="Repudiation::nonRepudiation">
    counterstakeholders to whom this knowledge is available shall be able to prove that the personal data exist or not.
  </case>
</switch>
```

The repudiation attribute defines whether counterstakeholders shall be able to prove that the personal data are linkable to the data subject's pseudonym, or whether data subjects shall be able to deny that the personal data are related to one of their pseudonyms.

7.2.3.2. Text Templates for Unlinkability-related Requirements

7.2.3.2.1. Undetectability Requirement Template Based on the above given definition of Pfitzmann and Hansen, an undetectability requirement of my taxonomy has the meaning described by the text template shown in Listing 7.8.

7.2.3.2.2. Data Unlinkability Requirement Template A data unlinkability requirement has the meaning given in Listing 7.9.

7.2.3.2.3. Anonymity Requirement Template The meaning of an anonymity requirement is described by the text template shown in Listing 7.10.

7.2.3.2.4. Pseudonymity Requirement Template A pseudonymity requirement in my taxonomy has the meaning given in Listing 7.11.

Listing 7.9: *Text template defining the meaning of an DataUnlinkabilityRequirement*

```
For each pair of personal data links of the dataSubject,
<switch enumeration="availability">
  <case literal="AvailabilityDegree::individual">
    counterstakeholders to whom the personal data personalData are related to
  </case>
  <case literal="AvailabilityDegree::authorized">
    counterstakeholders that have the right to link these data
  </case>
  <case literal="AvailabilityDegree::all">
    all counterstakeholders
  </case>
</switch>
  shall at most be able to link instances of the two elements of the pair to each other with
      linkability linkability.
<switch enumeration="repudiation">
  <case literal="Repudiation::plausibleDeniability>
    dataSubject shall be able to plausibly deny that the relations among the pairs of personal data
        available to counterstakeholders exist in that way.
  </case>
  <case literal="Repudiation::nonRepudiation>
    counterstakeholders shall be able to prove that the relations among the pairs of personal data
        available to them exist in that way.
  </case>
</switch>
```

7.2.3.3. Validation Conditions for Unlinkability-related Requirements

7.2.3.3.1. Unlinkability Requirement Constraints As mentioned earlier, I forbid to use the value none as availability degree for an unlinkability requirement. This is, because it would mean that no counterstakeholder is allowed to link the personal data with the desired degree to the data subject. Instead of specifying that the degree is not allowed, another degree should be chosen that allows weaker linkage, e.g., anonymous. This is expressed by the following validation condition.

VU1 An unlinkability requirement shall not have the availability degree none.

The repudiation attribute defines whether counterstakeholders shall be able to prove that the links between the personal data and other personal data or the data subject exist with the linkability available to them, or whether data subjects shall be able to deny the existence of these relations. If the linkability is anonymous, the counterstakeholders shall have no knowledge about the relations among the personal data and hence, the repudiation should be set to none.

VU2 An unlinkability requirement that does not allow linkage between personal data shall not specify a repudiation type.

7.2.3.3.2. Data Unlinkability Requirement Constraints The personalData reference of a DataUnlinkabilityRequirement shall be consistent to the data referenced by the contained links. Hence, I introduce the following validation condition.

VUD1 The personal data referenced by a data unlinkability requirement shall be equal to the set of data referenced by the associated links.

Listing 7.10: *Text template defining the meaning of an AnonymityRequirement*

```
<switch enumeration="availability">
  <case literal="AvailabilityDegree::individual>
    counterstakeholders to whom the personalData are related to
  </case>
  <case literal="AvailabilityDegree::authorized>
    counterstakeholders that have the right to link these data
  </case>
  <case literal="AvailabilityDegree::all>
    All counterstakeholders
  </case>
</switch>
shall at most be able to link the personal data personalData to the dataSubject with linkability linkability.
<switch enumeration="repudiation">
  <case literal="Repudiation::plausibleDeniability>
    dataSubject shall be able to plausibly deny that the personal data are linkable to him or her in the described way.
  </case>
  <case literal="Repudiation::nonRepudiation>
    counterstakeholders shall be able to prove that the personal data are linkable to the data subject in the decribed way.
  </case>
</switch>
```

7.2.3.3.3. Anonymity Requirement Constraints Every anonymity requirement has to be consistent to the RelatedTo relation in the metamodel. That means, if a related to relation specifies that some personal data can be linked with a certain linkability to a data subject, then each anonymity requirement about the same personal data and counterstakeholder must not have a weaker linkability.

VUA1 The linkability specified by an anonymity requirement must not be weaker than the weakest linkability specified by a related to relation for the data subject and personal data of the anonymity requirement.

7.2.3.3.4. Pseudonymity Requirement Constraints Similar to unlinkability requirements, I forbid to use the availability degree none for pseudonymity requirements, because it would mean that no counterstakeholder is allowed to link the personal data to the specified kind of pseudonym. Instead of specifying that the degree is not allowed, another pseudonym kind should be chosen that allows weaker linkage, e.g., transaction, or an anonymity requirement with linkability anonymous should be used if no relation to a pseudonym is intended.

VUP1 A pseudonymity requirement shall not have the availability degree none.

7.2.3.3.5. Consistency between Subclasses of Confidentiality Requirement To allow a counterstakeholder to link personal data to the data subject or other personal data, the counterstakeholder needs to know the respective personal data of the data subject. Hence, unlinkability requirements imply data confidentiality requirements that permit the availability of the respective personal data to the counterstakeholders. This leads to the following validation condition.

VU3 For each unlinkability requirement there shall be data confidentiality requirements that specify that the personal data may be available to the counterstakeholders.

Listing 7.11: *Text template defining the meaning of an PseudonymityRequirement*

```
<switch enumeration="availability">
  <case literal="AvailabilityDegree::individual>
    counterstakeholders to whom the personalData are related to
  </case>
  <case literal="AvailabilityDegree::authorized>
    Authorized counterstakeholders
  </case>
  <case literal="AvailabilityDegree::all>
    All counterstakeholders
  </case>
</switch>
shall only be able to relate the personal data personalData to a kind pseudonym and not to the dataSubject him- or herself.
<switch enumeration="repudiation">
  <case literal="Repudiation::plausibleDeniability>
    dataSubject shall be able to plausibly deny that the personal data are related to his or her pseudonym.
  </case>
  <case literal="Repudiation::nonRepudiation>
    counterstakeholders shall be able to prove that the personal data are related to the pseudonym in the decribed way.
  </case>
</switch>
```

Similar to the above validation condition, it has to be checked for each data confidentiality requirement, whether the counterstakeholders to whom the personal data may be available (availability degree different from none) are also allowed to know about the existence of this personal data. The latter needs to be specified by an undetectability requirement with the respective availability. This leads to the following validation condition.

VSD1 For each data confidentiality requirement that specifies that to certain counterstakeholders personal data shall be available, there shall be undetectability requirements that specify that the counterstakeholders are allowed to know about the existence of the personal data.

Pseudonymity requirements may refine anonymity requirements by assigning specific types of pseudonyms to the personal data of data subjects in order to prevent that counterstakeholders identify the individual to whom the personal data are related. To avoid inconsistencies between pseudonymity and anonymity requirements, I propose the following validation condition.

VUP2 For each pseudonymity requirement there shall not be an anonymity requirement for the same data subject and availability degree that (partially) shares personal data and counterstakeholders.

7.2.4. Transparency-related Privacy Requirements

In this section, I present the translation of the taxonomy of transparency requirements presented in Chapter 5 and its additions and modifications introduced by the taxonomy of intervenability requirements presented in Chapter 6 to the EMF metamodel used by ProPAn.

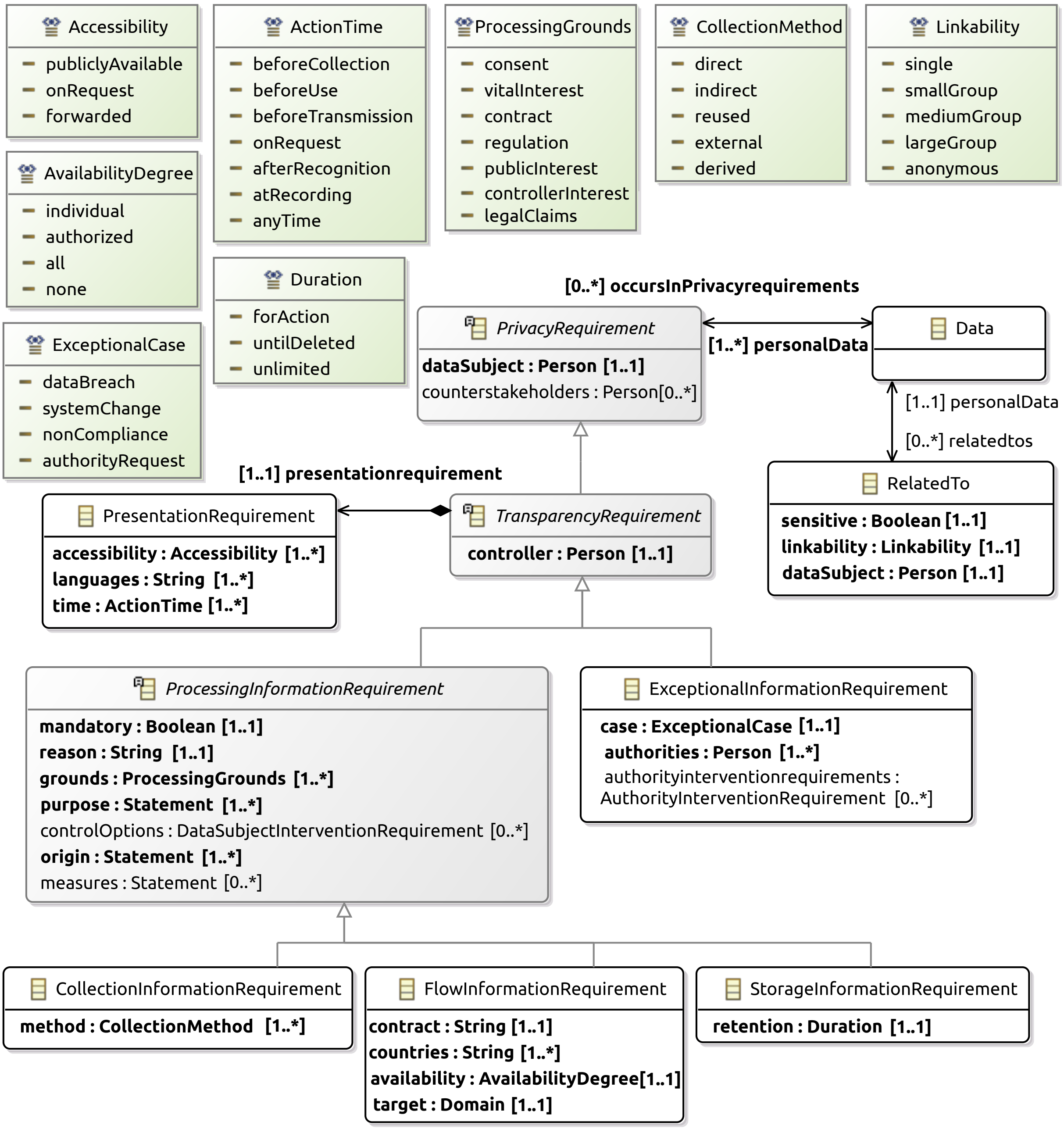

Accessibility
publiclyAvailable
onRequest
forwarded
ActionTime
beforeCollection
beforeUse
beforeTransmission
onRequest
afterRecognition
atRecording
anyTime
ProcessingGrounds
consent
vitalInterest
contract
regulation
publicInterest
controllerInterest
legalClaims
CollectionMethod
direct
indirect
reused
external
derived
Linkability
single
smallGroup
mediumGroup
largeGroup
anonymous
AvailabilityDegree
individual
authorized
all
none
Duration
forAction
untilDeleted
unlimited
ExceptionalCase
dataBreach
systemChange
nonCompliance
authorityRequest
[0..*] occursInPrivacyrequirements
PrivacyRequirement
dataSubject : Person [1..1]
counterstakeholders : Person[0..*]
[1..*] personalData
Data
[1..1] personalData
[0..*] relatedtos
RelatedTo
sensitive : Boolean [1..1]
linkability : Linkability [1..1]
dataSubject : Person [1..1]
[1..1] presentationrequirement
PresentationRequirement
accessibility : Accessibility [1..*]
languages : String [1..*]
time : ActionTime [1..*]
TransparencyRequirement
controller : Person [1..1]
ProcessingInformationRequirement
mandatory : Boolean [1..1]
reason : String [1..1]
grounds : ProcessingGrounds [1..*]
purpose : Statement [1..*]
controlOptions : DataSubjectInterventionRequirement [0..*]
origin : Statement [1..*]
measures : Statement [0..*]
ExceptionalInformationRequirement
case : ExceptionalCase [1..1]
authorities : Person [1..*]
authorityinterventionrequirements :
AuthorityInterventionRequirement [0..*]
CollectionInformationRequirement
method : CollectionMethod [1..*]
FlowInformationRequirement
contract : String [1..1]
countries : String [1..*]
availability : AvailabilityDegree[1..1]
target : Domain [1..1]
StorageInformationRequirement
retention : Duration [1..1]

Figure 7.6.: *Used Taxonomy of Transparency Requirements*

7.2.4.1. Metamodel for Transparency-related Requirements

In the following, I only discuss the differences to the class diagram shown in Figure 5.2 on page 78.

7.2.4.1.1. Changes in Enumerations When we compare Figure 5.2 on page 78 and the additions to it in Figure 6.2 on page 91 with the EMF model for transparency-related privacy requirements shown in Figure 7.6, we observe that the enumerations PresentationTime, AccessibilityType, and RetentionType are renamed to ActionTime, Accessibility, and Duration, respectively. I did this due to the use of these enumerations in other parts of ProPAn's EMF metamodel that are introduced later in this thesis.

Listing 7.12: *Text template defining the meaning of an PresentationRequirement*

```
The information contained in the related transparency requirements has to be presented time, in languages, and made accessible to data subjects by
<switch enumeration="accessibility">
  <case literal="Accessibility::publiclyAvailable">
     making the information publicly available and accessible.
  </case>
  <case literal="Accessibility::onRequest">
     presenting the information when it is requested from the data subject.
  </case>
  <case literal="Accessibility::forwarded">
     forwarding this information to the data subject when needed.
  </case>
</switch>
```

7.2.4.1.2. Transparency Requirement Class The attributes dataSubject, counterstakeholders, and personalData of the abstract class *TransparencyRequirement* in Figure 5.2 on page 78 are inherited from the abstract class *PrivacyRequirement* in Figure 7.6. Additionally, the attributes sensitiveData and dataLinkability are moved to the class RelatedTo. The purpose of this class is to define for a pair of a data subject and a personal data, whether these personal data are sensitive for the data subject, and to which degree the personal data are linkable to the data subject. In this way, transparency requirements can also group a set of personal data with elements that do not have homogeneous sensitivity and linkability.

7.2.4.1.3. Processing Information Requirement Class In ProPAn's privacy taxonomy, the *ProcessingInformationRequirement* is considered as abstract, because I do not allow instances of this class. I also changed the type and name of the attribute security to Statement and measures, respectively. This allows to directly reference the functional requirement, or domain knowledge from the problem frame model (cf. Section 2.3) that represents a protection measure that was selected to protect the data subject's personal data. I further added the attribute origin to assign the functional requirements or domain knowledge that represent and describe the processing about which the processing information requirement shall inform.

7.2.4.1.4. Flow Information Requirement Class Finally, I added to the FlowInformationRequirement the attributes availability and target. The attribute target specifies to which Domain of the problem frame model the personal data flows and the attribute availability allows to further narrow down which instances of the domain may receive the personal data (cf. attribute availability of *ConfidentialityRequirement*).

7.2.4.2. Text Templates for Transparency-related Requirements

7.2.4.2.1. Presentation Requirement Template The text template describing a PresentationRequirement is shown in Listing 7.12.

7.2.4.2.2. ExceptionalInformationRequirement The meaning of an ExceptionalInformationRequirement is given by the template in Listing 7.13.

7.2.4.2.3. CollectionInformationRequirement A CollectionInformationRequirement has the meaning defined by the text template shown in Listing 7.14.

Listing 7.13: *Text template defining the meaning of an ExceptionalInformationRequirement*

```
The dataSubject and authorities shall be informed in the case of a case regarding or affecting the personal data personalData of the dataSubject. authorities may excercise in this situation their rights described by authorityinterventionrequirements.
```

Listing 7.14: *Text template defining the meaning of an CollectionInformationRequirement*

```
The dataSubject shall be informed that his or her personal data personalData are
<if condition="mandatory">
    mandatorily
</if>
collected by the system-to-be that is run by the controller.
The applied collection methods to obtain the personal data from the data subject are method.
The data subject's possibilities to control the collection of his or her data are controlOptions.
The personal data are collected during origin for the purpose of purpose because reason.
The processing grounds on grounds.
The controller has selected the protection mechanisms measures to protect the personal data.
The details on how the information has to be presented to the data subject are defined in presentationrequirement.
```

Listing 7.15: *Text template defining the meaning of an FlowInformationRequirement*

```
The dataSubject shall be informed that his or her personal data personalData
<if condition="mandatory">
    mandatorily
</if>
flow to the target due to the system-to-be that is run by the controller.
The target is located in countries and contractual obligations contract exist between the target and the controller.
The data subject's possibilities to control the flow of his or her data are controlOptions.
The personal data flow during origin for the purpose of purpose because reason.
The processing grounds on grounds.
The controller has selected the protection mechanisms measures to protect the personal data.
The details on how the information has to be presented to the data subject are defined in presentationrequirement.
```

7.2.4.2.4. FlowInformationRequirement Listing 7.15 gives the meaning of a FlowInformationRequirement.

7.2.4.2.5. StorageInformationRequirement The text template describing a StorageInformationRequirement is shown in Listing 7.16.

7.2.4.3. Validation Conditions for Transparency-related Requirements

7.2.4.3.1. Collection Information Requirement To check consistency among collection information requirements, I introduce the following validation condition.

VTC1 There shall be at most one collection information requirement for each combination of data subject, personal data, and purpose.

Otherwise, inconsistent or redundant notification needs may be stated by different collection information requirements for the collection of the same personal data for the same purpose.

Listing 7.16: *Text template defining the meaning of an StorageInformationRequirement*

```
The dataSubject shall be informed that his or her personal data personalData are
<if condition="mandatory">
    mandatorily
</if>
stored by the system-to-be that is run by the controller.
The duration for which the personal data are retained in the system-to-be is retention.
The data subject's possibilities to control the storage of his or her data are controlOptions.
The personal data are stored during origin for the purpose of purpose because reason.
The processing grounds on grounds.
The controller has selected the protection mechanisms measures to protect the personal data.
The details on how the information has to be presented to the data subject are defined in
    presentationrequirement.
```

7.2.4.3.2. Flow Information Requirement To check consistency among flow information requirements, I introduce the following validation condition.

VTF1 There shall be at most one flow information requirement for each combination of data subject, personal data, purpose, and target.

7.2.4.3.3. Storage Information Requirement To check consistency among storage information requirements, I introduce the following validation condition.

VTS1 There shall be at most one storage information requirement for each combination of data subject, personal data, and purpose.

7.2.4.3.4. Exceptional Information Requirement To check consistency among exceptional information requirements, I introduce the following validation condition.

VTE1 There shall be at most one exceptional information requirement for each combination of data subject, personal data, case, and authority.

7.2.5. Intervenability-related Privacy Requirements

In this section, I present the translation of the taxonomy of intervenability requirements presented in Chapter 6 to the EMF metamodel used by ProPAn.

7.2.5.1. Metamodel for Intervenability-related Requirements

The part of the requirements taxonomy concerning intervenability requirements is shown in Figure 7.7. It corresponds to the metamodel shown in Figure 6.2 on page 91 and is hence not futher explained here.

7.2.5.2. Text Templates for Intervenability-related Requirements

7.2.5.2.1. Data Subject Intervention Requirement Template A data subject intervention requirement has the meaning specified by the text template in Listing 7.17.

7.2.5.2.2. Authority Intervention Requirement Template The text template shown in Listing 7.18 defines the meaning of an AuthorityInterventionRequirement.

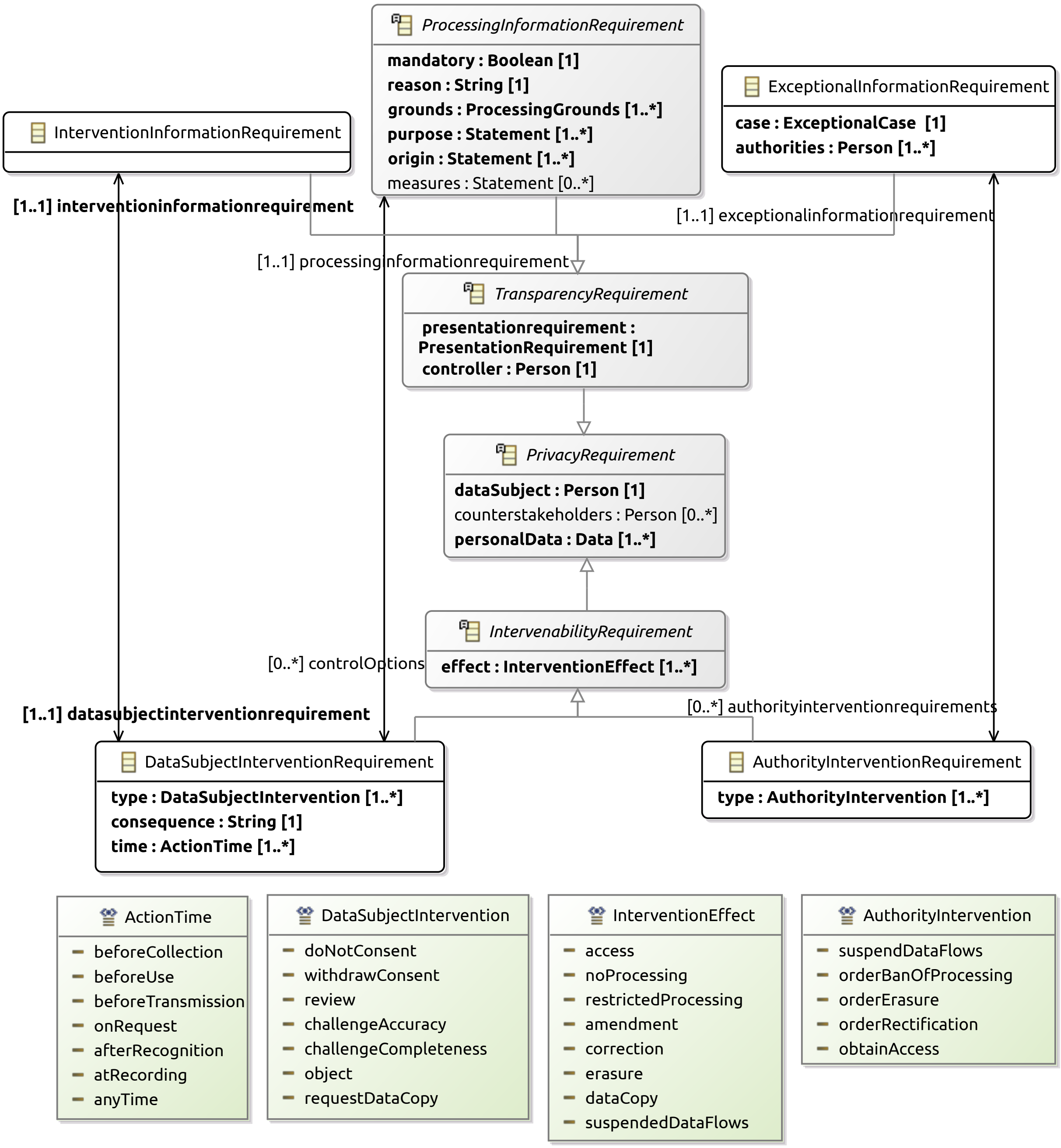

ProcessingInformationRequirement
mandatory : Boolean [1]
reason : String [1]
grounds : ProcessingGrounds [1..*]
purpose : Statement [1..*]
origin : Statement [1..*]
measures : Statement [0..*]
InterventionInformationRequirement
ExceptionalInformationRequirement
case : ExceptionalCase [1]
authorities : Person [1..*]
[1..1] interventioninformationrequirement
[1..1] exceptionalinformationrequirement
[1..1] processinginformationrequirement
TransparencyRequirement
presentationrequirement : PresentationRequirement [1]
controller : Person [1]
PrivacyRequirement
dataSubject : Person [1]
counterstakeholders : Person [0..*]
personalData : Data [1..*]
IntervenabilityRequirement
effect : InterventionEffect [1..*]
[0..*] controlOptions
[1..1] datasubjectinterventionrequirement
[0..*] authorityinterventionrequirements
DataSubjectInterventionRequirement
type : DataSubjectIntervention [1..*]
consequence : String [1]
time : ActionTime [1..*]
AuthorityInterventionRequirement
type : AuthorityIntervention [1..*]
ActionTime
beforeCollection
beforeUse
beforeTransmission
onRequest
afterRecognition
atRecording
anyTime
DataSubjectIntervention
doNotConsent
withdrawConsent
review
challengeAccuracy
challengeCompleteness
object
requestDataCopy
InterventionEffect
access
noProcessing
restrictedProcessing
amendment
correction
erasure
dataCopy
suspendedDataFlows
AuthorityIntervention
suspendDataFlows
orderBanOfProcessing
orderErasure
orderRectification
obtainAccess

Figure 7.7.: *Used Taxonomy of Intervenability Requirements*

7.2.5.3. Validation Conditions for Intervenability-related Requirements

7.2.5.3.1. Data Subject Intervention Requirement Constraints I specify that the combinations of intervention types and intervention effects are limited to the combinations shown in Table 6.4 on page 92 in the following validation condition.

VID1 For each data subject intervention requirement, the combinations of intervention types and intervention effects have to comply to Table 6.4 on page 92.

Data subject intervention requirements with intervention type review, challengeAccuracy, and challengeCompleteness have to be control options of flow or storage information requirements.

Listing 7.17: *Text template defining the meaning of an DataSubjectInterventionRequirement*

```
time, the dataSubject shall be able to type
<if condition="type→includes(DataSubjectIntervention::doNotConsent) or
    type→includes(DataSubjectIntervention::withdrawConsent) or
    type→includes(DataSubjectIntervention::object)">
  to the processing described in processinginformationrequirement. The intervention shall result in
      effect.
</if>
<if condition="type→includes(DataSubjectIntervention::challengeAccuracy) or
    type→includes(DataSubjectIntervention::challengeCompleteness)">
  of the personal data whose processing is described in processinginformationrequirement. The
      intervention shall result in a(n) effect of the personal data.
</if>
<if condition="type→includes(DataSubjectIntervention::review)">
  the personal data whose processing is described in processinginformationrequirement. The
      intervention shall result in access to the personal data.
</if>
<if condition="type→includes(DataSubjectIntervention::requestDataCopy)">
  of the personal data whose processing is described in processinginformationrequirement. The
      intervention shall result in providing a data copy of the personal data to the data subject.
</if>
This may result in the consequences consequence for the data subject.
```

Listing 7.18: *Text template defining the meaning of an AuthorityInterventionRequirement*

```
exceptionalinformationrequirement.authorities shall be able to type in the cases decribed in
    exceptionalinformationrequirement. The intervention shall result in effect.
```

VID2 Each data subject intervention requirement with intervention type review, challengeAccuracy, and challengeCompleteness has to be a control option of a flow or storage information requirement.

7.2.5.3.2. Authority Intervention Requirement Constraints I specify that the combinations of intervention types and intervention effects are limited to the combinations shown in Table 6.3 on page 92 in the following validation condition.

VIA1 For each authority intervention requirement, the combinations of intervention types and intervention effects have to comply to Table 6.3 on page 92.

7.2.5.3.3. Consistency between Processing Information and Data Subject Intervention Requirements If the processing of personal data is based on the data subject's consent, then the data subject shall have the right to do not give that consent or to withdraw previously given consent. This constraint is expressed by the following validation condition.

VTP1 For each processing information requirement whose grounds include consent, there shall be data subject intervention requirements associated to the processing information requirement with the intervention types doNotConsent and withDrawConsent.

7.3. Relation to Other Privacy Frameworks

In this section, I compare my privacy requirements taxonomy to the other privacy conceptual models identified during the literature review presented in Chapter 3.

7.3.1. Privacy Principles, Regulations, and Policies

Privacy principles and policies both emerge from privacy regulations. Either as more practical guidelines on how a software should be engineered to comply to privacy regulations, or to make explicit what personal data are processed by the system-to-be and how. A privacy regulation formulates privacy protection obligations that can be addressed by following privacy principles, which may go beyond the privacy protection needs formulated in a privacy regulation. A privacy policy formulates and documents the selected procedures and measures for a system-to-be with regard to the protection of the end-users' privacy.

Privacy principles form a collection of privacy practices and guidelines that are recommended in regulations and applied in privacy policies. Hence, they are representative for this class of privacy conceptual models based on privacy principles, regulations, and policies. I mapped the privacy principles that I identified in Section 2.4.3 on page 26 to the privacy requirements of my taxonomy that can be used to realize a principle and requirements that may be used to comply to the principle. Table 7.1 shows this mapping. An "X" in Table 7.1 (and anologously in Tables 7.2, 7.3, and 7.4) means that the respective privacy requirement will be instantiated to address an issue stated in the respective privacy principle, while an "O" means that instances of the respective privacy requirement may be used to address a privacy principle.

The privacy principle *Access* states that data subjects shall be able to access (availability requirement) and review (data subject intervention requirement) his or her personal data. These access and review rights shall only be available to the respective data subject which optionally leads to data confidentiality requirements. The *Individual participation* principle is covered by the different options to influence the processing of personal data described by data subject intervention requirements. To adhere to the *Accuracy and quality* principle, integrity requirements for the processed personal data need to be instantiated and possibly data subject intervention requirements that allow data subjects to challenge the accuracy and completeness of their personal data. The principles *Openness and transparency*, *Consent and choice*, *Purpose legitimacy*, and *Purpose specification* are all addressed by instantiating respective collection, storage, and flow information requirements. For the principle *Consent and choice*, data subject intervention requirements may be instantiated to reflect the intervention options to not consent or to withdraw previously given consent. The confidentiality-related requirements data confidentiality, undetectability, anonymity, pseudonymity, and data unlinkability could be instantiated to contribute to the privacy principles *Use limitation*, *Storage limitation*, *Collection limitation*, and *Data minimization*. This is, because these confidentiality requirements describe different possibilities to limit the use, storage, and collection of personal data, e.g., by removing the link between the personal data and the data subject, or by restricting the access to the personal data. Additionally, the *Collection limitation* principle requires that the data subject is informed about the collection and is able to intervene in this collection, which results in collection information and data subject intervention requirements. The *Security* principle requires appropriate technical measures to protect the processed personal data and consequently requires the security-related privacy requirements data confidentiality, integrity, and availability, and possibly the unlinkability-related requirements undetectability, anonymity, pseudonymity, and data unlinkability. To respect the *Accountability* and *Compliance* principles, it is necessary to document what and how personal data are processed and how data subjects and authorities are able to intervene in the processing in compliance with the relevant regulations. Hence, instances

Table 7.1.: *Mapping of Privacy Principles to my Privacy Requirements Taxonomy*

Privacy Principles	**Privacy Requirements**												
	SD	**SI**	**SA**	**UU**	**UA**	**UP**	**UD**	**TC**	**TS**	**TF**	**TE**	**ID**	**IA**
Access	O		X									X	
Individual participation												X	
Accuracy and quality		X										O	
Openness and transparency								X	X	X			
Consent and choice								X	X	X		X	
Purpose legitimacy								X	X	X			
Purpose specification								X	X	X			
Use limitation	O			O	O	O	O						
Storage limitation	O			O	O	O	O						
Collection limitation	O			O	O	O	O	X				X	
Data minimization	O			O	O	O	O						
Security	X	X	X	O	O	O	O						
Accountability								X	X	X	X	X	X
Compliance								X	X	X	X	X	X
Fairness	O	O	O	O	O	O	O	O	O	O	O	O	O

SD: Data confidentiality **SI**: Integrity **SA**: Availability **UU**: Undetectability
UA: Anonymity **UP**: Pseudonymity **UD**: Data unlinkability
TC: Collection information **TS**: Storage information **TF**: Flow information
TE: Exceptional information **ID**: Data subject intervention **IA**: Authority intervention

of the transparency-related requirements collection, storage, flow, and exceptional information requirement, and the intervenability-related requirements data subject, and authority intervention requirement may be needed. Implementing the *Fairness* principle means to implement appropriate security, unlinkability, transparency, and intervenability measures to ensure a fair processing of personal data and hence, may lead to an instantiation of all privacy requirements of my taxonomy.

Table 7.1 provides guidance on how privacy principles can be translated to the software-related privacy requirements of my taxonomy. However, the privacy principles, especially the principles *Accountability* and *Compliance*, not only cover requirements on the system-to-be, but also responsibilities and processes in the organization. Note that the latter is out of the scope of my taxonomy. It is also visible from Table 7.1 that the privacy principles do not explicitly require unlinkability-related requirements, but these are implicitly needed to realize the limitation-related privacy principles.

7.3.2. Data Privacy Taxonomy

Barker et al. (2009) propose in their data privacy taxonomy four dimensions of privacy to assess the privacy properties of a system. These four dimensions are *visibility*, *granularity*, *purpose*, and *retention*. With the *visibility* dimension it is analyzed to whom personal data might be disclosed. This dimension is covered by all confidentiality requirement subclasses that specify to whom which information (e.g., the personal data or links between data subjects and their personal data) shall be available, and can also be supported by flow information requirements that provide additional information on intended flows of personal data and availability requirements documenting the access of data subjects to their personal data (see Table 7.2). The *granularity* dimension aims to reflect in which detail personal data are processed, e.g., with which linkability to the data subject or to other data. This dimension is also covered by all subclasses of the confidentiality requirement in my taxonomy. Additionally, collection, storage, and flow information requirements may be instantiated to document under which circumstances personal data are processed. The dimension *purpose* is covered by the collection, storage, and flow information requirements of my taxonomy, because these also document for which purposes personal data are collected, stored, or provided to others. To document how long personal data are retained, storage information requirements may be instantiated. These contribute to the *retention* dimension of the data privacy taxonomy. Additionally, the *retention* dimension possibly requires that integrity requirements are instantiated for the personal data stored by the system-to-be to ensure that the personal data are kept up-to-date and accurate.

From Table 7.2, we can see that the data privacy taxonomy does not cover the privacy goal intervenability and exceptional information requirements. My taxonomy covers all dimensions proposed by Barker et al. and refines these to more concrete software-related privacy requirements.

Table 7.2.: *Mapping of Privacy Dimensions to my Privacy Requirements Taxonomy*

Privacy Dimension	**Privacy Requirements**												
	SD	**SI**	**SA**	**UU**	**UA**	**UP**	**UD**	**TC**	**TS**	**TF**	**TE**	**ID**	**IA**
Visibility	X		O	X	X	X	X			O			
Granularity	X			X	X	X	X	O	O	O			
Purpose								X	X	X			
Retention		O							X				

SD: Data confidentiality **SI**: Integrity **SA**: Availability **UU**: Undetectability
UA: Anonymity **UP**: Pseudonymity **UD**: Data unlinkability
TC: Collection information **TS**: Storage information **TF**: Flow information
TE: Exceptional information **ID**: Data subject intervention **IA**: Authority intervention

7.3.3. Contextual Integrity

The concept of contextual integrity proposed by Nissenbaum (2004) focuses on flow of personal data between agents. Hence, my confidentiality-related requirements data confidentiality, undetectability, anonymity, pseudonymity, data unlinkability, and flow information requirement may be used to reflect contextual integrity. Contextual integrity consists of four main constructs. The construct *roles* emphasizes that agents may act in different roles with different privileges. My taxonomy also considers recipients (in most cases called counterstakeholders or target) of personal data or other information as roles (see Table 7.3). With the constructs *informational norms*, Nissenbaum states that whether flows of personal data are acceptable or not depends on the norms that are valid in the context of that information exchange. Additionally, the construct *appropriateness* aims at evaluating whether it is fair and necessary to collect or send certain personal data to others for the processing purpose. My taxonomy allows to document the intended flows and collection of personal data and to evaluate based on this documentation whether the intended flows and collection comply to informational norms and are appropriate. The construct *principles of transmission* emphasizes that there are distinct transmission principles, such as confidentiality (a recipient may not provide the received personal data to others), reciprocity (in some cases the exchange of personal data shall be bi-directional, e.g., between friends), dessert (in some cases agents deserve to get specific personal data, e.g., that a co-worker has a contagious illness), and awareness and consent (that data subjects are aware of how their personal data flows and that they give consent to these flows).

Nissenbaum's contextual integrity may be used as a high-level framework to elicit or evaluate privacy requirements of my taxonomy, similar to the data privacy taxonomy discussed in the previous subsection. However, it does not cover integrity, availability, storage information, exceptional information, and authority intervention requirements (see Table 7.3).

Table 7.3.: *Mapping of Privacy Constructs to my Privacy Requirements Taxonomy*

Privacy	**Privacy Requirements**												
Construct	**SD**	**SI**	**SA**	**UU**	**UA**	**UP**	**UD**	**TC**	**TS**	**TF**	**TE**	**ID**	**IA**
Roles	O			O	O	O	O			O			
Informational norms	O			O	O	O	O	O		O			
Appropriateness	O			O	O	O	O	O		O			
Principles of transmission	O			O	O	O	O			O		O	

SD: Data confidentiality **SI**: Integrity **SA**: Availability **UU**: Undetectability
UA: Anonymity **UP**: Pseudonymity **UD**: Data unlinkability
TC: Collection information **TS**: Storage information **TF**: Flow information
TE: Exceptional information **ID**: Data subject intervention **IA**: Authority intervention

7.3.4. Privacy threats and goals

The engineering methods that are based on privacy threats as conceptual model, either are based on privacy goals that are negated as done, e.g., by Deng et al. (2011), or the authors do not provide further guidance on how to elicit the privacy threats. Hence, I focus for the comparison to my privacy requirements taxonomy on the privacy goals proposed in the literature.

From the literature review, we can mainly distinguish four (overlapping) sets of privacy goals. These are proposed by Kalloniatis et al. (2008), Deng et al. (2011), Jutla et al. (2013), and Feth et al. (2017). Table 7.4 shows the different privacy goals from these sources and how they are reflected in my taxonomy.

The privacy goals *confidentiality*, *integrity*, *availability*, *undetectability*, *anonymity*, *pseudonymity*, and *unlinkability* are represented as respective requirements in my taxonomy. The privacy goal *security* proposed by Jutla et al. can be mapped to the security-related privacy requirements of my taxonomy. Kalloniatis et al. propose *identification* as privacy goal. This privacy goal can be represented by respective anonymity, and data unlinkability requirements with the linkability single (cf. Section 7.2.3). The security goals *authentication* and *authorization* are considered by Kalloniatis et al. and Feth et al. in the context of privacy. I do not directly consider them in my taxonomy, because I consider these two as more technical privacy requirements that refine security- and unlinkability-related privacy requirements. I propose to first document the

Table 7.4.: *Mapping of Privacy Constructs to my Privacy Requirements Taxonomy*

	Privacy Requirements												
Privacy Goal	**SD**	**SI**	**SA**	**UU**	**UA**	**UP**	**UD**	**TC**	**TS**	**TF**	**TE**	**ID**	**IA**
Confidentiality (Deng et al.; Feth et al.)	X												
Integrity (Feth et al.)		X											
Availability (Jutla et al.; Feth et al.)			X										
Undetectability (Kalloniatis et al.; Deng et al.)				X									
Anonymity (Kalloniatis et al.; Deng et al.; Jutla et al.; Feth et al.)					X								
Pseudonymity (Kalloniatis et al.; Deng et al.; Jutla et al.; Feth et al.)						X							
Unlinkability (Kalloniatis et al.; Feth et al.)							X						
Security (Jutla et al.)	X	X	X										
Identification (Kalloniatis et al.)					X		X						
Authentication (Kalloniatis et al.; Feth et al.)	O	O	O	O	O	O	O						
Authorization (Kalloniatis et al.; Feth et al.)	O	O	O	O	O	O	O						
Plausible deniability (Deng et al.)	X			X	X	X	X						
Non-repudiation (Feth et al.)	X			X	X	X	X						
Notice (Jutla et al.)								X	X	X			
Content awareness (Deng et al.)								X	X	X			
Transparency (Feth et al.)								X	X	X	X		
Agreement and consent (Jutla et al.)												X	
Policy and consent compliance (Deng et al.)												X	O
Data protection (Kalloniatis et al.)	O	O	O	O	O	O	O	O	O	O	O	O	O

SD: Data confidentiality **SI**: Integrity **SA**: Availability **UU**: Undetectability
UA: Anonymity **UP**: Pseudonymity **UD**: Data unlinkability
TC: Collection information **TS**: Storage information **TF**: Flow information
TE: Exceptional information **ID**: Data subject intervention **IA**: Authority intervention

related security- and unlinkability-related requirements and to refine these later to mechanisms that may be based on the concepts of authentication and authorization. I integrated the goals *plausible deniability* and *non-repudiation* into all confidentiality-related privacy requirements to directly document which degree of repudiation shall be available to the counterstakeholders to whom specific information shall be available (see Section 7.2.2.1.1).

The privacy goals *notice*, *content awareness*, and *transparency*, are all reflected by my taxonomy of transparency requirements. Similarly, the goals *agreement and consent*, and *policy and consent compliance* are covered by my taxonomy of intervenability requirements that is derived from ISO 29100 (ISO/IEC, 2011) and the EU General Data Protection Regulation (European Commission, 2016) and allows to define requirements to comply to these two. The privacy goal *data protection* proposed by Kalloniatis et al. is not well-specified by them. However, I expect that all aspects of this privacy goal can be covered by my taxonomy.

We can see from Table 7.4 that most privacy goals considered in the literature are reflected by my taxonomy. Only the goals *authentication*, *authorization*, and *data protection* have no requirements that directly represent them. However, they can be represented as more abstract requirements that represent what shall be achieved with these goals (in the case of *authentication* and *authorization*), or with more concrete privacy requirements refining them (in the case of *data protection*).

7.3.5. Privacy as Individual Concept, Concerns, and Social Expectations and Norms

The privacy conceptual models that I call privacy as individual concept (Bellotti and Sellen, 1993; Hong et al., 2004), concerns (Gürses et al., 2011; Spiekermann and Cranor, 2009), and social expectations and norms (Murukannaiah et al., 2016) are all high-level and open, without providing detailed concepts or constructs. These models do not give much guidance on how privacy needs can be identified and are in most cases focused on the personal perception of privacy and social norms. Barker et al.'s data privacy taxonomy and Nissenbaum's conceptual integrity are a step forward to formalize and structure these high-level and open privacy models. As I am able to map the key concepts of the latter two to my taxonomy of privacy requirements, I also expect that my taxonomy can be used to refine the privacy needs identified following the conceptual models considering privacy as individual concept, concerns, and social expectations and norms to software-related privacy requirements.

7.4. Conclusions

In this chapter, I have presented my complete taxonomy of privacy requirements that are used by the ProPAn method introduced in the following part of this thesis. In total, this chapter has the following four contributions.

First, I introduce all privacy requirements in a hierarchical EMF metamodel. The contained privacy requirements are all refinements of the six protection goals proposed by Hansen et al. (2015). The metamodel details the privacy requirements with attributes that allow to characterize them and to tailor their instances to a specific system-to-be to which they apply. Additionally, the metamodel models relations between privacy requirements, e.g., between transparency and intervenability requirements. The EMF metamodel allows to document privacy requirements in a structured and machine-readable way and also builds the foundation of the tool support for the ProPAn method.

Second, I provide text templates for all instantiatable privacy requirements of my taxonomy. These text templates allow to generate human-readable representations of the privacy requirements and also define their semantics using natural language. These text templates are especially valuable to create text documents from the EMF metamodel that are dedicated to an audience that is not familiar with model-driven engineering.

Third, I provide validation conditions both in natural language and formalized as OCL constraints on the EMF metamodel (see Appendix A). The OCL constraints can automatically be evaluated on instances of my privacy requirements. The validation conditions help to check the soundness of privacy requirement instances themselves and the consistency between different privacy requirement instances.

Fourth, I mapped my taxonomy to the privacy conceptual models that I identified during my literature review on privacy requirements engineering methods (see Chapter 3). The comparison to the other conceptual models has shown that my taxonomy is capable to document *what* privacy requirements the system-to-be shall satisfy, without being focused on *how* these shall be realized. The privacy conceptual models *privacy principles*, *privacy regulations*, *data privacy taxonomy*, *contextual integrity*, *privacy as personal notation*, *privacy concerns*, and *social expectations and norms* help to determine *why* specific privacy requirements are needed. Hence, they can be used to elicit the privacy requirements that then can be modeled as instances of my privacy requirements taxonomy.

The proposed privacy requirements taxonomy is used in several steps of the ProPAn method. In Chapter 12, I introduce a systematic method to identify the needed privacy requirements of my taxonomy for a given system-to-be based on its functional requirements in a computer-aided way. A method to elicit threats to the privacy requirements of my taxonomy is introduced in Chapter 13. I show a pattern-based and aspect-oriented way to present privacy enhancing technologies (PETs), including a description how these PETs positively and negatively influence the privacy requirements of my taxonomy in Chapter 16. Finally, I discuss how the privacy requirements of my taxonomy can be operationalized to functional requirements based on the pattern-based and aspect-oriented presentation of PETs in Chapter 17.

Part III.

Problem-based Privacy Analysis

Chapter 8

Overview of the ProPAn Method

In this chapter, I provide an overview of the Problem-based Privacy Analysis (ProPAn) method. I introduce the seven main steps, their inputs and outputs, and the control flow of the ProPAn method as shown in Figure 8.1 as a UML2 activity diagram. The activity diagram shows on the left hand side the seven steps and the control flow of the method. On the right hand side, the different models are shown that are used to store the needed inputs and outputs of the ProPAn method. The fork symbols inside the steps indicate that these steps contain substeps. I provide a detailed presentation of each of the seven steps in a separate chapter of my thesis. Note that all steps of the ProPAn method shall be performed by an analysis team that comprises expertise in requirements engineering, privacy, and the application domain (cf. Section 3.2.2).

In Sections 8.1-8.7, I briefly describe the *purpose*, *inputs*, *outputs*, and the following *control flow*, i.e., the evaluation of the decision nodes following the respective activity, for each of the seven steps. I describe the relations between the models used during the ProPAn method in Section 8.8. Finally, I relate the steps of the ProPAn method to the tasks of the high-level privacy requirements engineering methodology (see Section 3.2.2) in Section 8.9.

8.1. Privacy Context Elicitation

The goal of the first step of the ProPAn model is to enhance a given Problem Frame Model. As inputs for the step Privacy Context Elicitation, the Context diagram, Problem diagrams representing the functional requirements of the system-to-be, and optionally Domain Knowledge Diagrams representing facts and assumptions about the machine's environment are needed (see also Figure 8.1). These are expected to be provided in a Problem Frame Model that is an instance of the UDEPF metamodel introduced in Section 2.3. The Problem Frame Model is extended during this step with additional privacy-relevant domain knowledge. This domain knowledge includes data flows outside the scope of the machine (e.g., communication among biddable domains), availability of personal data at domains that is independent of the machine, and the documentation of connection domains. The domain knowledge is documented as Domain knowledge diagrams in the given Problem Frame Model. A detailed description of this step and supporting material (e.g., questionnaires) can be found in Chapter 9. After the step Privacy Context Elicitation, the step Privacy Threshold Analysis is performed.

8.2. Privacy Threshold Analysis

The Problem diagrams representing the functional requirements of the system-to-be and the domains involved in their fulfillment, and the Domain knowledge diagrams capturing the privacy-related domain knowledge that was elicited in the step Privacy Context Elicitation form the inputs of the step Privacy Threshold Analysis. Both inputs are retrieved from the Problem Frame Model. The purpose of the step Privacy Threshold Analysis is to elicit the personal data processed by

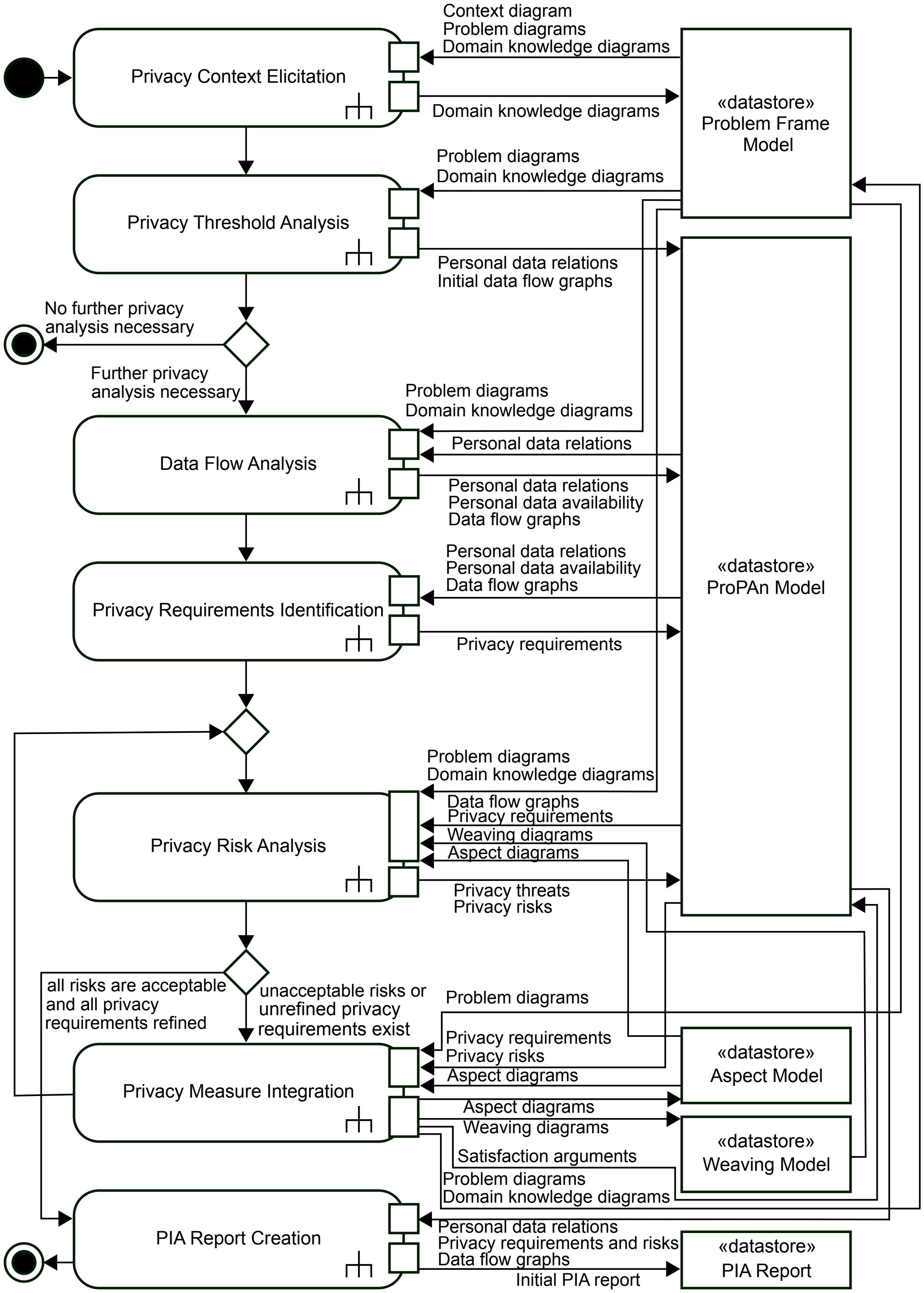


Figure 8.1.: *Overview of the ProPAn method*

the system-to-be and their relations to the corresponding data subjects (Personal data relations). Additionally, Initial data flow graphs are created that over-approximate the flow of personal data through the system-to-be due to the functional requirements and the domain knowledge. The Personal data relations and the Initial data flow graphs are stored in a ProPAn model that is created for the system-to-be. The ProPAn meta-model was already partly introduced in Chapter 7. I introduce the ProPAn metamodel successively in the chapters which detail the steps of the ProPAn method and the artifacts used in these.

Based on the identified personal data and the initial data flow graphs, the analysis team has to decide whether a further privacy analysis is necessary, or not. For example, if no personal data is processed by the system-to-be, then no further privacy analysis is necessary and the ProPAn method is finished. Another example is that the initial data flow graph shows that sensitive personal data possibly flows to a domain to which the personal data shall not be available. In that case, a further privacy analysis is necessary and the ProPAn method is continued by performing the step Data Flow Analysis. I describe the step Privacy Threshold Analysis in more detail in Chapter 10.

8.3. Data Flow Analysis

The goal of the step Data Flow Analysis is to assess how the identified personal data flows through the system-to-be based on its functional requirements and domain knowledge. The functional requirements are provided as Problem diagrams and the domain knowledge as Domain knowledge diagrams, both stored in the Problem Frame Model. Additionally, the Personal data relations elicited during the Privacy Threshold Analysis are used as inputs. The inputs are used to systematically elicit at which domains which personal data are available (Personal data availability), whether other personal data are derived from the already identified personal data during the processing (Personal data relations), and how the identified personal data flows through the system due to the functional requirements and domain knowledge (Data flow graphs). These three outputs are all stored in the ProPAn Model. The details on this steps can be found in Chapter 11. After this step, the privacy requirements can be identified based on the elicited and intended flow of personal data.

8.4. Privacy Requirements Identification

During the Privacy Requirements Identification, the Personal data relations, Personal data availability, and the Data flow graphs that together describe the intended personal data processing behavior of the system-to-be are used as inputs to derive the Privacy requirements that the system-to-be shall satisfy. All inputs and outputs of this step are stored in the ProPAn Model. I explain this step in more detail in Chapter 12. The following step is the Privacy Risk Analysis.

8.5. Privacy Risk Analysis

The goal of the Privacy Risk Analysis step is to identify threats to the system's privacy requirements (Privacy threats) and to evaluate their implied risks (Privacy risks). These two outputs are persisted in the ProPAn Model. The privacy risk analysis is performed based on the Problem diagrams and Domain knowledge diagrams of the Problem Frame Model, the Data flow graphs created during the step Data Flow Analysis, and the Privacy requirements derived during the step Privacy Requirements Identification. If the step Privacy Measure Integration was already performed at least once (cf. iteration in Figure 8.1) and privacy measures in the form of cross-cutting functional requirements (also called *aspects*) and represented as Aspect diagrams were *weaved* into the functional requirements using Weaving diagrams, then these are also used as inputs to the Privacy Risk Analysis to evaluate their impact on the identified privacy threats and risks. The nature of an aspect is, that it describes functionality that is relevant in other functionalities, e.g., logging or encryption. The idea of aspect-oriented requirements engineering (AORE) is to identify such cross-cutting concerns and to isolate them from the functionalities they shall be integrated into by introducing placeholders for domains and interfaces (so-called *join points*) that need to be instantiated when the aspect is integrated into a functional requirement. The integration of aspects into functional requirements is called *weaving*. The concepts of aspect-oriented requirements engineering are explained in more detail in Chapter 14.

After the analysis and evaluation of the threats and risks to the privacy requirements, the analysis team has to decide whether the identified risks are acceptable and all privacy requirements are sufficiently refined to functional requirements, or whether an unacceptable risk was identified or a privacy requirement exists that is not yet sufficiently operationalized. In the first case, the privacy analysis part of the ProPAn method is finished and the step PIA Report Creation can be performed. In the second case, the step Privacy Measure Integration has to be performed to mitigate the identified privacy risks and to refine the privacy requirements. I explain the step Privacy Risk Analysis in Chapter 13.

8.6. Privacy Measure Integration

During the step privacy measure integration, the analysis team considers the identified Privacy requirements, Privacy threats, and Privacy risks that are not yet sufficiently operationalized or mitigated. To operationalize privacy requirements or to mitigate privacy risks, PETs that are represented in Aspect diagrams are used. If the needed PETs were already documented in Aspect Models, these can be reused. This is, because the PETs are described mostly independently from a concrete system in Aspect Models as cross-cutting functional requirements. The instantiation of the join points and the specification how the Aspect diagrams are combined with the Problem diagrams is documented in Weaving diagrams. The Weaving diagrams are stored in a Weaving Model that is created for the system-to-be. This model connects the aspects described in the Aspect Model (independent of the system-to-be) with the Problem Frame Model that contains the requirements model of the system-to-be. If no Aspect Model exists for a PET to be used, Aspect diagrams are created to represent the used PETs, and these are then stored in an Aspect Model. This allows the analysis team to reuse the Aspect diagrams during the development of another system. The metamodels of the Aspect Model and the Weaving Model are introduced in Chapter 14.

After the integration of PETs into the functional requirements of the system-to-be, the step Privacy Risk Analysis has to be performed again by the analysis team to check whether the PETs sufficiently operationalize the privacy requirements, reduce the risks, and do not introduce new unacceptable risks. It is possible that a few iterations are needed until all identified privacy risks are acceptable and all privacy requirements are operationalized. The step Privacy Measure Integration is explained in-depth in Chapter 17.

8.7. PIA Report Creation

In the last step of the ProPAn method, the produced artifacts stored in the ProPAn Model are used to create a textual and human-readable Initial PIA report (PIA stands for privacy impact assessment) that documents the personal data processed by the system-to-be, how these flow through the system, the privacy requirements (including their refinements), and the privacy risks (including their mitigations). This report documents the privacy needs and implications of the system-to-be and how these are planned to be addressed. Performing privacy impact assessments is mandatory in different legislations, e.g., the need to perform data protection impact assessments in cases where personal data are likely to be processed is formulated in the EU General Data Protection Regulation (European Commission, 2016). I explain how the artifacts produced during the ProPAn method can be used to support the creation of PIA reports in Chapter 18 as part of the evaluation of the ProPAn method.

8.8. Relations Among the Used Models

I illustrate the relations between the four models that are used as part of the ProPAn method in Figure 8.2. The figure shows that the Problem Frame Model, the ProPAn Model, and the Weaving Model are created for the system-to-be, while the Aspect Model is designed to be independent of the system-to-be. This allows the analysis team to reuse the modeled aspects for the development of different systems. The link between the Aspect Model and the Problem Frame Model is established by the Weaving Model that references these. The ProPAn Model references the requirements specification and system model contained in the Problem Frame Model and the description of the privacy measures contained in the Weaving Model. In this way, the Problem Frame Model and the Aspect Model do not depend on the other models. This allows the analysis team to reuse them in other contexts. That is, the Aspect Model may provide PETs to be used for other software developments, and the Problem Frame Model may be used in contexts where the software quality privacy is not relevant and the model should not be extended with privacy-related artifacts.

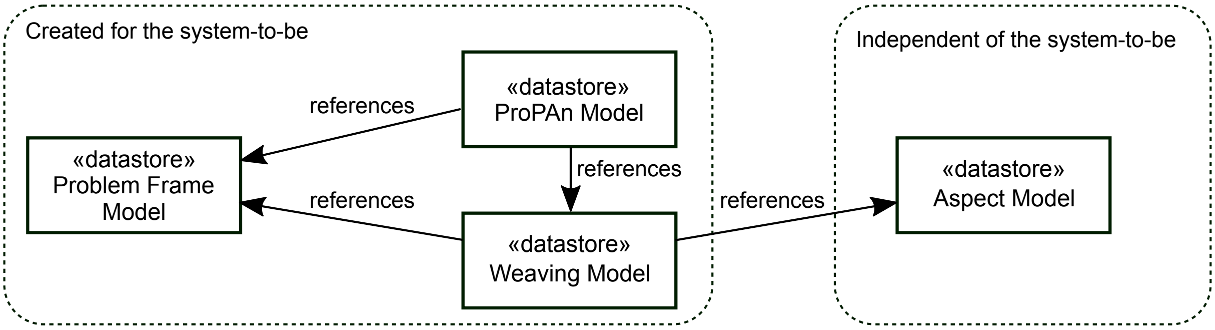

Created for the system-to-be
«datastore» ProPAn Model
references
«datastore» Problem Frame Model
references
references
«datastore» Weaving Model
references
Independent of the system-to-be
«datastore» Aspect Model

Figure 8.2.: *Relations among the models used by the ProPAn method*

8.9. Relation of the ProPAn Method to the High-level Privacy Requirements Engineering Methodology

In Section 3.2.2, I introduced a high-level privacy requirements engineering methodology consisting of the five steps Extend req. specification and system model, Elicit privacy requirements, Elicit privacy risks, Refine privacy requirements, and Treat privacy risks (cf. Figure 3.3 on page 36). The ProPAn method supports all these tasks.

The task Extend req. specification and system model is realized by the steps Privacy Context Elicitation, Privacy Threshold Analysis, and Data Flow Analysis. In these steps, the requirements specification is extended with additional domain knowledge that is relevant for a privacy analysis, the system model is enriched with information about what personal data of which data subjects is processed by the system-to-be, in which quality the personal data are available at the domains of the system-to-be, and due to which functional requirements and domain knowledge the personal data flows through the system.

The steps Privacy Requirements Identification and Privacy Risk Analysis implement the tasks Elicit privacy requirements and Elicit privacy risks, respectively. Both steps utilize for the identification of the privacy requirements and risks all artifacts produced during the first three steps of the ProPAn method.

The step Privacy Measure Integration implements the two tasks Refine privacy requirements and Treat privacy risks that are subtasks of the process Privacy Requirements Operationalization. For both tasks, the step Privacy Measure Integration has corresponding substeps (introduced in Chapter 17) that are concerned with refining privacy requirements and treating privacy risks.

CHAPTER 9

Privacy Context Elicitation

The first step of the ProPAn method is to elicit additional privacy-relevant context information based on a given problem frame model. I introduce this step in detail in this chapter. The context elicitation is separated into two parts. First, the indirect environment, i.e., domains that are not part of the direct environment of the machine, is elicited. This indirect environment can contain data subjects whose personal data are processed, counterstakeholders who may get access to the personal data processed by the system-to-be, and interfaces between known domains that are not relevant for the functional behavior of the system-to-be, e.g., communication between biddable domains. Second, interfaces between domains may need to be refined by introducing connection domains that mediate between the domains connected by the interface. It is especially important to make connection domains explicit if these introduce additional and possibly unintended information flows. The first part is based on my paper (Meis, 2014), while the second part is mainly based on (Beckers et al., 2014a). I am the main author of the latter paper. Kristian Beckers, Stephan Faßbender and Maritta Heisel provided the PACTS method (Beckers et al., 2013a) as background of the paper and valuable comments on the contribution of the paper. Stefanos Gritzalis and Christos Kalloniatis contributed the real-life case study used in the paper.

This chapter is structured as follows. Section 9.1 gives an introduction. The elicitation process for the indirect environment is described in Section 9.2 and guidelines to model the indirect environment using domain knowledge diagrams are given in Section 9.3. Section 9.4 introduces the elicitation method for connection domains and Section 9.5 describes how the elicited connection domains shall be modeled using domain knowledge diagrams. In Section 9.6, I present a small empirical experiment evaluating the elicitation questionnaires that was conducted during the presentation of (Meis, 2014) at the IFIP Summer School on Privacy and Identity Management for Emerging Services and Technologies held 17-21 June, 2013 in Nijmegen. The tool support for this step of the ProPAn method is described in Section 9.7. Related work and how other privacy requirements engineering methods deal with the elicitation and documentation of the system's indirect environment is discussed in Section 9.8. I finally conclude this chapter in Section 9.9.

9.1. Introduction

The quality of a privacy analysis strongly depends on the domain knowledge which is considered during the analysis. The elicitation of domain knowledge for the development of a software system has normally a limited scope to limit the costs of the analysis. For an assessment of a system's functionalities, often only those stakeholders and domains are identified that directly take part or are part of a functionality of the system-to-be. I call this set of entities the *direct environment* of the machine. For a privacy analysis, the scope has to be widened, because privacy requirements on a system or threats to these can originate or stem from the *indirect*

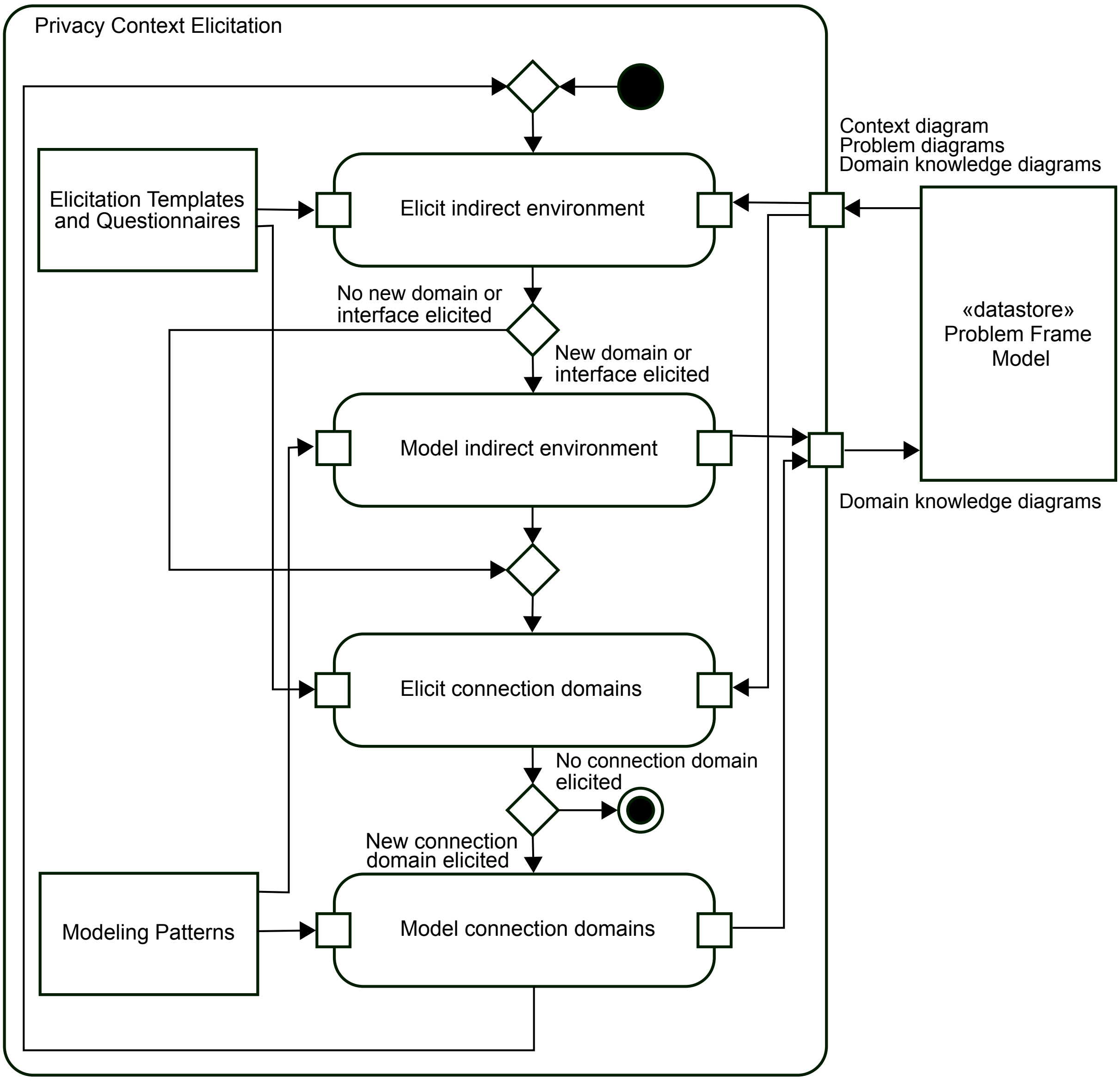

Privacy Context Elicitation
Elicitation Templates and Questionnaires
Elicit indirect environment
Context diagram
Problem diagrams
Domain knowledge diagrams
«datastore» Problem Frame Model
No new domain or interface elicited
New domain or interface elicited
Model indirect environment
Domain knowledge diagrams
Elicit connection domains
No connection domain elicited
New connection domain elicited
Modeling Patterns
Model connection domains

Figure 9.1.: *Detailed view on the step* Privacy Context Elicitation *of the ProPAn method*

environment. The indirect environment may contain data subject whose personal data are processed by the system and whose privacy is vulnerable, or counterstakeholders who possibly get access to personal data and hence, affect the data subjects' privacy negatively. Furthermore, (early) functional requirements often do not mention connection domains that refine interfaces between domains. In most cases this is done to abstract from the technical realization of these interfaces. When personal data are communicated using such an interface, it is important to check whether a connection domain mediates between the domains of the interface and possibly stores the personal data for a period or possibly allows access to the personal data to others.

To address the identification of the privacy-relevant indirect environment of the machine, I propose a stepwise method to elicit and model the indirect environment for a system-to-be based on a given problem frame model. The elicited indirect environment is added to the same problem frame model using domain knowledge diagrams. The method is supported by questionnaires and templates that support an analysis team to elicit the indirect environment, and by modeling patterns that support the documentation of the identified knowledge. I provide generic questionnaires to elicit the indirect environment and connection domains, however, the method can further be supported by providing questionnaires and templates that are specific to an application domain. As an example of such application-domain-specific material, I consider the domain of cloud computing.

Figure 9.1 shows the substeps, used artifacts, and the process flow that refine the step Privacy Context Elicitation of the ProPAn method. The first step is Elicit indirect environment. This step takes the context, problem, and domain knowledge diagrams of the Problem Frame Model for the system-to-be as input. Additionally, Elicitation Templates and Questionnaires are used to assist the analysis team during the elicitation. If new domains or interfaces are identified during the first step, these are modeled using Modeling Patterns and added to the Problem Frame Model in domain knowledge diagrams during the step Model indirect environment. Otherwise, the process is continued with the step Elicit connection domains. Similar to the step Elicit indirect environment, the step Elicit connection domains takes the context, problem, and domain knowledge diagrams of the Problem Frame Model and Elicitation Templates and Questionnaires as input. If a connection domain is identified during the elicitation step, it is added to the Problem Frame Model in domain knowledge diagrams by applying Modeling Patterns. Otherwise, the Privacy Context Elicitation step is done. After the step Model connection domains, the process is started again with the step Elicit indirect environment. This is done to analyze the indirect environment of the newly identified connection domains.

The details on the four steps, the elicitation templates and questionnaires, the modeling patterns, and an exemplary application of these steps on the electronic health system (EHS) scenario (see Chapter 4) are provided in the following four sections.

9.2. Elicit Indirect Environment

For the elicitation of the domain knowledge on the indirect environment, I use questionnaires. All questions aim at the elicitation of indirect data subjects, counterstakeholders, or at the identification of hidden information flows in the considered system. Indirect data subjects and counterstakeholders are data subjects and counterstakeholders who are not yet considered in the system-to-be, because they do not have a direct interface to the machine. The questionnaires have to be answered for a single domain of the system-to-be. I distinguish two kinds of questions. Questions with the prefix *GE1* elicit counterstakeholders who can gain personal data from the domain under consideration. Questions with the prefix *GE2* elicit data subjects whose personal data are processed by the domain. I developed general questionnaires for the different domain types (lexical, causal, and biddable (cf. Section 2.2)) to provide questions tailored to these domain types. These general questionnaires are introduced in Section 9.2.1. Additionally, specific application domains, e.g., cloud computing and the internet of things (IoT), introduce themselves an indirect environment containing counterstakeholders or data subjects. Hence, it may be valuable to create application-domain-specific questionnaires and templates that assist analysis teams to identify privacy-relevant domain knowledge for specific application domains. In Section 9.2.2, I present questionnaires and templates that support the identification of privacy-relevant counterstakeholders when the system-to-be shall (partially) be realized using cloud computing. During the elicitation process, the analysis team has to answer the questionnaires for all domains documented in the given problem frame model. I illustrate the application of this substep in Section 9.2.3 using the EHS example.

9.2.1. General Elicitation Questionnaire

The questionnaire for causal and lexical domains is shown in Table 9.1. I refined the first question type using the Volere stakeholder analysis template proposed by Alexander and Robertson (2004). This template suggests the *negative stakeholders* competitor (question GE1.1) and

Table 9.1.: *Domain knowledge elicitation questionnaire for causal and lexical domains*

No.	Question
GE1	**Elicitation of Counterstakeholders**
GE1.1	Is there a competitor that also uses the domain?
GE1.2	Could the domain be attacked by a hacker?
GE1.3	Does the domain provide information to legislators, law enforcement agencies, or other authorities?
GE1.4	Is the domain also used in other systems? State possible counterstakeholders that have access to the domain in these systems.
GE2	**Elicitation of Data Subjects**
GE2.1	Is the domain also used in other systems? State possible data subjects whose personal data may be accessible through the domain.
GE2.2	Is initially personal data of data subjects stored in the domain?
GE2.3	Does the domain store or process personal data of data subjects directly, indirectly, or implicitly connected to it?

hacker (question GE1.2). Furthermore, I refined question type GE1 by asking for the *baseline stakeholder* legislator (question GE1.3) suggested by Sharp et al. (1999). Additionally, I added the possible counterstakeholders law enforcement agency and authority to question GE1.3. Competitors, hackers, legislators, law enforcement agencies, and other authorities are all possible indirect counterstakeholders that are usually not considered as part of the direct environment of a software. Questions GE1.4 and GE2.1 elicit counterstakeholders or data subjects that can gain or provide personal data due to a re-use of a domain. Causal or lexical domains can themselves be sources of personal data, e.g., an existing database with contact information of customers can be modeled as lexical domain. Question GE2.2 elicits the data subjects of these personal data. Systems may contain hidden information flows, i.e., storage or processing of data subjects' personal data that are directly, indirectly, or implicitly connected to the domain. Question GE2.3 elicits from which data subjects a hidden information flow exists to the domain.

The questionnaire for biddable domains is shown in Table 9.2. Question GE1.1 aims at the trustworthiness of a biddable domain in the system-to-be. With this question, I want to identify indirect counterstakeholders with whom the biddable domain possibly shares personal data retrieved from the system-to-be. Hence, question GE1.1 elicits the source of so-called *social engineering attacks*. Questions GE1.2 and GE2.1 elicit implicit communications between biddable domains in the system. Question GE1.3 is the same as question GE1.3 from the previous questionnaire. Question GE2.2 elicits those indirect data subjects for whom a domain in the direct environment acts on behalf of. These indirect data subjects are of high relevance for the privacy analysis because their personal data are stored and processed in the system-to-be in all likelihood.

It is reasonable to extend both questionnaires with questions specific to an application domain to give further assistance for the elicitation process. The questionnaires are easily extensible with questions aiming at the elicitation of privacy-relevant counterstakeholders (question type GE1) and data subjects (question type GE2). In the following section, I introduce a template and questionnaire to elicit cloud-specific counterstakeholders.

9.2.2. Cloud-specific Elicitation Templates and Questionnaires

In the case that the machine or parts of it shall be realized using a cloud infrastructure or storage, this cloud is identified in the step Elicit connection domains as connection domain (see Section 9.4), or is already part of the given problem frame model. For a domain representing

Table 9.2.: *Domain knowledge elicitation questionnaire for biddable domains*

No.	Question
GE1	**Elicitation of Counterstakeholders**
GE1.1	Is the biddable domain vulnerable to social engineering attacks?
GE1.2	Does the biddable domain provide information to another biddable domain?
GE1.3	Does the biddable domain provide information to legislators or law enforcement agencies?
GE2	**Elicitation of Stakeholders**
GE2.1	Does the biddable domain get information of another biddable domain?
GE2.2	Does the biddable domain act on behalf of customers or wards (e.g. children)?

Table 9.3.: *Overview of cloud stakeholders and their properties in the cloud deployment scenarios*

Group	Stakeholder	Private	Community	Public
Provide and maintain cloud	Cloud Provider	yes	yes	yes
	Cloud Administrator	yes	yes	maybe
	Cloud Support	yes	yes	maybe
Use cloud to build services	Cloud Customer	yes	yes	no
	Cloud Developer	yes	yes	no
Use Services	End Customer	yes	maybe	no
Indirect Environment	Legislator	yes	maybe	no

a cloud or a part of it, different indirect counterstakeholders may be relevant. For the privacy analysis, the analysis team is interested in the counterstakeholders that are able to access the data provided to the cloud. These counterstakeholders vary for different cloud types. Beckers et al. (2013a) identified for their PACTS method stakeholders that are relevant for clouds. The authors represent the cloud stakeholders and their relationship to the cloud using a cloud system analysis pattern. I derived the seven possible counterstakeholders listed in Table 9.3 from the cloud system analysis pattern. Table 9.3 groups the stakeholders of the cloud system analysis pattern into four groups. The first group consists of the stakeholders that provide and maintain the cloud. These are the *Cloud Provider* that provides the cloud, and the *Cloud Support* and *Cloud Administrator* that both work for the cloud provider and have directly or indirectly access to the cloud. The second group summarizes the stakeholders that use the cloud to build services. These are the *Cloud Customer*, who deploys his or her infrastructure and services into the cloud of the cloud provider, and the *Cloud Developer*, who works for the cloud customer. The third group consists of the stakeholders that use the services that are run in the cloud. Only the *End Customer* of the cloud customer belongs to this group. The last group is the indirect environment of the cloud. I consider *Legislator*s as relevant stakeholders, as they are possibly allowed to access the data of the cloud due to regulations. The relevant legislators for a cloud are given by the locations of the cloud, cloud provider, cloud customer, and end customer.

Commonly, three cloud deployment scenarios are distinguished: *private*, *community*, and *public* clouds (Mell and Grance, 2011). A private cloud is created by a company or organization for private and internal usage. A community cloud is shared among several companies that have similar requirements on the cloud infrastructure to reduce the costs to build and maintain the cloud. Providers of public clouds sell their cloud infrastructure and services to customers and end customers. Depending on the kind of cloud, the cloud-specific counterstakeholders may be

Table 9.4.: *Cloud-specific counterstakeholder questionnaire*

No.	Question
CE1	**Elicitation of Counterstakeholders**
CE1.1	Is the counterstakeholder known? List all known instances of the counterstakeholder.
CE1.2	Are the listed counterstakeholders fully trusted?

known or not. Table 9.3 gives an overview whether these generic counterstakeholders are known in the respective deployment scenario. For each *maybe* entry, the analysis team has to decide whether the respective counterstakeholder is known in the concrete cloud deployment scenario.

For each cloud (part) present in the problem frame model and each possible counterstakeholder listed in Table 9.3, the questions listed in Table 9.4 have to be answered. First, all concrete instances of the cloud stakeholder shall be listed if these are known. Second, the analysis team has to decide whether the identified counterstakeholders are *fully trusted.* I consider a stakeholder as fully trusted if the analysis team can neglect the assumption that the stakeholder introduces privacy issues. This may be guarantied by contractual obligations, regular certifications, or audits of the respective stakeholder. Only those cloud stakeholders that are not fully trusted need to be modeled in the following step.

9.2.3. Application to EHS

For the application of the step Elicit indirect environment, I only consider a subset of the requirements on the EHS listed in Chapter 4, namely R1 (Manage EHRs), R2 (Browse EHRs), R3 (Accounting), and R4 (Billing). Note that the biddable domain patient is not part of the corresponding problem diagrams and hence, may be overlooked as data subject, because patients do not directly interact with the machine in the subsystem described by the four requirements.

The five domains for which the questionnaires have to be answered are the biddable domain doctor, the two lexical domains EHR and invoice, and the two causal domains insurance application and financial application. Table 9.5 shows the answers to the questionnaires for the five domains. For the lexical domains EHR and invoice, I do not identify additional counterstakeholders, because these shall exclusively used by the EHS and no direct access to them shall be allowed for other domains. I identified the patient as possible data subject of whom personal data may be initially existing in the domain EHR. For the invoices, I do not consider that these initially contain data. The financial and insurance application are similar concerning the indirect environment they introduce. Both domains may be vulnerable to hacker attacks, may have to provide data to tax authorities, and have employees and customers that also use the applications for different purposes. As doctors are bound to professional discretion, I do not consider them to be vulnerable to social engineering attacks. However, it would be possible to consider malicious doctors who break their professional discretion, and to elicit the counterstakeholders to whom they possibly disclose information. Doctors may provide information of patients' health status to the patients' families and may also get information from the patients' families concerning family diseases. Additionally, doctors communicate with patients during the treatment and doctors act on behalf of patients in the EHS. A longer list of possible indirect counterstakeholders and data subjects for the domains doctor and financial application can be found in Table 9.12, which shows the results of a small empirical evaluation of the general questionnaires (see Section 9.6).

If I assume that cloud-technology shall be used to store the EHRs, then in an earlier application of the step Elicit connection domains (see Section 9.4) a domain *EHS cloud* would have been introduced as connection domain between the machine EHS and the lexical domain EHR. As the EHRs are expected to contain sensitive personal data and a collaboration with other eHealth

Table 9.5.: *Answers to the general questionnaires for the domains EHR, invoice, financial and insurance application, and doctor*

No.	EHR	Invoice	Financial/Insurance App.	Doctor
GE1.1	-	-	-	-
GE1.2	-	-	Hacker	Family of patient
GE1.3	-	-	Tax authority	-
GE1.4	-	-	Employee, Customer	-
GE2.1	-	-	Employee, Customer	Family of Patient, Patient
GE2.2	Patient	-	-	Patient

Table 9.6.: *Answers to the cloud-specific questionnaires for a private cloud storing the EHRs*

Stakeholder	CE1.1	CE1.2
Cloud Provider	EHS provider	yes
Cloud Administrator	EHS administrator	no
Cloud Support	EHS administrator	no
Cloud Customer	EHS provider	yes
Cloud Developer	EHS developer	yes
End Customer	EHS provider	yes
Legislator	Germany, EU	yes

providers is not foreseen, a private cloud is considered as deployment scenario. As specified in Table 9.3, all cloud stakeholders are known in the case of a private cloud deployment scenario. Hence, the analysis team needs to list all known cloud stakeholders and has to specify whether these are fully trusted or not. Table 9.6 shows the answers to the cloud-specific questionnaires for all cloud stakeholders. The EHS provider is considered as cloud provider, cloud customer, and end customer. This is, because the cloud is provided by the EHS provider, the EHS provider is the only stakeholder who creates services based on the cloud, and the EHS provider exclusively uses the services of the cloud. The EHS administrators takes the role of the cloud stakeholders Cloud Administrator and Cloud Support, and the EHS developers take also the role of the Cloud Developer. As relevant Legislators, I consider the German state and the European Union (EU), because the EHS shall be operated in Germany. I consider the EHS provider, EHS developer, and the legislators Germany and the EU to be fully trusted. For demonstration purposes, I do not consider the EHS administrator as fully trusted.

9.3. Model Indirect Environment

In this section, I explain the substep Model indirect environment. In this step, the elicited indirect environment is added to the problem frame model using domain knowledge diagrams. The domain knowledge diagrams can be obtained by instantiating modeling patterns. In Section 9.3.1, I explain how the answers to the general questionnaires are translated to domain knowledge diagrams and Section 9.3.2 describes how the cloud-specific answers are translated. The modeling of the indirect environment of the EHS example is shown in Section 9.3.3.

9.3.1. General Modeling Guidelines

I propose to use *domain knowledge diagrams* (see also Section 2.2.4) to model the domain knowledge about the domains that were elicited using the questionnaires in the previous step. These domain knowledge diagrams are added to the same problem frame model that models the

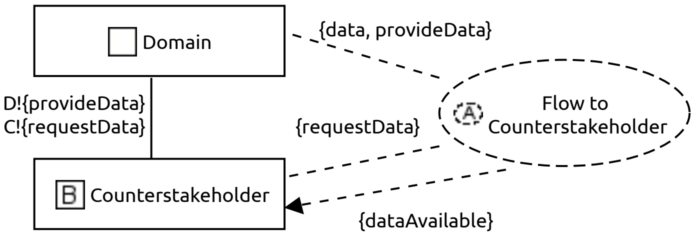


Figure 9.2.: *Pattern for a domain knowledge diagram for question type GE1*

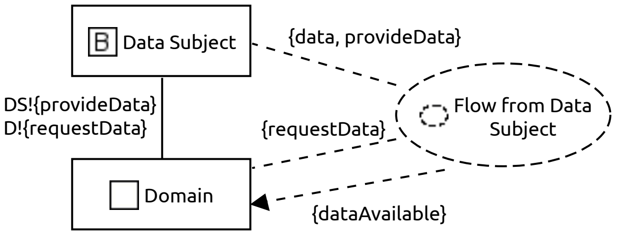


Figure 9.3.: *Pattern for a domain knowledge diagram for question type GE2*

functional requirements of the system-to-be.

In general, I differentiate two kinds of elicited domain knowledge. First, the analysis team may elicit indirect counterstakeholders who can gain information from the system-to-be (question type GE1). Second, the analysis team may elicit indirect data subjects of whom personal data are possibly stored and processed by the system-to-be (question type GE2). The identified indirect counterstakeholders and data subjects are modeled as biddable domains in both cases, because the questions aim at identifying natural or legal persons. To represent the first kind of domain knowledge, we create a domain knowledge diagram by instantiating the pattern shown in Figure 9.2. In the other case, we instantiate the pattern shown in Figure 9.3. The domain Domain is instantiated with the domain for which the analysis team answered the question, and the Counterstakeholder and Data Subject with the newly identified biddable domain. The analysis team has to select an appropriate name and description for the domain knowledge that represents the fact or assumption that data may be provided from the Domain to the Counterstakeholder, or that data of the Data Subject are available at the Domain. The domain knowledge is either a fact if it is a truth that always holds, or an assumption that could also be false under some circumstances. A *refers to* reference (dashed line without arrow head) is drawn between the domain knowledge and the source of the information flow (for question type GE1 the Domain and for GE2 the Data Subject) and a *constrains* reference (dashed line with arrow head) is drawn between the domain knowledge points and the target of the information flow (for question type GE1 the Counterstakeholder and for GE2 the Domain). In addition to the constrains reference, a refers to reference can optionally be added between the target of the information flow, e.g., if the target initiates the interaction between the two domains. Furthermore, the domain knowledge diagram shall contain an interface between the two involved domains to specify how these communicate with each other to exchange the data. The interface can be very abstract, especially in cases where the data flow is due to another systems. In such cases, the analysis team may decide to refine these abstract interfaces by introducing connection domains in the step Elicit connection domains (see Section 9.5).

9.3.2. Cloud-specific Modeling Guidelines

Only those indirect counterstakeholders that are not considered as fully trusted have to be modeled in domain knowledge diagrams, because for the others, it is assumed that they will not

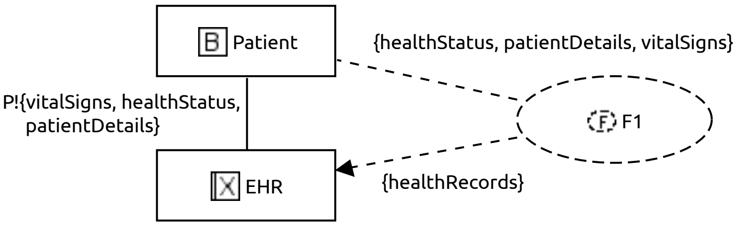

B Patient
{healthStatus, patientDetails, vitalSigns}
P!{vitalSigns, healthStatus, patientDetails}
F F1
X EHR
{healthRecords}

Figure 9.4.: *Domain knowledge for the domain EHR*

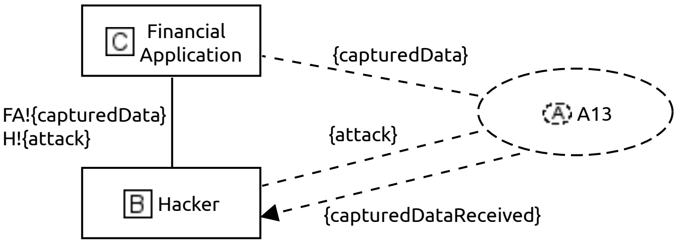


Figure 9.5.: *Domain knowledge about the indirect counterstakeholder hacker for the domain financial application*

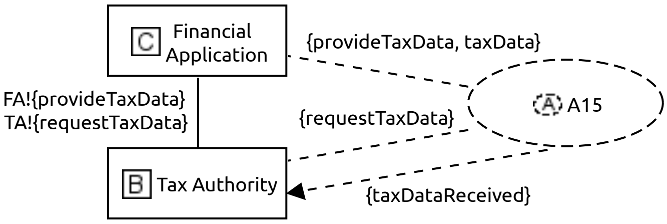


Figure 9.6.: *Domain knowledge about the indirect counterstakeholder tax authority for the domain financial application*

introduce privacy issues. To model the indirect counterstakeholders, the domain knowledge pattern for question type GE1 (see Figure 9.2) is instantiated. If a cloud stakeholder is unknown (i.e., question CE1 was answered with *no*), then a biddable domain is added as counterstakeholder with the respective cloud stakeholder role and the prefix *unknown* as name. e.g., Unknown End Customer.

9.3.3. Application to EHS

For the EHS example, I identified for the domains EHR, financial and insurance application, and doctor the indirect counterstakeholders and data subjects shown in Table 9.5 in the previous step. Additionally, I elicited indirect counterstakeholders for a private cloud scenario and decided that the EHS administrator is not fully trusted and should be modeled for the further privacy analysis (cf. Table 9.6). Note that the identification of the EHS cloud as relevant connection domain is described in step Elicit connection domain (see Section 9.4) and its integration into the problem frame model in step Model connection domain (see Section 9.5).

The domain knowledge diagram for the lexical domain EHR is shown in Figure 9.4. The domain knowledge diagram is an instance of the domain knowledge pattern shown in Figure 9.3. Fact F1 documents that Patients' healthStatus, patientDetails, and vitalSigns are already contained in the healthRecords managed by the EHR due to the re-use of EHRs created with another eHealth system. Note that the interface between Patient and EHR abstracts away the other EHS system with which the health records were created. If this EHS system is expected to be of relevance, it could be made explicit during the steps Elicit connection domains and Model connection domains.

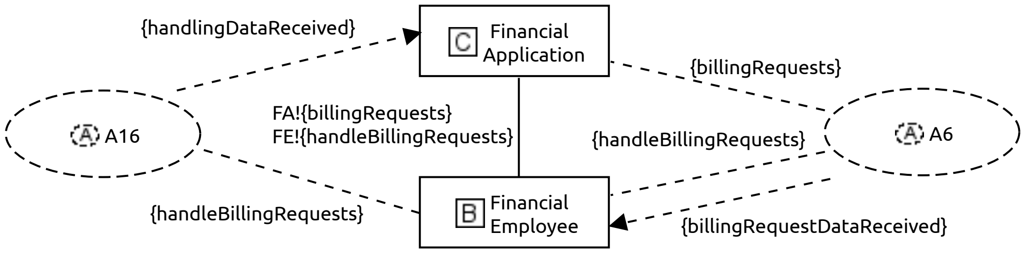


Figure 9.7.: *Domain knowledge about the indirect counterstakeholder employee for the domain financial application*

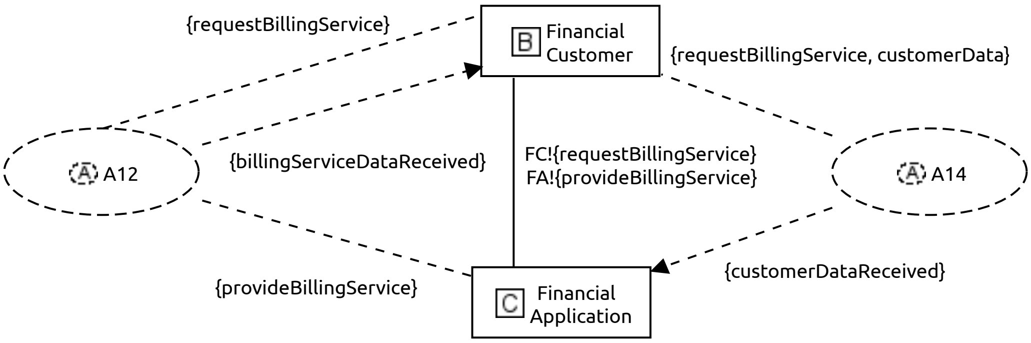

{requestBillingService}
B Financial Customer
{requestBillingService, customerData}
A12
{billingServiceDataReceived}
FC!{requestBillingService}
FA!{provideBillingService}
A14
{customerDataReceived}
{provideBillingService}
C Financial Application

Figure 9.8.: *Domain knowledge about the indirect counterstakeholder customer for the domain financial application*

As the financial and insurance application introduce similar indirect counterstakeholders, I only show the domain knowledge diagrams for the financial application. Figures 9.5 and 9.6 are an instances of the domain knowledge pattern for question type GE1 (see Figure 9.2) and Figures 9.7 and 9.8 are obtained using both patterns by instantiating Domain with Financial Application, and Counterstakeholder, as well as, Data Subject with Financial Employee and Financial Customer. Assumption A13 states that a Hacker may attack the Financial Application to capture data (cf. Figure 9.5). Figure 9.6 shows assumption A15, which states that a Tax Authority may request tax data, which is then provided by the Financial Application. The Financial Employee is considered as counterstakeholder (assumption A6 in Figure 9.7) and as data subject (assumption A16). That is, a Financial Employee may receive data about the billing requests from the Financial Application, and the Financial Application may receive data about how the billing request is handled by the Financial Employee. Similarly, a Financial Customer may use the services provided by the Financial Application (see Figure 9.8). Consequently, the Financial Customer may gain data from the Financial Application related to the requested services (assumption A12), and the Financial Application gets data from the Financial Customer to provide the requested service (assumption A14).

The domain knowledge diagram for the doctor is shown in Figure 9.9. It documents that a Doctor gets to know the vital signs, health status, and patient details (including contact information and demographics) from his or her Patient (assumption A1). Additionally, Doctors may get information about the patient's health status or family diseases from the patient's Family (assumption A3), and the Family may gain information about the patient's health status from the Doctor (assumption A2).

The domain knowledge about the cloud-related indirect counterstakeholder EHS Administrator of the connection domain EHS Cloud (introduced in the next step) is shown in Figure 9.10. Assumption A20 documents that the EHS Administrator administrates the EHS Cloud and may gain specific data from the EHS Cloud during this task.

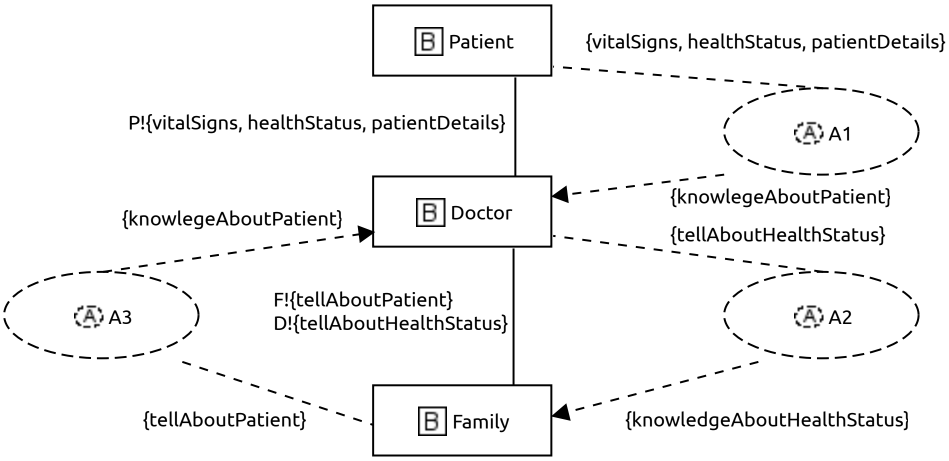


Figure 9.9.: *Domain knowledge for the domain doctor*

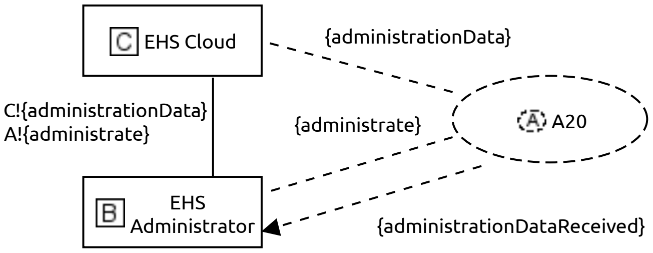


Figure 9.10.: *Domain knowledge about the indirect counterstakeholders for the domain cloud*

9.4. Elicit Connection Domains

I present the details on the substep Elicit connection domains in this section. The step is concerned with the identification of privacy-relevant connection domains, i.e., domains that refine interfaces between domains in the problem frame model and that possibly provide personal data of data subjects to domains of the system-to-be or provide data to counterstakeholders. Similar to substep Elicit indirect environment, I propose templates and questionnaires to support the elicitation process. The general and cloud-specific elicitation processes are introduced in Section 9.4.1 and Section 9.4.2, respectively. The substep is applied on the EHS example in Section 9.4.3.

9.4.1. General Elicitation Questionnaire

To elicit privacy-relevant connection domains, the analysis team has to answer the three questions listed in Table 9.7 for each interface occurring in the context diagram, or a problem or domain knowledge diagram. That is, for each interface between domains, it has to be decided whether the interface abstracts from a connection domain that mediates between these (question GC1). Then the analysis team has to list the phenomena that are refined by the identified connection domain (question GC2). That means, it is possible that a connection domain refines an interface only partially. Finally, the analysis team has to decide whether this connection domain possibly introduces indirect counterstakeholders or data subjects (question GC3). Note that these indirect counterstakeholders and data subjects are elicited during the step Elicit indirect environment after the connection domain was modeled in step Model connection domain (cf. Figure 9.1 on page 134).

Table 9.7.: *General connection domain elicitation questionnaire*

No.	Question
GC1	Does the interface abstract from a connection domain that mediates between the domains?
GC2	Which phenomena of the interface are refined by the connection domain?
GC3	Does the connection domain possibly process personal data of indirect data subjects or provide intendedly or unintendedly data to indirect counterstakeholders?

Table 9.8.: *Cloud-specific connection domain elicitation questionnaire*

No.	Question
CC1	Is the domain virtualized by or stored in a cloud? Name the cloud which shall be used.

Table 9.9.: *Answer to the general connection domain questionnaire for the interface between EHS and doctor*

Interface	GC1	GC2	GC3
EHS - Doctor	I/O device	all	no

9.4.2. Cloud-Specific Elicitation Templates

If it is known or planned that specific domains are virtualized in a cloud or data represented by a lexical domain are stored in a cloud, then a respective cloud should be added to the problem frame model as connection domain. Hence, the question listed in Table 9.8 has to be answered for each domain in the problem frame model. A positive answer to question CC1 means that all interfaces of the domain that is virtualized or stored in cloud are refined by the identified cloud (corresponds to GC1 in Table 9.7), that all phenomena of the interfaces are refined by the cloud (corresponds to GC2), and that there are possibly indirect counterstakeholders who have access to the cloud as discussed in Section 9.2 (corresponds to GC3). Hence, answers to the cloud-specific questionnaire can directly be translated to answers of the general questionnaire.

9.4.3. Application to EHS

The context diagram for the EHS shown in Figure 4.1 on page 66 provides an overview of all interfaces in the EHS (except the interfaces introduced by the domain knowledge diagrams shown in Figures 9.4-9.10). All causal domains in the context diagram already represent connection domains that I do not further refine. The interface between the EHS and the Doctor can be refined with the I/O device that the Doctor uses to access the EHS. The I/O device shall only be used by doctors, hence, no indirect data subjects and counterstakeholders are expected (see Table 9.9).

As mentioned earlier, if I assume that the electronic health records shall be stored using cloud technology, then I answer question CC1 (see Table 9.8) positively. The cloud used is called EHS cloud (cf. Table 9.10). As mentioned before, the answer to the cloud-specific questionnaire can be translated to an answer to the general questionnaire. This translation is shown in Table 9.11.

Table 9.10.: *Answer to the cloud-specific connection domain elicitation questionnaire for the domain EHR*

Domain	CC1
EHR	EHS cloud

Table 9.11.: *Answer to the cloud-specific connection domain questionnaire translated to the general connection domain questionnaire*

Interface	GC1	GC2	GC3
EHS - EHR	EHS cloud	all	yes

9.5. Model Connection Domains

The substep Model connection domains is introduced in this section. During this step, the elicited connection domains are modeled using domain knowledge diagrams and added to the problem frame model. I again propose modeling patterns that can be instantiated to obtain the needed domain knowledge diagrams. The general modeling guidelines are presented in Section 9.5.1. Section 9.5.2 explains how the general modeling guidelines can be used to model cloud-specific connection domains. The application of this substep on the EHS example is presented in Section 9.5.3.

9.5.1. General Modeling Guidelines

For each connection domain that was identified in the previous step, the analysis team has to document in the problem frame model which interfaces this connection domain refines. For this, I propose to model the connection domain in domain knowledge diagrams that document the domain knowledge that the connection domain mediates between the involved domains, instead of adding the connection domain to the problem diagrams and refining the functional requirements. This approach allows a modular introduction of connection domains without affecting the already existing artifacts. As interfaces may be refined using different connection domains (e.g., using different cloud deployment scenarios), the modular integration allows to easily assess the impact of different variants without changing the description of the basic functionality of the system-to-be.

To document the domain knowledge that a connection domain refines an interface between domains, the domain knowledge pattern shown in Figure 9.11 can be instantiated. The interface between Domain 1 and Domain 2 that shall be refined by the connection domain is shown in Figure 9.12. It shows that Domain 1 and Domain 2 share the phenomena P1 (controlled by Domain 1) and P2 (controlled by Domain 2). Figure 9.12 additionally shows two statements that describe the effect of the phenomena. Effect of P1 documents that when Domain 1 issues P1, then this has the effect P1effect on Domain 2. Analogously, Effect of P2 documents how the phenomenon P2 affects Domain 1. I introduce these statements to illustrate that the domain knowledge to be documented for a connection domain refining the interface has to allow to reason that the behavior of the connection domain still guarantees these statements.

The phenomena RP1 and FP1 in Figure 9.11 refine the phenomenon P1. RP1 is the refined version of P1 that is controlled by Domain 1 and observed by the Connection Domain. Statement Connection Domain refines P1 documents that relation between P1 and RP1 and constrains that the connection domain issues in consequence the phenomenon FP1. FP1 is controlled by Connection Domain and observable by Domain 2. It refines the phenomenon P1 from the point of view of Domain 2. How FP1 affects Domain 2 is documented in statement Effect of FP1 and also how this effect corresponds to the effect of P1. This domain knowledge shall allow to reason

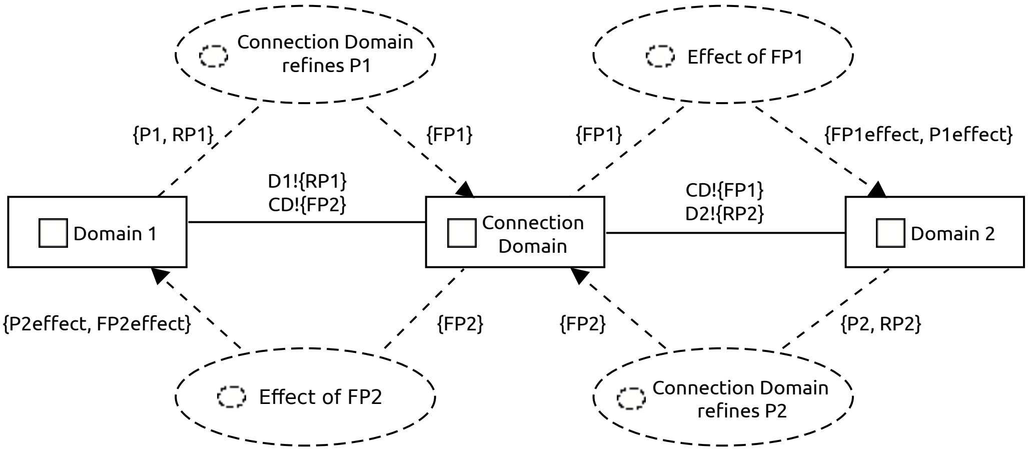


Figure 9.11.: *Pattern for introducing a connection domain refining an interface between two domains*

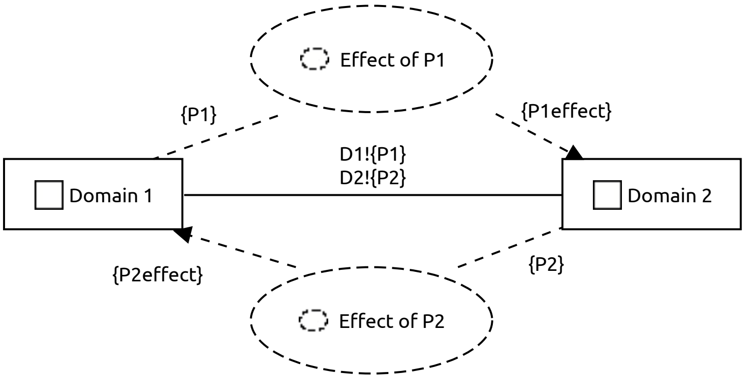


Figure 9.12.: *Statements about the abstract interface between Domain 1 and Domain 2*

that the introduced connection domain still guarantees the statements shown in Figure 9.12. This can be expressed by the following entailment relationship:

$$\mathsf{Connection\ Domain\ refines\ P1}, \mathsf{Effect\ of\ FP1} \vdash \mathsf{Effect\ of\ P1}$$

The same applies analogously for the phenomena P2, FP2, RP2, P2effect, and FP2effect and the statements Connection Domain refines P2, and Effect of FP2. If an interface is only unidirectional, then the respective phenomena and statements are omitted. Note that type of all domains and statements in Figures 9.11 and 9.12 are left open. That means, the domains and statements can be instantiated with arbitrary types (unless these violate the validation conditions for problem frame models introduced in Section 2.3).

After all elicited connection domains are modeled, the Privacy Context Elicitation is continued with the step Elicit indirect environment until no further connection domains are identified (see also Figure 9.1 on page 134).

9.5.2. Cloud-specific Modeling Guidelines

As discussed in the previous section, the answers to the cloud-specific connection domain elicitation questionnaire can be translated to answers of the general connection domain elicitation questionnaire. Hence, the pattern shown in Figure 9.11 can also be used for introducing a cloud as connection domain using the translated answers.

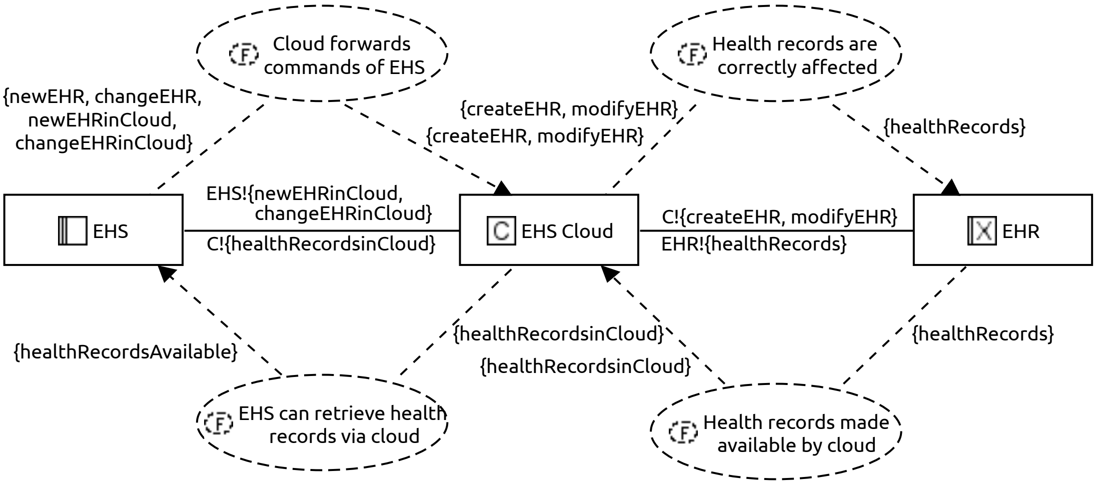

F Cloud forwards commands of EHS
F Health records are correctly affected
{newEHR, changeEHR, newEHRinCloud, changeEHRinCloud}
{createEHR, modifyEHR}
{createEHR, modifyEHR}
{healthRecords}
EHS
EHS!{newEHRinCloud, changeEHRinCloud}
C!{healthRecordsinCloud}
C EHS Cloud
C!{createEHR, modifyEHR}
EHR!{healthRecords}
X EHR
{healthRecordsAvailable}
{healthRecordsinCloud}
{healthRecordsinCloud}
{healthRecords}
F EHS can retrieve health records via cloud
F Health records made available by cloud

Figure 9.13.: *Domain knowledge for the EHS Cloud that stores the EHRs*

9.5.3. Application to EHS

Based on the answers to the elicitation questions shown in Tables 9.9 and 9.11, only the connection domain EHS cloud needs to be modeled. The domain knowledge diagram for the connection domain EHS Cloud is shown in Figure 9.13. It documents that the commands newEHR and changeEHR (cf. Figure 4.1 on page 66) are refined by the commands newEHRinCloud and changeEHRinCloud. The fact Cloud forwards commands of EHS states that the latter commands cause the EHS Cloud to issue the commands createEHR and modifyEHR. How these two commands affect the healthRecords is documented by the fact Health records are correctly affected. Note that I instantiated the phenomena FP1effect and P1effect both with the phenomenon healthRecords, and also the phenomena P2 and RP2 are instantiated with this phenomenon. Fact Health records made available by cloud states that the EHS Cloud makes available the healthRecords of the EHR to the EHS using the phenomenon healthRecordsinCloud. When the EHS observes this phenomenon, the respective health records are available to it (fact EHS can retrieve health records via cloud). Note that I decided to model all statements about the EHS Cloud as facts, because I expect the EHS Cloud to guarantee these statements as it is also provided by the provider of the EHS.

9.6. Empirical Evaluation

I evaluated my general questionnaires for the elicitation of indirect counterstakeholders and data subjects during the presentation of the paper (Meis, 2014) at the IFIP Summer School on Privacy and Identity Management for Emerging Services and Technologies held 17-21 June, 2013 in Nijmegen. After the introduction of ProPAn and the running example, the audience of the presentation was randomly split into two groups. Both groups had 10 minutes time to identify indirect counterstakeholders and data subjects for the doctor and the financial application of the running example. One group had to guess indirect stakeholders without assistance and the other group used the developed questionnaires for the elicitation. There were 12 participants in the control group (without assistance) and 15 in the questionnaire group.

I consider 20 indirect counterstakeholders and data subjects from the overall amount of 30 indirect counterstakeholders and data subjects identified by the participants of the experiment as relevant. The 20 relevant indirect counterstakeholders and data subjects are listed in the first column of Table 9.12. The following columns contain three numbers and show how often the indirect stakeholder of the row was identified as Data Subject (DS) or counterstakeholder (CS) for the doctor and the financial application (Fin. App.). The first number in these columns

Table 9.12.: *Summarized results of the evaluation (privacy-relevant cells are printed in* **bold** *font)*

Indirect Stakeholder	**Doctor**						**Fin. App.**						**Sum**		
	DS			**CS**			**DS**			**CS**					
	C	Q	T	C	Q	T	C	Q	T	C	Q	T	C	Q	T
Insurance companies	**0**	**4**	**4**	**8**	**9**	17	0	1	1	**1**	**6**	**7**	9	20	29
Patients	**5**	**8**	13	**1**	**6**	**7**	**0**	**5**	**5**	0	0	0	6	19	25
Other doctors	**4**	**5**	**9**	**5**	**6**	11	0	0	0	0	2	2	9	13	22
Nurses and staff	**3**	**3**	**6**	**5**	**6**	11	0	0	0	0	0	0	8	9	17
Pharmacy companies	**2**	**2**	**4**	**3**	**6**	**9**	0	1	1	**1**	**1**	**2**	6	10	16
Government and politicians	0	0	0	**5**	**2**	**7**	0	0	0	**1**	**6**	**7**	6	8	14
Family of patients	**4**	**2**	**6**	**0**	**4**	**4**	0	0	0	0	0	0	4	6	10
Hacker	0	0	0	0	0	0	0	0	0	**0**	**7**	**7**	0	7	7
Law enforcement agencies	0	1	1	**0**	**2**	**2**	0	0	0	**1**	**3**	**4**	1	6	7
Financial companies	0	2	2	**2**	**1**	**3**	**0**	**0**	**0**	**0**	**0**	**0**	2	3	5
Provider of financial app	0	0	0	0	0	0	**1**	**0**	**1**	**3**	**1**	**4**	4	1	5
Friends and family of doctor	0	0	0	**0**	**4**	**4**	0	0	0	0	0	0	0	4	4
Journalist	0	0	0	**1**	**2**	**3**	0	0	0	**0**	**1**	**1**	1	3	4
Researchers	0	1	1	**1**	**2**	**3**	0	0	0	0	0	0	1	3	4
Customers of financial app	0	0	0	0	0	0	**0**	**2**	**2**	**1**	**0**	**1**	1	2	3
Doctor	0	0	0	0	0	0	**0**	**2**	**2**	**0**	**1**	**1**	0	3	3
Employees of financial app	0	0	0	0	0	0	**0**	**3**	**3**	**0**	**0**	**0**	0	3	3
Social engineering attacker	0	0	0	**0**	**3**	**3**	0	0	0	0	0	0	0	3	3
Competitor of financial app	0	0	0	0	0	0	0	0	0	**0**	**2**	**2**	0	2	2
Employers	0	0	0	**1**	**1**	**2**	0	0	0	**0**	**0**	**0**	1	1	2

Table 9.13.: *Precision, specificity, accuracy, and recall of both participant groups*

Group	**Precision**	**Specificity**	**Accuracy**	**Recall**
control group	93,39%	36,28%	41,67%	12,82%
questionnaire group	90,65%	37,65%	44,97%	20,17%

indicates how often the indirect stakeholder was considered by the control group (C), the second how often by the questionnaire group (Q), and the third gives the total amount of considerations (T). I printed a cell of the table in bold font if I consider the indirect stakeholder of the row as a relevant data subject (DS) or counterstakeholder (CS) for the domain given by the column.

Based on the relationships that I consider as relevant (bold cells in Table 9.12) and the indirect stakeholders identified by the participants, I computed the average precision, specificity, accuracy, and recall of both groups shown in Table 9.13. For the computation, I counted for each participant the numbers of

1. correctly identified indirect stakeholders (true positives),
2. correctly not identified indirect stakeholders (true negatives),
3. incorrectly identified indirect stakeholders (false positives), and
4. incorrectly not identified indirect stakeholders (false negatives).

The precision is the percentage of the correctly identified indirect stakeholders in the set of identified indirect stakeholders, while the specificity is the percentage of correctly not identified indirect stakeholders in the set of not identified indirect stakeholders. A high precision means that identified indirect stakeholders are relevant for the privacy analysis, and a high specificity means that not identified indirect stakeholders are not relevant for the privacy analysis. The accuracy is defined as the percentage of the sum of the correct identified and not identified indirect stakeholders in the set of all possible results. Hence, a high accuracy means that the number of incorrectly (not) identified stakeholders is low. The recall is the percentage of the correctly identified indirect stakeholders in the set of all correct indirect stakeholders and hence, a high recall means that the number of correctly identified indirect stakeholders is close to the set of all correct indirect stakeholders.

The precision of both groups is above 90%. The questionnaire group identified a few more unexpected indirect (counter)stakeholder relationships (false positives) than the control group, which leads to a smaller precision. The questionnaire group has a slightly larger accuracy and specificity than the control group. The recall of both groups is below 20%, which is surely caused by the limited time of 10 minutes that the participants had for the elicitation. Nevertheless, the questionnaire group identified 1,5 times more correct indirect counterstakeholder and data subject relationships. In summary, the questionnaires seem to help to increase the number of correctly identified indirect counterstakeholders and data subjects and their relationships to the domains of the problem frame model significantly. The trade-off of the questionnaires is that the precision is slightly decreased. But this is reasonable because my main focus is the elicitation of all relevant indirect counterstakeholders and data subjects for a privacy analysis that is as complete as possible.

9.7. Tool Support

The UDEPF tool provides support to create the needed domain knowledge diagrams. I tested wizards based on the elicitation questionnaires and modeling patterns in earlier versions of the ProPAn tool. However, the wizards did not significantly reduce the effort to create the domain knowledge diagrams in comparison to the manual creation of these using the UDEPF tool. Hence, the creation and validation capabilities of the UDEPF tool sufficiently support the first step of the ProPAn method.

9.8. Comparison to the State of the Art

In this section, I compare my privacy context elicitation to the state of the art in stakeholder analysis (see Section 9.8.1) and privacy requirements engineering (see Section 9.8.2).

9.8.1. Stakeholder Analysis

Stakeholder analysis originates from information systems research (Pouloudi, 1999). I describe in the following the research of requirements engineers on this field.

Sharp et al. (1999) present a method for the identification of stakeholders for requirements engineering. The authors distinguish four groups of *baseline stakeholders*, namely users, developers, legislators, and decision-makers. For each baseline role, the method identifies *supplier stakeholders* who provide information, *client stakeholders* who process or inspect the products, and *satellite stakeholders* who interact or support the baseline stakeholders and vice versa.

Alexander and Robertson (2004) recommend a combination of two methods. The first method is the onion model (Alexander, 2005) for the identification of stakeholders. The model arranges different generic stakeholder roles around the product, which is the center of the onion. The distance of a stakeholder to the product expresses how directly the stakeholder interacts with the product. The second method is to use the Volere stakeholder analysis template (Robertson and Robertson, 2006). This template suggests 72 stakeholder roles that are divided into 14 stakeholder classes which again are divided into 4 categories of stakeholder classes. The template shall support the elicitation of stakeholders that hold relevant knowledge for the project.

The stakeholder analysis approaches all aim at the identification of stakeholders that are relevant to successfully complete a project. In contrast, I focus only on those stakeholders whose privacy is affected or those counterstakeholders that can harm the privacy of data subjects in the system-to-be and not at the time of development. However, I considered the mentioned approaches to derive the general indirect environment elicitation questionnaires, as discussed in Section 9.2.

9.8.2. Privacy Requirements Engineering

Most privacy requirements engineering methods do not consider the elicitation and documentation of the system's indirect environment. In the following, I discuss briefly the state of the art methods that I introduced in Chapter 3 and that explicitly consider the identification of stakeholders.

Liu et al. (2003) consider actors of the system as insider attackers. Hence, they consider already identified actors as counterstakeholders, but they do not propose do elicit further actors. Crespo et al. (2015) state that all relevant stakeholders (including data subjects and counterstakeholders) have to be identified for a privacy analysis. However, they do not provide guidance on how to elicit and document these in a systematic way. Drgon et al. (2016) derive the relevant stakeholders from use cases, but they do not further elicit the indirect environment of these. This is, because the indirect environment is not in the scope of the use cases. De and Le Métayer (2016) provide a template to document relevant stakeholders and the relations between them. However, they do not provide guidance or a method to elicit the relevant stakeholders.

9.9. Conclusions

I introduced the first step of the ProPAn method called Privacy Context Elicitation, in this chapter. The four substeps of this step are considered with the elicitation and modeling of the indirect environment of the system-to-be and connection domains refining interfaces between domains in the system-to-be. The elicitation steps are supported by templates and questionnaires, and the modeling steps with patterns that describe how the elicited domain knowledge can be added as domain knowledge diagrams to the given problem frame model (cf. Figure 9.1). I have introduced general elicitation questionnaires and I have shown that application-domain-specific elicitation questionnaires can created to support the elicitation process using the domain of cloud computing. The proposed method to introduce connection domains and the corresponding domain knowledge pattern is not specific to privacy analyses and can in general be used to modularly add connection domains to a problem frame model.

All substeps of the method are illustrated using the EHS example and the modeling of the domain knowledge diagrams is supported by the UDEPF tool. A comparison with the state of the art methods in privacy requirements engineering has shown that no other method provides such a systematic and supported elicitation and modeling method as the ProPAn method does. Additionally, I have presented the results of an empirical evaluation of the general elicitation questionnaire for the indirect environment of the system-to-be. This evaluation has shown that the questionnaires help to identify indirect counterstakeholders and data subjects by increasing the recall by factor 1,5 (cf. Table 9.13).

Having identified the privacy-relevant indirect environment and connection domains for our system-to-be, the analysis team can proceed with the step Privacy Threshold Analysis, which I introduce in the following section.

Chapter 10

Privacy Threshold Analysis

During the second step of the ProPAn method, a Privacy Threshold Analysis (PTA) is performed. In this chapter, I provide the details on this step. The goal of a PTA is to determine the need of a detailed privacy analysis of a system (often called Privacy Impact Assessment (PIA)). For a PTA, the personal data that are processed by the system have to be identified and evaluated. Based on the identified personal data, the analysis team has to decide whether the system is privacy-sensitive and hence, needs to be analyzed in depth.

The identification process for personal data is partly based on the paper (Meis and Heisel, 2015) and its extended version (Meis and Heisel, 2016a). I am the main author of both papers and Maritta Heisel provided valuable feedback that helped to improve the papers.

I provide an introduction to the topic of privacy threshold analysis and the contributions of this chapter in Section 10.1. Sections 10.2, 10.3, 10.4, 10.5 present the substeps Identify data subjects and personal data, Model personal data relations, Generate initial data flow graphs, and Decide on privacy threshold, respectively. I investigate whether the state of the art privacy requirements engineering methods have similar steps to ProPAn's privacy threshold analysis in Section 10.6. This chapter is concluded in Section 10.7.

10.1. Introduction

Prior and as part of a detailed privacy assessment, it is recommended to first perform a privacy threshold analysis (PTA) to assess whether a detailed privacy analysis needs to be performed. For example, the National Institute of Standards and Technology (NIST) (McCallister and Scarfone, 2010) recommends to perform a PTA to identify the personal data processed by the system-to-be and to determine whether a privacy impact assessment shall be performed or not.

There exist several templates that support to conduct of a PTA that are proposed by US agencies, e.g., the FedRAMP Privacy Threshold Analysis and Privacy Impact Assessment Template[1], the USAID Privacy Threshold Analysis Template[2], the Homeland Security PTA[3], the FBI PTA[4], the Privacy Threshold Analysis (PTA) to Identify Systems Containing Personally Identifiable Information (PII)[5], and the Privacy Impact Assessment (PIA) Guide[6]. All templates have in common that they ask whether personal data are processed by the considered

[1] `https://www.fedramp.gov/assets/resources/templates/SSP-A04-FedRAMP-PIA-Template.docx` accessed on 26 April 2018

[2] `https://www.usaid.gov/sites/default/files/documents/1868/508maj.pdf` access on 26 April 2018

[3] `https://www.dhs.gov/sites/default/files/publications/privacy-dhs-pta-template-20140123.pdf` access on 26 April 2018

[4] `https://epic.org/foia/FBI-PTA-Template.pdf` access on 26 April 2018

[5] `https://www.epa.gov/sites/production/files/2017-01/documents/pta-form-2015.pdf` access on 26 April 2018

[6] `https://www.opm.gov/information-management/privacy-policy/privacy-references/piaguide.pdf` access on 26 April 2018

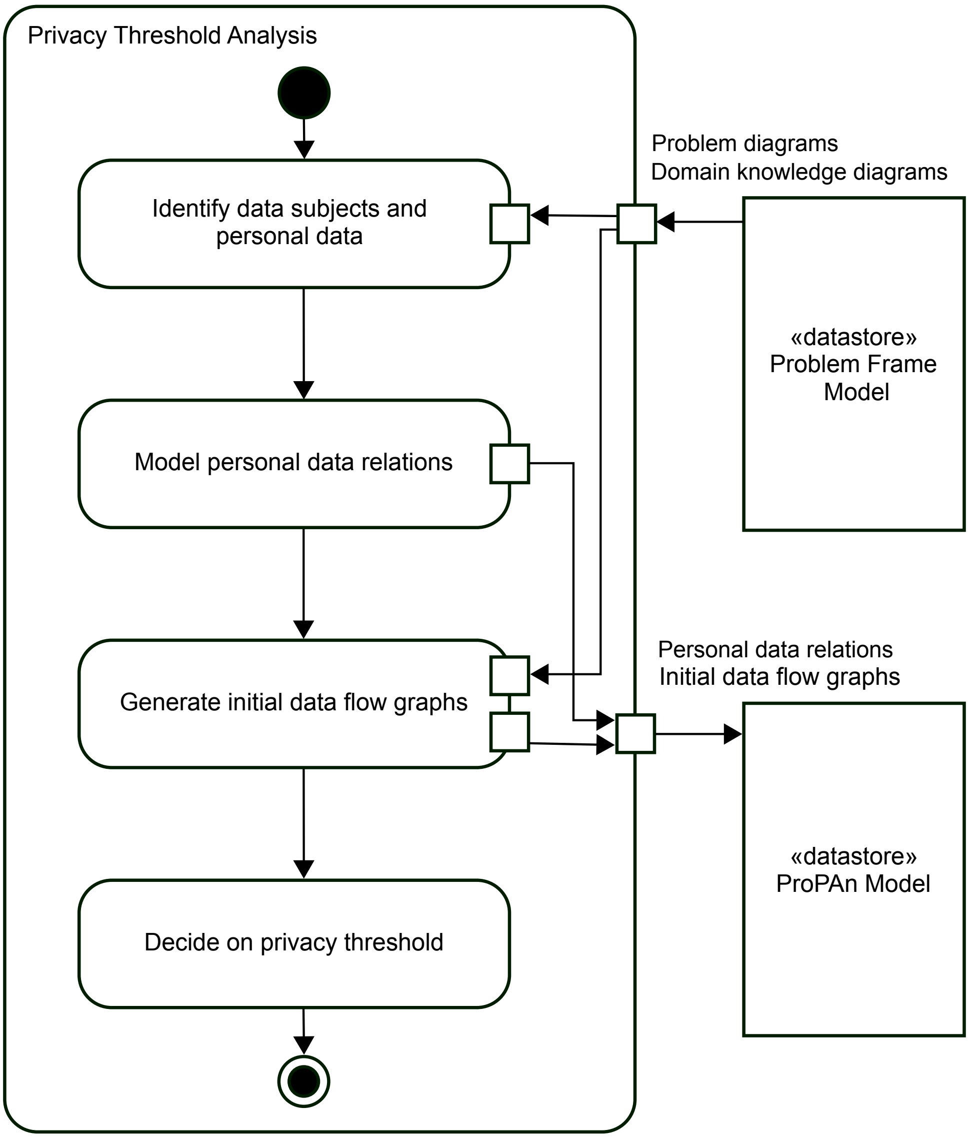

Privacy Threshold Analysis
Identify data subjects and personal data
Model personal data relations
Generate initial data flow graphs
Decide on privacy threshold
Problem diagrams
Domain knowledge diagrams
«datastore» Problem Frame Model
Personal data relations
Initial data flow graphs
«datastore» ProPAn Model

Figure 10.1.: *Detailed view on the step* Privacy Threshold Analysis *of the ProPAn method*

system, whose personal data these are, whether these data are identifying the data subjects. The PTA templates also provide guidance on the circumstances when a further privacy analysis of the system is necessary. In most cases, a further privacy analysis is necessary if personal data are processed that allow to identify the related data subjects.

However, the PTA templates do not support a privacy analysis team to determine whether, what, and whose personal are processed by the system-to-be. Additionally, the filled out PTA is not linked to artifacts describing the system-to-be, i.e., the system model or the requirements specification.

To address these shortcomings and to support the analysis team to perform a privacy threshold analysis (including the identification of personal data processed by the system-to-be), I propose a systematic and tool-supported method to derive the information needed from a given problem frame model (enhanced with privacy-relevant domain knowledge). The identified personal data and their relations to the data subjects, and further information, such as, how the personal data are collected and due to which statements the data are collected, are persisted in a ProPAn model. The ProPAn model extends the problem frame model of the system-to-be and allows to document privacy-related information.

Figure 10.1 shows the refinement of the step Privacy Threshold Analysis of the ProPAn method. The first substep is to Identify data subjects and personal data. This substep uses as input the problem and domain knowledge diagrams provided by the Problem Frame Model of the system-to-be. After having identified the data subjects and personal data, the personal data and their relations to the data subjects are modeled in the substep Model personal data relations. The personal data and their relations and properties are documented in the ProPAn Model for the system-to-be. This model references the elements of the problem frame model and extends this model with privacy-related information without modifying the problem frame model. Having modeled the personal data relations, initial data flow graphs that over-approximate the flow of personal data inside the system-to-be (and its indirect environment) are automatically generated in the step Generate initial data flow graphs. These initial data flow graphs are also stored in the ProPAn Model. Finally, the analysis team has to decide whether a detailed privacy analysis has to be performed in the substep Decide on privacy threshold. In this substep, the decision for the decision point after the step Privacy Threshold Analysis in Figure 8.1 is made. The identified personal data relations and the initial data flow graphs are used to support the analysis team to make a decision.

I provide the details on the four steps, how these are supported by the ProPAn tool, and an exemplary application of these steps on the electronic health system (EHS) scenario (see Chapter 4) in the following four sections.

10.2. Identify Data Subjects and Personal Data

The goal of the first substep of the Privacy Threshold Analysis is to Identify data subjects and personal data. For this, the problem diagrams and domain knowledge diagrams of the Problem Frame Model are used as inputs (cf. Figure 10.1).

10.2.1. Procedure for the Identification of Personal Data

Each biddable domain in the problem frame model can potentially be a data subject. Hence, the analysis team has to check for each biddable domain, whether personal data of this domain are processed by the system-to-be. Candidates for personal data are all phenomena controlled by the biddable domain, and these are processed if they are referenced by a statement contained in a problem or domain knowledge diagram. The set of all phenomena of a domain that a statement refers to can be derived from a problem frame model using the attribute referedToPhenomena defined in Listing 10.1 (for the EMF metamodel see Figure 2.10 on page 22).

I distinguish two cases for the identification of personal data from the referred-to phenomena. First, a phenomenon can be a causal phenomenon and second, it can be a symbolic phenomenon. Causal phenomena represent events or commands a domain issues, and symbolic phenomena represent a state, value, or information. If the phenomenon is symbolic, then the analysis team

Listing 10.1: *Additional attributes for a domain that represent the phenomena that are referenced by a statement*

```
context Domain
def: referredToPhenomena : Set(Phenomenon) =
 self.statementreferences→select(sr | sr.oclIsTypeOf(RefersToReference)).contains
def: personalDataCandidates : Set(Phenomenon) =
 self.referredToPhenomena→select(p|p.oclIsTypeOf(SymbolicPhenomenon))
def: personalDataContainers : Set(Phenomenon) =
 self.referredToPhenomena→select(p|p.oclIsTypeOf(CausalPhenomenon))
```

has to check whether this phenomenon represents personal data. The set of all these direct candidates for personal data can be retrieved from the problem frame model using the attribute personalDataCandidates defined in Listing 10.1. If the phenomenon is causal, then the analysis team has to check whether it contains or transmits personal data. These phenomena that possibly contain or transmit personal data can be queried from the problem frame model using the attribute personalDataContainers defined in Listing 10.1.

Having selected a biddable domain, the analysis team has to answer the questions listed in Table 10.1 for each causal phenomenon. By answering the first question, symbolic phenomena are identified that are transmitted by the causal phenomenon, e.g., as parameter. The second question aims at identifying symbolic phenomena that correspond to metadata provided by the causal phenomenon or documented by the observer of it during its occurrence, e.g., the IP address of the issuer and the date of occurrence. By answering the two questions, the analysis team identifies additional symbolic phenomena that are candidates for personal data of the corresponding biddable domain. Note that the additionally elicited symbolic phenomena are not added to the problem frame model. Instead, data objects are created for them in the ProPAn model during the substep Model personal data relations.

Table 10.1.: *Causal phenomena evaluation questions*

No.	Question	Answer type
CP1	Does the causal phenomenon contain or transmit symbolic phenomena?	Set(Phenomena)
CP2	Does the causal phenomenon contain or transmit metadata related to its occurrence (including logging of the causal phenomenon at the observer)?	Set(Phenomena)

Table 10.2.: *Symbolic phenomena evaluation questions*

No.	Question	Answer type
SP1	Is the symbolic phenomenon related to the biddable domain?	Boolean
SP2	Does the symbolic phenomenon represent sensitive personal data of the biddable domain?	Boolean
SP3	To which degree is the symbolic phenomenon itself linkable to the biddable domain?	Linkability
SP4	How is the symbolic phenomenon collected from the biddable domain due to the statements referring to it (or the phenomenon containing it)?	Set(CollectionMethod)

For each symbolic phenomenon contained or transmitted by a causal phenomenon, and each symbolic phenomenon controlled by the selected biddable domain and referred to by a statement, the questions listed in Table 10.2 have to be answered. First, the analysis team has to decide whether the symbolic phenomenon represents data related to the biddable domain and hence, qualifies to be personal data. If this is not the case, the analysis team does not have to answer the following questions for the symbolic phenomenon. Then, it is elicited whether the data represent sensitive personal data. To decide on this, the analysis team may use catalogs or definitions of sensitive personal data as provided by ISO/IEC (2011); European Commission (2016). The third question elicits to which degree the personal data themselves are linkable to the data subject (biddable domain). For example, the personal data could be an identifier, such as name and address, or bank account number, or they could be data not directly linkable to an individual, such as last name, city, or balance. The analysis team shall document the linkability using the enumeration Linkability (see Figure 10.2), which I also use in my privacy requirements taxonomy (cf. Chapter 7). The analysis team may also use lists of identifying personal data as provided by (ISO/IEC, 2011; European Commission, 2016) to determine the degree of linkability of the identified personal data. The last question asks how the personal data are collected from the data subject. To decide on this, the analysis team shall consider the statements that reference the symbolic phenomenon or the causal phenomenon from which the symbolic phenomenon was identified. These statements can automatically be derived from the problem frame model. How the personal data are collected is documented by a set of collection methods using the enumeration CollectionMethod (see Figure 10.2). Possible collection methods are:

direct collection from the data subject. That is, the data subject actively provides the data.

indirect collection from the data subject. That is, the data subject does not actively provide the data, but they are collected, e.g., by observing the data subject. The data subject is not necessarily aware of this collection of data.

reused personal data are processed. That is, the personal data were already collected for another purpose from the data subject and are now reused for the purposes of the system-to-be.

external collection of personal data. That is, the collection of the personal data from the data subject is not in the scope of the machine. The collection is performed by an external entity.

derived personal data are processed. That is, the personal data are not collected from the data subject, but are derived from other data available to the system-to-be.

After identifying the personal data of all biddable domains in the problem frame model, the analysis team can proceed with the modeling of the personal data and their elicited properties. This is done in the substep Model personal data relations (see Section 10.3).

10.2.2. Application to EHS

In the remainder of my thesis, I consider the seven functional requirements introduced in Chapter 4 and the domain knowledge identified in Chapter 9, excluding the cloud-specific domain knowledge. Additional domain knowledge can be identified for the EHS, but I do not consider this domain knowledge to keep the EHS example simple.

The biddable domains in the problem frame model are Patient, Doctor, Researcher, Family, Financial Employee, Financial Customer, Tax Authority, and Hacker. I only present the application of this step for the domains Patient and Doctor, because these are the most privacy-relevant biddable domains in the EHS.

Table 10.3.: *Personal data identified for the data subject* Patient

Phenomenon	Statements	SP1	SP2	SP3	SP4
vitalSigns	A1, F1, R6	yes	yes	mediumGroup	direct, indirect, reused
healthStatus	A1, F1	yes	yes	mediumGroup	indirect, reused
patientDetails	A1, F1	yes	yes	single	indirect, reused

The Patient does not control causal phenomena, hence the questions from Table 10.2 do not have to be answered. Table 10.3 shows the symbolic phenomena of patients, which statements refer to these, and the answers to the questions listed in Table 10.1. Note that the first two columns can automatically be derived from the problem frame model. The phenomena vitalSigns, healthStatus, and patientDetails are all considered as sensitive personal data, and are collected directly (R6), indirectly (A1), and by reusing already collected data from patients (F1). The patientDetails are considered to identify the individual they are related to, because the patient details shall also include the patient's name and contact address. The vitalSigns and healthStatus only allow to narrow down the set of individuals they may belong to to a medium-sized group. Note that the analysis team has to define the meaning of small, medium and large groups, and the linkability literal anonymous. For example, a small group of patients could be defined to consist of 2-10 patients, a medium group of 11-100, and a large group of 101-1000, and if the group consists of more than 1000 patients, it could be considered as anonymous.

From the doctor, only causal phenomena are referred to by the statements in the EHS problem frame model. Table 10.4 shows the symbolic phenomena identified to be contained or transmitted by the causal phenomena. The causal phenomena modifyEHR and createEHR transmit the details of the doctor, treatments he or she gave, and notes he or she added to the patients' health records. Additionally, it is planned to log the device used by the doctor used for interactions with the EHS and the date of these interactions. The phenomenon initiateAccounting contains an identifier to document the doctor who initiated the accounting process. With the phenomenon browseEHR, doctors provide the search query they entered, and the accessed EHRs shall also be registered. The communication of the doctor with the patient's family is represented by the phenomenon tellAboutHealthStatus. During such communication the doctor may tell the family about his or her details (including name and contact information), and treatments he or she gave to the patient. Additionally, the date of this communication is known to the patient's family.

Table 10.4.: *Data identified from the causal phenomena of the domain* Doctor

Phenomenon	Statements	CP1	CP2
modifyEHR	R1	doctorDetails, treatments, notes	usedDeviceAndDate
createEHR	R1	doctorDetails, treatments, notes	usedDeviceAndDate
initiateAccounting	R3	doctorId	usedDeviceAndDate
browseEHR	R2	enteredSearchQuery, accessedEHRs	usedDeviceAndDate
tellAboutHealthStatus	A2	doctorDetails, treatments	interactionDate

Table 10.5.: *Personal data identified for the data subject* Doctor

Phenomenon	Statements	SP1	SP2	SP3	SP4
doctorDetails	R1, A2	yes	yes	single	direct
treatments	R1, A2	yes	yes	smallGroup	direct
notes	R1	yes	yes	smallGroup	direct
enteredSearchQuery	R2	yes	no	smallGroup	direct
accessedEHRs	R2	yes	no	smallGroup	direct
doctorId	R3	yes	no	single	direct
usedDeviceAndDate	R1, R2, R3	yes	no	single	indirect
interactionDate	A2	yes	no	mediumGroup	indirect

The identified symbolic phenomena contained or transmitted by the doctor's causal phenomena are further assessed in Table 10.5. The statements in the second column are derived from the causal phenomena which contain or transmit the symbolic phenomena. doctorDetails are considered as sensitive personal data that allow to identify the individual doctor they belong to. The phenomena treatments and notes are both sensitive personal data of the doctor that may be linked to a small group of doctors they belong to. The enteredSearchQuery and accessedEHRs are not considered as sensitive. However, they may also be linked to a small set of possible related individuals. The doctorId communicated during the accounting can be linked to the individual doctor it belongs to. All these symbolic phenomena are collected directly from the doctors. In contrast, the metadata usedDeviceAndDate and interactionDate are indirectly collected. While the usedDeviceAndDate may allow to identify the individual doctor, especially if personalized devices shall be used, the interactionDate is considered to be only linkable to a medium-sized group of possible doctors.

10.3. Model Personal Data Relations

In this step, the previously elicited information about the personal data that are processed by the system-to-be, are added to the ProPAn Model.

10.3.1. Metamodel for the Documentation of Personal Data Relations

Figure 10.2 shows the part of the ProPAn metamodel that allows to document the previously elicited information. The class Contains is used to document that a causal phenomenon contains or transmits a symbolic phenomenon. A contains object references two phenomenon objects, namely the containing phenomenon and the contained phenomenon. A phenomenon can be referenced by arbitrary many contains objects. Additionally, the class Contains inherits from the abstract class TraceableElement a reference to a collection of statements. This collection is used to document from which statements of the problem frame model the contains relationship between the phenomena was identified. The contains relation represented by the class Contains is irreflexive and transitive. Note that the ProPAn tool computes the transitive closure of the contains relation when this is necessary. Hence, it is not necessary to add instances of the class Contains for all relations in the transitive closure. Listing 10.2 shows the OCL invariant

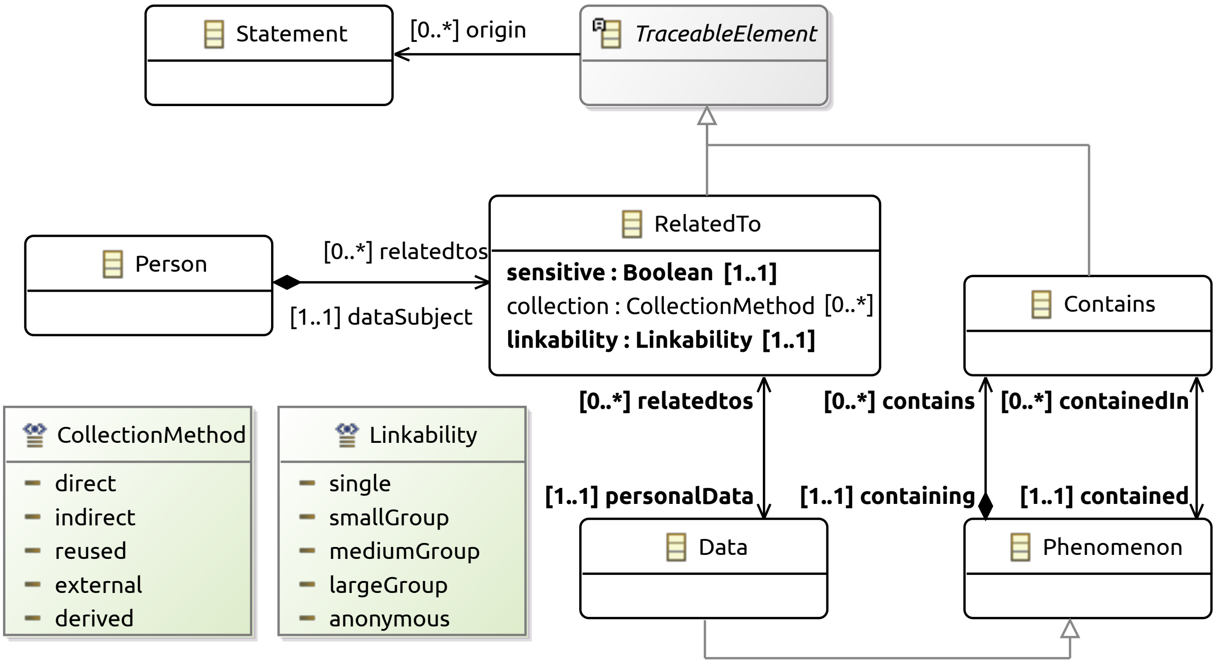


Figure 10.2.: *The class RelatedTo of the ProPAn model*

Listing 10.2: *OCL invariant specifying that the contains relation is irreflexive*

```
context Phenomenon
def: containsClosure : Set(Phenomenon) =
  self.contains.contained→closure(p|p.contains.contained)
inv: self.containsClosure→excludes(self)
```

Listing 10.3: *Extension to the text template defining the meaning of an instance of the class* RelatedTo

```
The system-to-be collects personalData due to origin.
<foreach collection="collection" iterator="cm">
  <switch enumeration="cm">
    <case literal="CollectionMethod::direct">
      The data subject directly provides the personal data.
    </case>
    <case literal=CollectionMethod::indirect">
      The personal data are indirectly collected from the data subject, e.g., by observation.
    </case>
    <case literal="CollectionMethod::reused">
      The personal data were already collected for another purpose and is now reused.
    </case>
    <case literal=CollectionMethod::external">
      An external entity collects the personal data from the data subject and provides them to the system-to-be.
    </case>
    <case literal=CollectionMethod::derived">
      The personal data are not collected from the data subject, but derived from other personal data available to the system-to-be.
    </case>
  </switch>
</foreach>
```

that specifies that the contains relation is irreflexive. The OCL operation *closure* computes the reflexive and transitive closure for a starting set by iteratively applying the provided expression to the elements of the set and adding the new elements to the starting set until all elements of the set have been considered once (see also (Object Management Group, 2014)).

I already introduced the class RelatedTo in Figure 7.1 on page 99 and provided a text template defining the meaning of instances of the class RelatedTo in Listing 7.4 on page 100. For presentation purposes, I omitted in Chapter 7 the attribute collection and the relation to the statements from which the relation was identified. The latter is inherited from the abstract class *TraceableElement*. The text template shown in Listing 10.3 extends the template shown in Listing 7.4. That is, the text template shown in Listing 10.3 may be appended to the text template shown in Listing 7.4. For an introduction to the text template language, I refer to Section 7.1 on page 97. The class RelatedTo allows to model that a data object is related to a person object, i.e., the data object represents personal data of the person object that consequently represents a data subject. Additionally, the class provides attributes to document the answers to the questions listed in Table 10.2.

As already mentioned in Chapter 8, I designed the ProPAn metamodel such that it can reference elements of the UDEPF metamodel, without introducing dependencies from the UDEPF metamodel to the ProPAn metamodel. Figure 10.3 shows how this is realized. The ProPAnModel has a reference to the ProblemFrameModel that it extends. The non-abstract classes Domain, Phenomenon, and Statement contained in the ProPAnModel are wrappers for the ab-

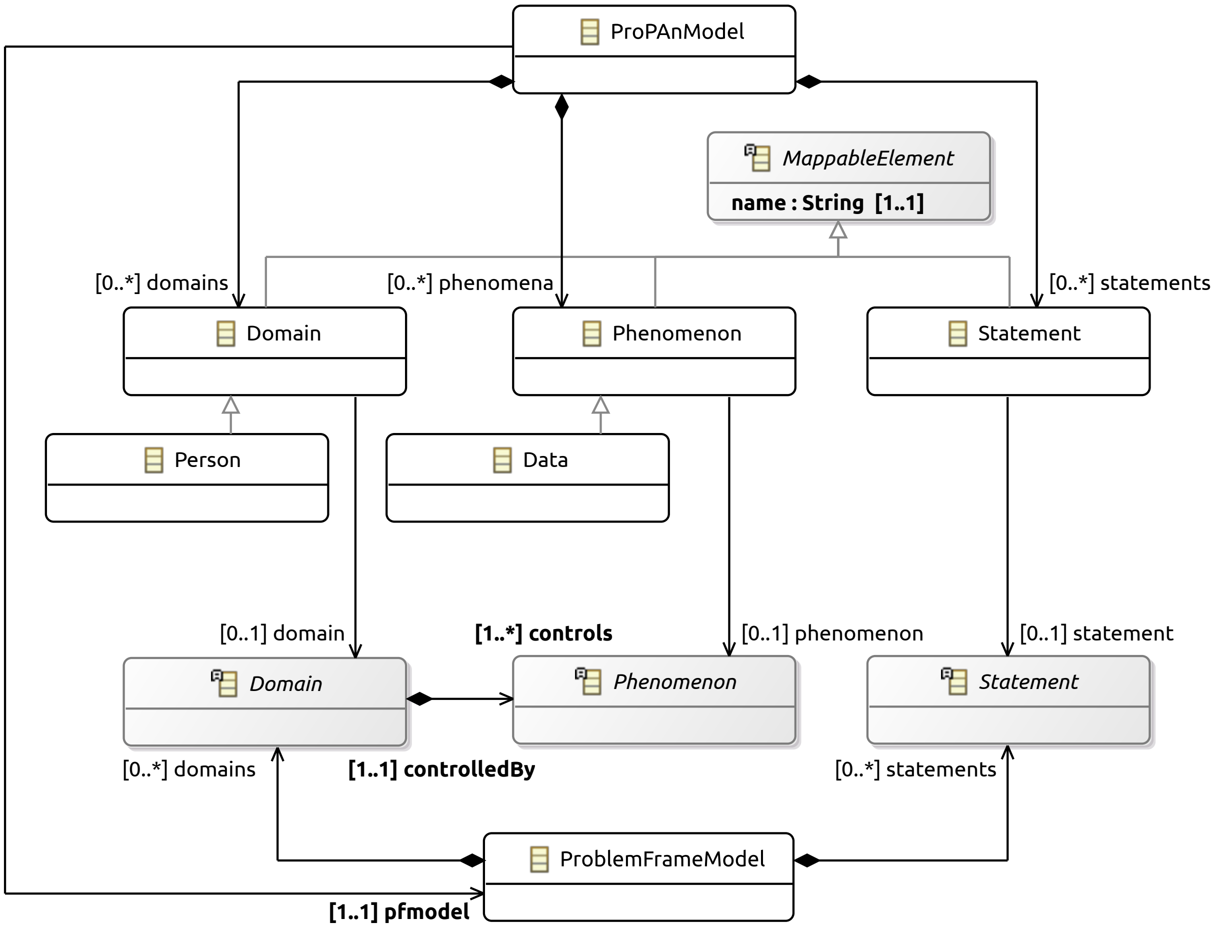


Figure 10.3.: *Relations of the ProPAn model to the problem frame model*

Listing 10.4: *OCL invariant for the class* Person

```
context Person
inv: self.domain.oclIsTypeOf(BiddableDomain)
```

stract classes *Domain*, *Phenomenon*, and *Statement* contained in the ProblemFrameModel. These wrapper classes have a reference to the classes of the ProblemFrameModel they represent in the ProPAnModel. This modeling allows to extend the abstract classes *Domain*, *Phenomenon*, and *Statement* from the UDEPF metamodel with attributes and relations that are relevant for a privacy analysis, without modifying the UDEPF metamodel. Figure 10.3 also shows that I introduce the classes Person and Data as refinements of the classes Domain and Phenomenon, respectively. The class Person represents BiddableDomains, and the class Data wraps the class SymbolicPhenomenon. Hence, I obtain the validation conditions shown in Listings 10.4 and 10.5. When a ProPAn model is created, then it is automatically filled with instances of the classes Domain, Person, Phenomenon, Data, and Statement based on the related problem frame model. For each biddable domain, other domain, symbolic phenomenon, and causal phenomenon in the problem frame model, exactly one instance of the class Person, Domain, Data, and Phenomenon is created that references it, respectively. This is formalized in the OCL constraint given in Listing 10.6 and the OCL constraints specified in Listings 10.4 and 10.5.

Note that the related-to relation represented by instances of the class RelatedTo allows to specify that data are personal data of different persons. However, for each person there shall

Listing 10.5: *OCL invariant for the class Data*

```
context Data
inv: self.phenomenon.oclIsTypeOf(SymbolicPhenomenon)
```

Listing 10.6: *OCL invariant for the class ProPAnModel*

```
context ProPAnModel
inv: self.pfmodel.domains→forAll(pf_d|
        self.domains→one(propan_d|propan_d.domain = pf_d))
inv: self.pfmodel.phenomena→forAll(pf_p|
        self.phenomena→one(propan_p|propan_p.phenomenon = pf_p))
inv: self.pfmodel.statements→forAll(pf_s|
        self.statements→one(propan_s|propan_s.statement = pf_p))
```

Listing 10.7: *OCL invariant guaranteeing that at most one related-to relation is instantiated per pair of person and data*

```
context Person
inv: self.relatedtos→isUnique(personalData)
```

be at most one related-to relation for a data object. This is formalized by the OCL constraint listed in Listing 10.7.

10.3.2. Procedure to Model the Identified Data Subjects and Personal Data

For each person in the ProPAn model, a *personal data diagram* is created using the ProPAn tool. A personal data diagram is a view on the ProPAn model and does not have a class representing it in the ProPAn metamodel. It contains all instances of the class RelatedTo that reference the person as data subject. These instances are represented as white rectangles with the name of the personal data in the first line, and the values of the attributes sensitive, linkability, collectionMethod, and origin in the following lines. A personal data diagram also contains all phenomena of the ProPAn model that reference a causal phenomenon that is controlled by the biddable domain that the person represents. These phenomena are represented using gray rectangles with the name of the phenomenon in the first line, and the list of statements that refer to it in the second line. Additionally, a personal data diagram contains an aggregation link for each instance of the class Contains if the containing and the contained phenomena are contained in the personal data diagram (either as phenomenon or as personal data of a RelatedTo instance). The aggregation link is annotated with the statements from which the contains relation was identified (attribute origin). Figure 10.4 shows an example of a personal data diagram. It shows that the Causal phenomenon contains or transmits the Personal data from causal phenomenon due to statement Statement 1. It also contains the Personal data from symbolic phenomenon. For both personal data objects, the attributes of the corresponding related-to instances are shown. The attribute data subject is implicitly given by the person for which the personal data diagram is created.

For each symbolic phenomenon that was identified to be contained in or transmitted by a causal phenomenon based on the questionnaire shown in Table 10.1, and that was identified to be personal data based on the questionnaire shown in Table 10.1, an instance of the class Data is created that represents the elicited symbolic phenomenon. Additionally, an instance of the class Contains is created that connects the phenomenon in the ProPAn model that represents the causal phenomenon to the instance of the class Data. The attribute origin of the contains

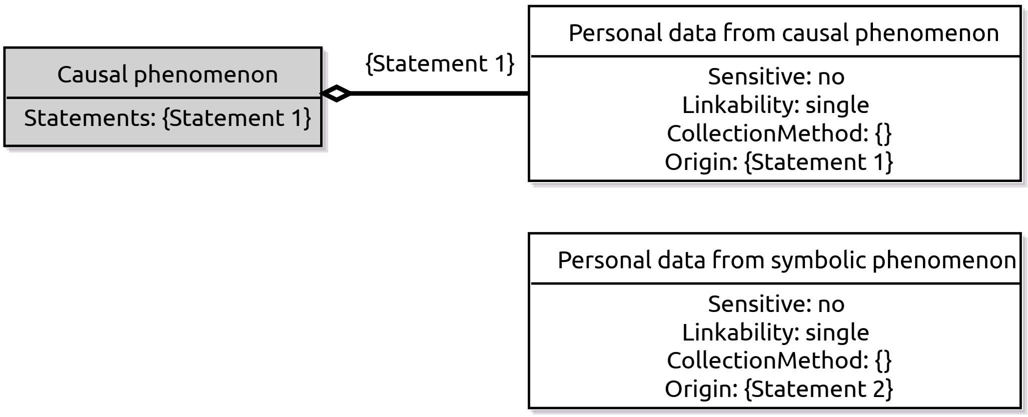


Figure 10.4.: *Example of a personal data diagram*

relation is initially set to the set of statements that refer to the causal phenomenon that contains or transmits the personal data. The analysis team may decide edit the set of statements.

For each biddable domain and each symbolic phenomenon that was identified to be personal data of the biddable domain, an instance of the class RelatedTo is instantiated. This instance references the person (dataSubject) that represents the biddable domain in the ProPAn model, and the data (personalData) that represents the symbolic phenomenon. The instance of the class RelatedTo also documents the answers to the questions of Table 10.2 concerning the sensitivity and linkability of the personal data, and how they are collected using the attributes sensitive, linkability, and collection. Additionally, the statements of the ProPAn model that represent those statements of the problem frame model that refer to the symbolic phenomenon (or to the causal phenomenon that contains or transmits the symbolic phenomenon) are connected to the RelatedTo instance (origin).

The analysis team may decide to add additional contains relations when they identify that specific personal data are contained in other personal data. For example, the personal data *postal address* could be contained in the personal data *contact information*. These containment relationships are used in the detailed analysis of the flow of personal data through the system in the step Data Flow Analysis (see Chapter 11). Note that it is not necessary to add additional contains relationships, or to further refine the identified personal data using contains relations at this point.

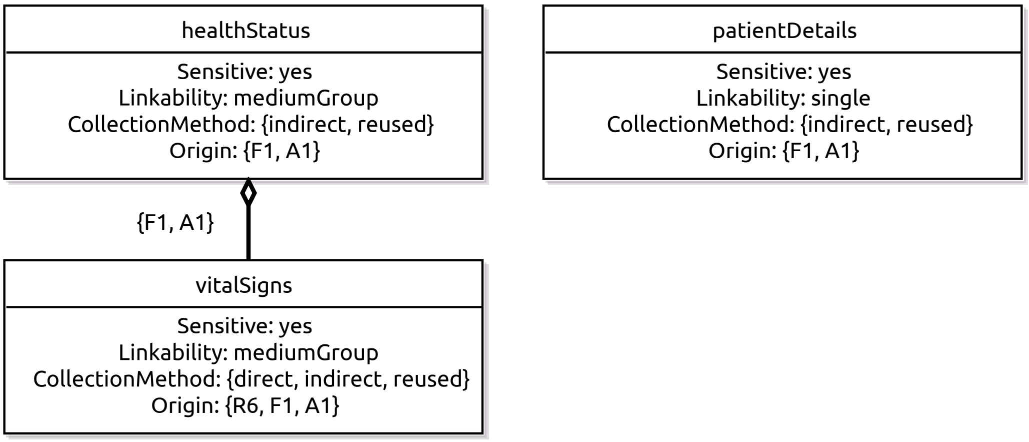


Figure 10.5.: *Personal data diagram for the data subject Patient*

10.3.3. Application to EHS

The input for the modeling is the elicited personal data for the data subjects Patient (see Table 10.3) and Doctor (see Table 10.5) and whether these are transmitted or contained by causal phenomena (Table 10.4).

Figure 10.5 shows the personal data diagram for the data subject Patient. This diagram shows the instances of the class RelatedTo that correspond to the answers for the identification of personal data based on symbolic phenomena listed in Table 10.3. Additionally, I decided to model that the vital signs are contained in the health status.

The personal data diagram for the data subject Doctor is presented in Figure 10.6. It contains all causal phenomena controlled by the domain Doctor listed in Table 10.4, all personal data identified to be transmitted or contained in the causal phenomena, and the contains relationships that document which causal phenomena contain or transmit which personal data. The properties of the personal data (more precisely of the RelatedTo instance for the data and the doctor) reflect the values elicited in Table 10.5. Additionally, I modeled that the personal data usedDeviceAndDate contain the personal data interactionDate. This is, the personal data usedDeviceAndDate represent the combination of the used device for and the date of the interaction with the EHS.

10.4. Generate Initial Data Flow Graphs

Based on the identified personal data and the statements of the problem frame model, the flow of the personal data through the system can be over-approximated. The flow of personal data is inferred from the *refers to* and *constrains* references of the statements in the problem and domain knowledge diagrams. That is, data may flow from domains referred to by a statement to the domains constrained by the same statement. I assume at this point that no other flows can occur in the system-to-be. If additional flows are intended or allowed, then a functional requirement is missing or incorrectly represented as problem diagram, or a data subject, counterstakeholder, or interface was overlooked during the step Privacy Context Elicitation (see Chapter 9).

10.4.1. Metamodel for Initial Data Flow Graphs

The metamodel for the initial data flow graph that is generated in this substep of the Privacy Threshold Analysis is shown in Figure 10.7. The initial DataFlowGraph is generated for one Person and contains DataFlowEdges. Each data flow edge references one domain as source and one as target. Note that the association allows to assign multiple domains as sources, because later the abstract class *DFGEdge* is refined by another edge type that allows multiple sources. However, this is not allowed for instances of the class DataFlowEdge. This is formalized in Listing 10.8. Additionally, each data flow edge references a collection of data that flows from the source to the target, and the collection of statements from which the flow of personal data was derived.

10.4.2. Generation of Initial Data Flow Graphs

The graph is generated starting at the domain that represents the data subject. For the data subject, all statements are collected that refer to him or her, and from these statements all

Listing 10.8: *Constrain on the class DataFlowEdge*

```
context DataFlowEdge
inv: self.source→size() = 1
```

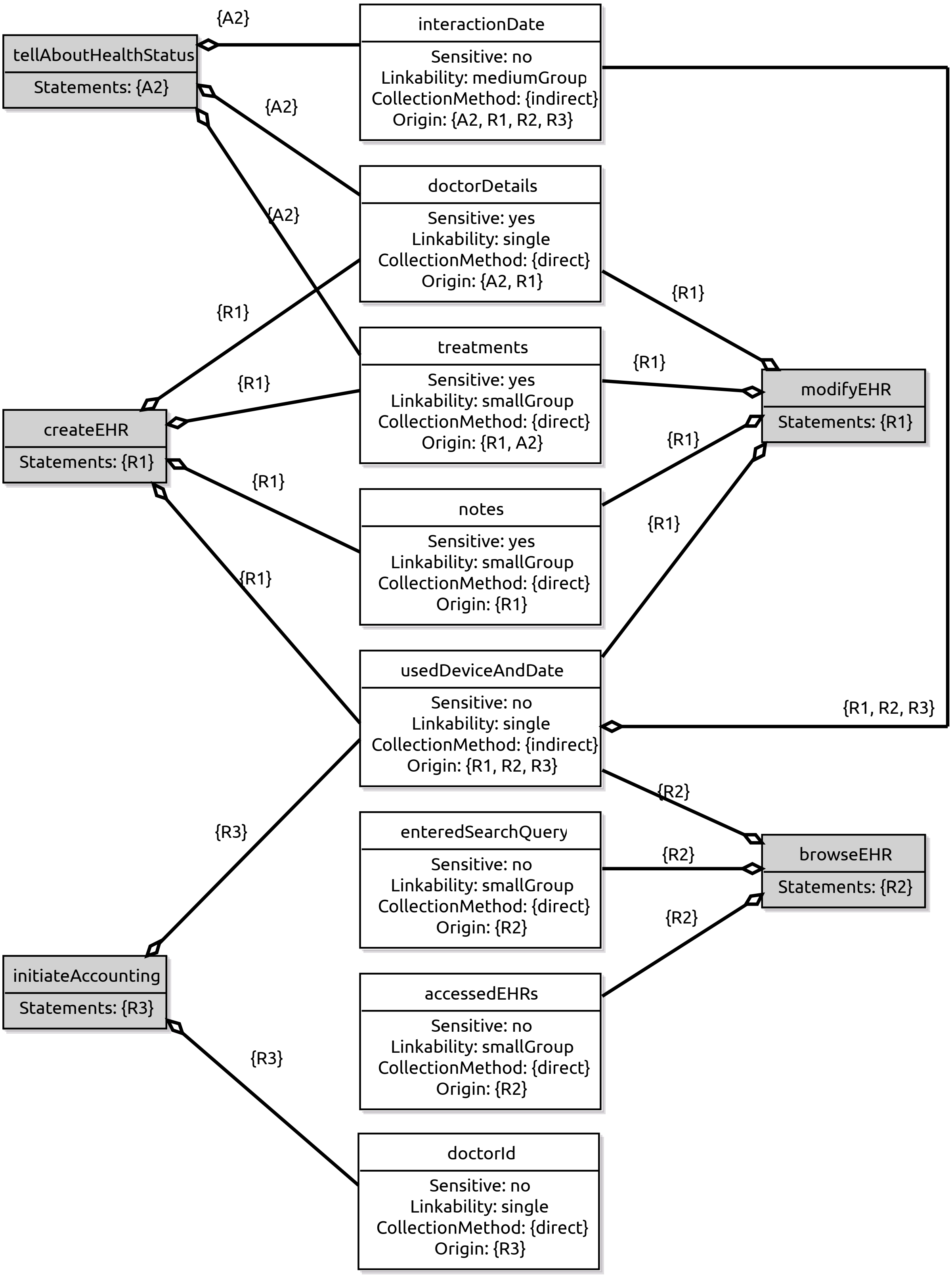


Figure 10.6.: *Personal data diagram for the data subject* Doctor

domains that are constrained by these. For each constrained domain, a data flow edge is created with the data subject as source and the constrained domain as target. This process is iteratively

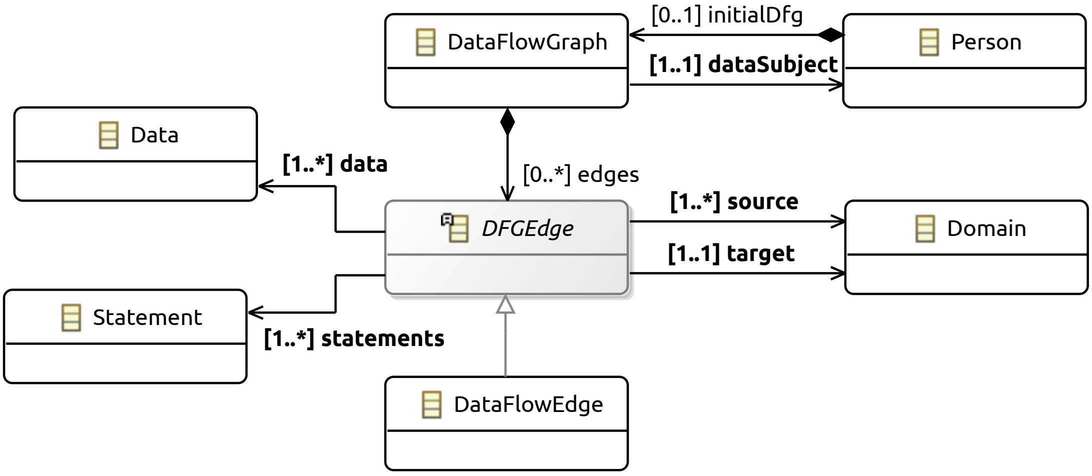


Figure 10.7.: *The metamodel for the initial data flow graph*

Listing 10.9: *Additional attributes for the classes* Statement *and* Domain *of the UDEPF metamodel*

```
context problemframes::Statement
def: constrains : Set(Domain) = self.statementreferences→select(sr|
    sr.oclIsTypeOf(ConstrainsReference)).domain
def: refersTo : Set(Domain) = self.statementreferences→select(sr|
    sr.oclIsTypeOf(RefersToReference)).domain

context problemframes::Domain
def: referredToBy : Set(Statement) = self.statementreferences→select(sr|
    sr.oclIsTypeOf(RefersToReference)).statement
def: initDfgEdgesOneStep :
    Set(Tuple(source:Domain, statement:Statement, target:Domain)) =
      self.referredToBy→collect(s|s.constrains→collect(d|
        Tuple{source=self, statement=s, target=d}))→reject(t|
          t.source = t.target)
def: initDfgEdges :
  Set(Tuple(source:Domain, statement:Statement, target:Domain)) =
    self.initDfgEdgesOneStep→closure(t|t.target.initDfgEdgesOneStep)
```

continued for all domains that are the target of a data flow edge. This process can be formalized by the additional attributes to the classes Statement and Domain from the UDEPF metamodel specified by the OCL expressions shown in Listing 10.9. The attributes constrains and refersTo for the class Statement return the set of all domains the statement constrains and refers to, respectively. The attribute referredToBy for the class Domain returns the set of all statements that refer to it. The attribute initDfgEdgesOneStep formalizes one step of the edge generation as described above for the data subject. The edges for the domain are represented as tuples consisting of the domain itself as source, the statement that refers to the domain, and the domains constrained by the statement as target. The attribute initDfgEdges iteratively applies the step of the graph generation (initDfgEdgesOneStep) starting at the domain itself and proceeding with the domains that are the target of a contained edge using the closure operator.

For the initial data flow graph, the data annotated at all edges are the personal data of the data subject for which the graph is generated. That is, the flow of personal data is over-approximated in the sense that it is expected that all personal data available at the source domain flow to the target domain due to the annotated statements.

Using the additional attribute initDfgEdges (see Listing 10.9), I specified the properties that the initial data flow graph of a data subject shall have as three OCL invariants on the class

Listing 10.10: *Invariants on the initial data flow graph*

```
context Person
inv: self.initialDfg.edges→forAll(e|
  e.data = self.relatedtos.personalData and e.statements→forAll(s|
    s.statement.refersTo→includes(source.domain) and
    s.statement.constrains→includes(target.domain)))
inv: self.initDfgEdges→forAll(edgeToFind|self.initialDfg→exists(e|
  e.source = edgeToFind.source and e.target = edgeToFind.target and
  e.statements→includes(edgeToFind.statement))
inv: self.initialDfg.edges→isUnique(e|Tuple{source=e.source, target=e.target})
```

Person in Listing 10.10. First, the data annotated at the data flow edges has to be the personal data of the person, and the statements annotated at the edge have all to refer to the source of the edge and to constrain the target. Second, each edge represented by a tuple returned by the attribute initDfgEdges has to be represented by a data flow edge in the initial data flow graph that has the same source and target, and includes the statement of the edge to be represented in its set of statements. Third, there shall be at most one data flow edge between two domains in the initial data flow graph.

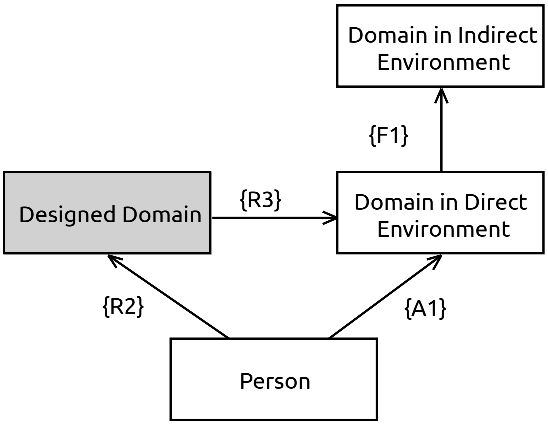


Figure 10.8.: *Example for an initial data flow graph*

The initial data flow graphs for the identified data subjects (persons for whom personal data were identified) can be generated fully automatically by the ProPAn tool. Figure 10.8 shows how an initial data flow graph is presented in the ProPAn tool. The data flow edges are presented as arrows pointing from the source domain to the target domain. The edges are annotated with the statements due to which the data flow exists. The flowing personal data are not annotated for the initial data flow graph, because they are for all edges all personal data of the data subject. A domain is presented as white rectangle if it represents a given domain (part of the machine's environment; cf. Chapter 2) and as a gray rectangle if it represents a designed domain (part of the machine). This differentiation between given and designed domains visualizes which data flows correspond to a collection of personal data by the machine (arrow pointing from a given domain to a designed domain), and which data flows correspond to a flow of personal data from the machine to a domain of its direct environment (arrow pointing from a designed domain to a given domain). I present the data flow graphs level-wise. The bottom level contains the data subject, which is the domain Person in Figure 10.8. On the next level, all domains are put to which personal data flow from the data subject. This process is continued following a breadth-first strategy, i.e., if all edges starting at the domains of one level were considered, then the edges starting at the domains of the just created level are considered to create the following

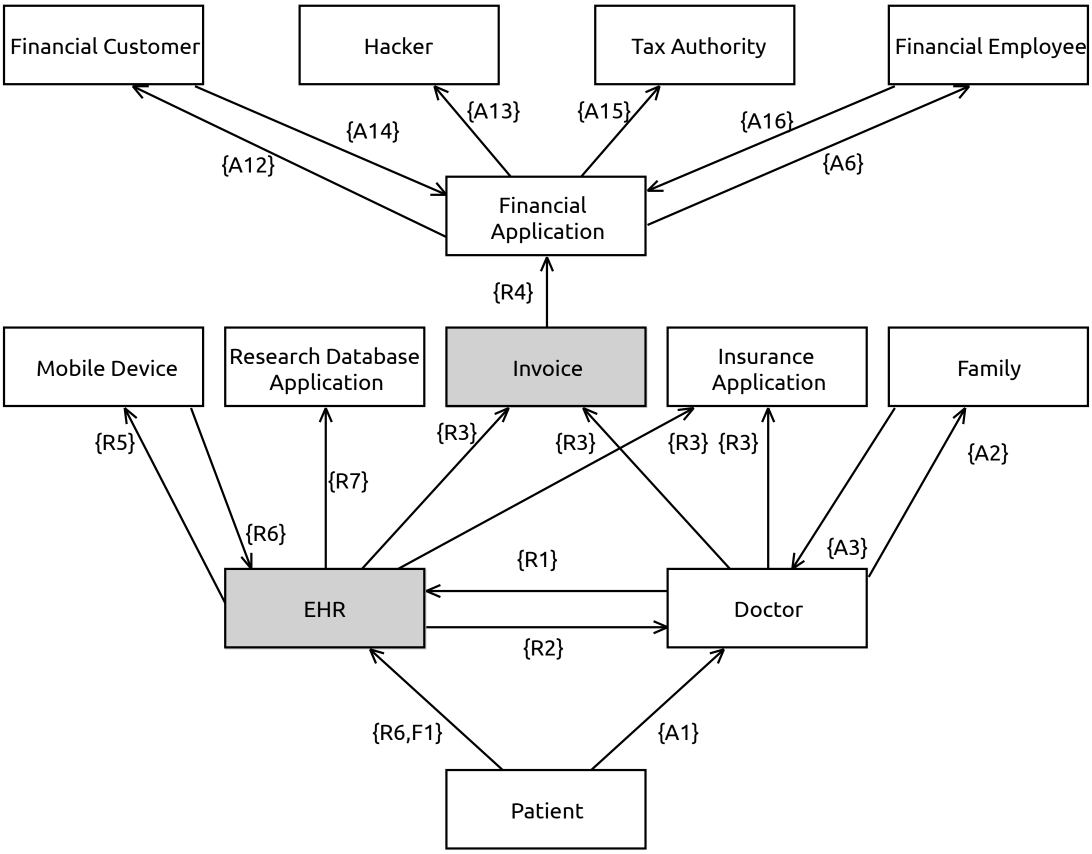


Figure 10.9.: *Initial data flow graph for the data subject Patient*

level. Note that a domain keeps the first level assigned to it, that arrows may point from a higher-level to a lower-level, and that the initial data flow graph may be cyclic (cf. Figure 10.9).

10.4.3. Application to EHS

The personal data possibly flowing through the system-to-be are listed in Figures 10.5 and 10.6 for the data subjects patient and doctor, respectively. These personal data are added to the data flow edges of the respective initial data flow graphs.

The initial data flow graph for the data subject patient and doctor are shown in Figure 10.9 and Figure 10.10. These graphs visualize the possible flows of the patient's and doctor's personal data in the system-to-be based on the functional requirements, facts, and assumptions documented in the problem frame model.

As mentioned before, I only consider the previously introduced functional requirements, facts, and assumptions for the EHS. In a more elaborated analysis of the EHS, the domain knowledge that vital signs of the patient are stored on his or her mobile device, that the patient's family may gain information from him or her, and that researchers can access the information available at the research data base application should also be documented. This would lead to additional data flow edges between the patient and the mobile device, the patient and his or her family, and the research database application and the researcher.

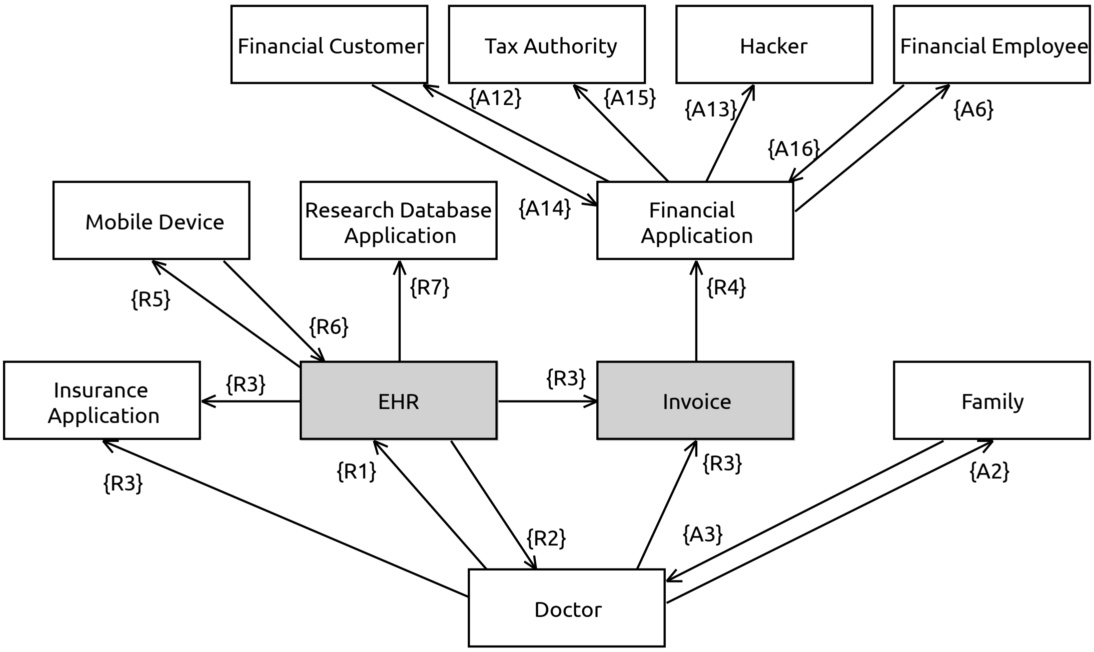


Figure 10.10.: *Initial data flow graph for the data subject* Doctor

10.5. Decide on Privacy Threshold

Having collected the personal data processed by the system-to-be and how they flows through the system, the analysis team has to decide whether a further privacy analysis is necessary in the substep Decide on privacy threshold.

10.5.1. Threshold Analysis Questionnaire

To support this step, I derived five questions from existing privacy threshold analysis templates (listed in Section 10.1). These questions are listed in Table 10.6 and all aim at the identification of personal data. Initial answers to these questions can automatically be derived from the ProPAn model using the OCL expressions listed in Listing 10.11.

Question TA1 aims at eliciting all personal data collected by the machine. The machine collects personal data if there is a data flow edge in a data subject's data flow graph starting at a given domain and ending at a designed domain, because the designed domains belong to the machine and the given domains to the environment of the machine (cf. Section 2.2). The attribute collectionEdges specified in Listing 10.11 collects all these edges for a DataFlowGraph. The attribute TA1 for a ProPAnModel collects for all instances of the class Person the personal

Table 10.6.: *Threshold analysis questions*

No.	Question	Answer type
TA1	Which personal data are collected by the machine?	Set(Data)
TA2	Which personal data are stored by the machine?	Set(Data)
TA3	Which personal data are provided to other domains by or due to the machine?	Set(Data)
TA4	Which personal data are processed by the machine identifying the data subject?	Set(Data)
TA5	Which sensitive personal are data processed by the machine?	Set(Data)

Listing 10.11: *OCL expressions to derive initial answers to the threshold analysis questions*

```
context DataFlowGraph
def: collectionEdges : Set(DFGEdge) =
  self.edges→select(e|e.target.domain.designed and
    e.source.domain→exists(d|not d.designed))
def: storageEdges : Set(DFGEdge) =
  self.edges→select(e|e.target.domain.designed)
def: flowEdges : Set(DFGEdge) =
  self.edges→select(e|(e.source.domain→exists(d|d.designed) and
    not e.target.domain.designed) or
      e.statements.statement→exists(s|s.oclIsTypeOf(Requirement)))

context ProPAnModel
def: TA1 : Set(Data) =
  self.domains→select(d|d.oclIsTypeOf(Person)).initialDfg.collectionEdges.data
def: TA2 : Set(Data) =
  self.domains→select(d|d.oclIsTypeOf(Person)).initialDfg.storageEdges.data
def: TA3 : Set(Data) =
  self.domains→select(d|d.oclIsTypeOf(Person)).initialDfg.flowEdges.data
def: TA4 : Set(Data) =
  TA1→union(TA2)→union(TA3)→select(data|
    data.relatedtos.linkability→includes(Linkability::single))
def: TA5 : Set(Data) =
  TA1→union(TA2)→union(TA3)→select(data|
    data.relatedtos.sensitive→includes(true))
```

data referenced by a collection edge in his or her initial data flow graph.

All personal data stored by the machine shall be elicited by answering question TA2. The machine stores personal data if there is a data flow edge in a data subject's data flow graph ending at a designed domain. In comparison to TA1, TA2 also considers machine-internal data flows that may lead to a storage of personal data. The attribute storageEdges specified in Listing 10.11 collects all these edges for a DataFlowGraph. The attribute TA2 for a ProPAnModel collects for all instances of the class Person the personal data referenced by a storage edge in his or her initial data flow graph.

Question TA3 aims at eliciting all personal data that flows from the machine (or a designed domain) to a given domain, or that flows between given domains due to a functional requirement, and hence, might not occur in the enviroment without the machine integrated into it. The attribute flowEdges specified in Listing 10.11 collects all data flow edges for a DataFlowGraph. The attribute TA3 for a ProPAnModel collects for all instances of the class Person the personal data referenced by a flow edge in his or her initial data flow graph.

Which of the processed personal data (collected, stored, or provided by the machine to other domains) allow to identify the data subject and are sensitive personal data is elicited by questions TA4 and TA5, respectively.

If the ProPAn tool returns for all questions an empty set of data, then the machine does not process personal data and hence, a further privacy analysis is not necessary. The analysis team may also decide that a detailed privacy analysis is not necessary if the processed personal data are not sensitive and do not allow to identify the data subjects. However, the analysis team has to decide on the privacy threshold in compliance with applicable legislation. If the analysis team has decided that a further privacy analysis is necessary, the ProPAn method is continued with the step Data Flow Analysis described in Chapter 11.

10.5.2. Application to EHS

The answers to questions TA1-TA5 derived from the ProPAn model of the EHS are given in Table 10.7. The answers to TA1 and TA3 contain all personal data of the patient and doctor that have been identified. Note that these results are based on the over-approximation of the data flows. The detailed data flow analysis performed in the step Data Flow Analysis (see Chapter 11) may reveal that not all of these personal data are processed by the machine.

Table 10.7.: *Threshold analysis answers for the EHS*

No.	Personal data
TA1	{healthStatus, vitalSigns, patientDetails, interactionDate, doctorDetails, treatments, notes, usedDeviceAndDate, enteredSearchQuery, accessedEHRs, doctorId}
TA2	{healthStatus, vitalSigns, patientDetails, interactionDate, doctorDetails, treatments, notes, usedDeviceAndDate, enteredSearchQuery, accessedEHRs, doctorId}
TA3	{healthStatus, vitalSigns, patientDetails, interactionDate, doctorDetails, treatments, notes, usedDeviceAndDate, enteredSearchQuery, accessedEHRs, doctorId}
TA4	{patientDetails, doctorDetails, usedDeviceAndDate, doctorId}
TA5	{healthStatus, vitalSigns, patientDetails, doctorDetails, treatments, notes}

The answers to questions TA4 and TA5 show that the machine of the EHS processes sensitive personal data and personal data that allows to identify the data subject. Hence, a detailed privacy analysis is necessary for the EHS.

10.6. Comparison to the State of the Art

In this section, I discuss briefly the state of the art methods that I introduced in Chapter 3 and that explicitly consider the identification of personal data and data subjects, and a privacy threshold analysis. Several methods have steps in which personal data are explicitly documented or modeled (Antignac et al., 2016; Yee, 2017; Oliver, 2016; De and Le Métayer, 2016; Ahmadian and Jürjens, 2016; van Blarkom et al., 2003; Crespo et al., 2015; Drgon et al., 2016), while only two methods consider a privacy threshold analysis or initial privacy impact assessment (Crespo et al., 2015; Drgon et al., 2016).

Antignac et al. (2016), Yee (2017), and Oliver (2016) propose to add annotations to DFDs that indicate the processed personal data and the data subjects. However, the authors do not provide support to identify the personal data or data subjects. The initial data flow graphs that I propose to generate from the problem frame model are comparable to the annotated DFDs proposed by Antignac et al., Yee, and Oliver.

De and Le Métayer (2016) collect the personal data processed by a system using a template summarizing seven attributes (form, precision, volume, purpose, retention, visibility, and intervenability). These attributes are not collected in this step of the ProPAn method, but in the step Data Flow Analysis (see Chapter 11) for each domain at which the personal data are available, and in the step Privacy Requirements Identification (see Chapter 12) in the form of privacy requirements. De and Le Métayer provide no support for the elicitation of the personal data.

Ahmadian and Jürjens (2016) allow to annotate UML diagrams with stereotypes to indicate the processing of personal data. In comparison to ProPAn, the authors do not provide support to identify the personal data.

van Blarkom et al. (2003) also require the identification of personal data in their method, because they use these as assets for a risk assessment, but they do not describe a method to identify these.

Crespo et al. (2015) and Drgon et al. (2016) describe high-level privacy engineering methods and hence, they do not provide details on how to elicit personal data and to perform a privacy threshold analysis. However, the steps to identify personal data and to perform a privacy threshold analysis are contained in their methods.

I can conclude that the systematic identification and modeling of personal data based on a problem frame model, as well as, the generation of the initial data flow graphs are novel contributions to the state of the art. These contributions additionally support executing a privacy threshold analysis that is not yet methodological supported by the state of the art methods.

10.7. Conclusions

In this chapter, I have introduced the step Privacy Threshold Analysis of the ProPAn method. I introduced a detailed procedure with supporting questionnaires to elicit and model personal data in a ProPAn model, to generate initial data flow graphs visualizing the potential data flows through the system-to-be, and to decide on the privacy threshold for a system-to-be. All steps are supported by the ProPAn tool and utilize artifacts of the problem frame and ProPAn model. I applied all substeps of the Privacy Threshold Analysis on the EHS scenario. Furthermore, I described how a ProPAn model is connected to the problem frame model it is created for (see Figure 10.3) in this chapter.

My proposed privacy threshold analysis provides more support for the identification of personal data than any other state of the art privacy requirements engineering method. The ProPAn tool provides support for filling out PTA templates (cf. Section 10.1) because it can derive from the created ProPAn model which personal data are processed by the system-to-be, and whether these are sensitive or allow to identify the related data subject. Furthermore, I have shown in the substep Generate initial data flow graphs that graphs similar to DFDs can automatically be generated from a problem frame model.

The following step of the ProPAn method (Data Flow Analysis) is described in Chapter 11. In this step, I use the personal data identified in this step, and the problem and domain knowledge diagrams to perform a detailed analysis of the data flows in the system-to-be. As a result of the following step, the analysis team obtains a data flow graph (similar to the initial data flow graph) that reflects the data flows intended by the functional requirements and domain knowledge of the system-to-be.

Chapter 11

Computer-aided Data Flow Analysis

The third step of the ProPAn method is called Data Flow Analysis. In this chapter, I provide the details on this step. The goal of the data flow analysis is to perform an in-depth analysis of the flows of personal data implied by the functional requirements and domain knowledge of the system-to-be. In the previous step (Privacy Threshold Analysis), these were over-approximated to perform an early privacy threshold analysis. In this step, a fine-grained analysis of the data flows is performed. As result of the data flow analysis, the analysis team obtains information about the availability of personal data at the different domains of the system and hence, how the personal data flows through it.

The data flow analysis process is partly based on the paper (Meis and Heisel, 2015) and its extended version (Meis and Heisel, 2016a). I am the main author of both papers and Maritta Heisel provided valuable feedback that helped to improve the papers.

I provide an introduction to the purpose of this step in Section 11.1. The substeps Initialize data flow analysis, Identify personal data flow from a statement, and Generate final data flow graph of this step are introduced in Sections 11.2, 11.3, and 11.4, respectively. I investigate whether the state of the art privacy requirements engineering methods have similar steps to ProPAn's data flow analysis in Section 11.5, and I conclude this chapter in Section 11.6.

11.1. Introduction

To derive the privacy requirements on a system and threats to them, it has to be elicited how the personal data processed by the system-to-be flows through it. This includes the elicitation of the information

1. how the personal data are collected from the data subjects,
2. in which amount the personal data are available at the domains of the system,
3. how long the personal data are retained at the domains,
4. due to which statements the personal data are available at a domain,
5. for which purpose the personal data need to be available at the domain, and
6. which personal data available at a domain need to be linkable to each other.

The review of the state of the art of privacy requirements engineering presented in Chapter 3 has shown that no method exists that supports the systematic elicitation of the previously mentioned information. However, most methods need this information as input.

In this chapter, I describe a systematic method to elicit the previously mentioned information in a systematic and tool-supported method based on the requirements, facts, and assumptions given by the problem and domain knowledge diagrams of a problem frame model, and the

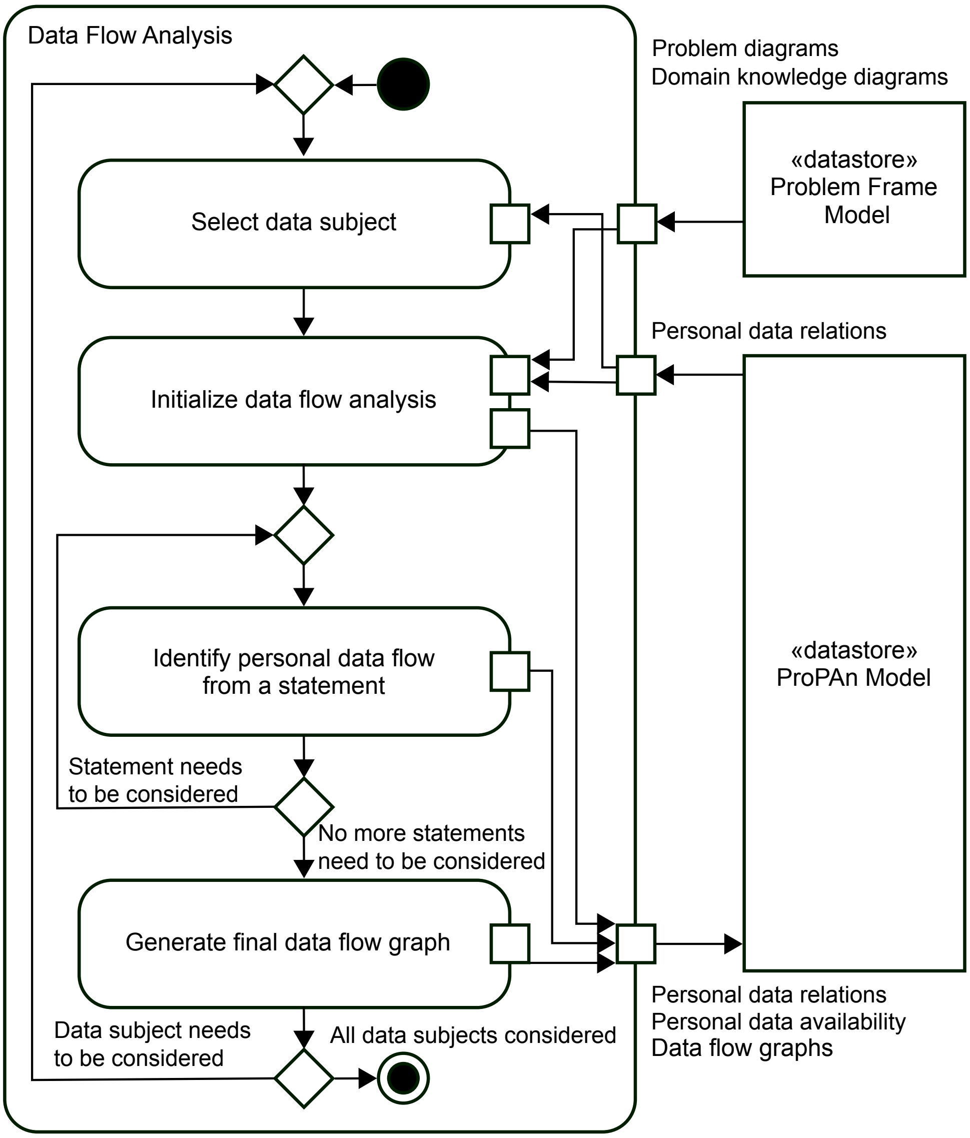


Figure 11.1.: *Detailed view on the step* Data Flow Analysis *of the ProPAn method*

personal data identified during the previous step of the ProPAn method (see Chapter 10). Figure 11.1 provides a detailed view on the substeps of the step Data Flow Analysis. As first step, the data subject has to be selected for whom the data flow analysis shall be performed. Data subjects are all persons in the ProPAn model for whom personal data were identified in the step Privacy Threshold Analysis. Then the data flow analysis is initialized in the substep Initialize data flow analysis. During this substep, a so-called *data flow analysis graph* is generated for the selected data subject that guides the analysis of the flows of the data subject's personal data. Additionally, it is documented which personal data are initially available at the data subject who is the *source* of the personal data flows. The substep Identify personal data flow from a statement is iteratively applied to the statements of the data flow analysis graph that still need to be considered, until no more statements have to be considered. During this step, the flows of the data subject's personal data due to a selected statement and the above listed information are identified and added to the ProPAn model. Additionally, new personal data (contained or derived from the already identified personal data) may be identified and added to the ProPAn model. Finally, the so-called *final data flow graph* that visualizes the elicited flows of personal data for the selected data subject is generated in the step Generate final data flow graph. This graph is similar to a data flow diagram (DFD) that is used as input of several state of the art privacy requirements engineering methods.

11.2. Initialize Data Flow Analysis

In this section, I describe the initialization of the data flow analysis for an arbitrarily selected data subject. The flow of a data subject's personal data through the system is analyzed based on the functional requirements of the system-to-be, and the domain knowledge elicited during the step Privacy Context Elicitation (see Chapter 9). The analysis of the intended data flows is performed for a specific data subject and starts at the data subject as source of his or her personal data. The personal data may flow through the system due to its functional requirements and the domain knowledge elicited in step Privacy Context Elicitation. This is, a functional requirement, assumption, or fact may imply a flow of personal data from the domains it *refers to* to the domains it *constrains*. On the same basis, I generate the initial data flow graphs (see Section 10.4). Note that if the analysis team notices that a specific data flow is not covered by the functional requirements, or domain knowledge, a functional requirement is missing, or privacy-relevant domain knowledge was missed in the step Privacy Context Elicitation. To document which personal data are available at the domains of the system, I introduce available data relations that connect data and domain objects of the problem frame model.

The generation of the so-called *data flow analysis graph* that guides the data flow analysis is explained in Section 11.2.1. The initialization of the available data relations for the data subject and his or her personal data is introduced in Section 11.2.2. I present the application of the substep Initialize data flow analysis to the EHS example in Section 11.2.3

11.2.1. Data Flow Analysis Graph

A data flow analysis graph for a data subject visualizes the "network" of statements and domains of the problem frame model starting at the data subject. A data flow analysis graph is a bipartite graph similar to a petri-net with domains and statements as nodes (see Figure 11.4). In terms of petri-nets, the domains correspond to places at which data are available and to which data flow, and the statements correspond to transitions that "consume" the data available at the input places and make them available at the output places. Note that in contrast to standard petri-net semantics, the transitions do not consume the data available at the places. That is, the data remains available at the domains once they flew there.

The metamodel for data flow analysis graphs is shown in Figure 11.2. A DataFlowAnalysisGraph is created for exactly one Person, the data subject. Data flow analysis graphs contain two kinds of edges. Both types of edges connect a Statement to a Domain, and are annotated with a Phenomenon set.

An InputEdge starts from a domain and points to a statement. Input edges are generated from the RefersToReferences (see Figure 2.10 on page 22) of the problem frame model. An input edge's statement, domain, and phenomena (from the ProPAn model) correspond to the statement, domain, and phenomena of the problem frame model that the corresponding RefersToReference references. Analogously, an OutputEdge starts from a statement and points to a domain. Output edges are generated from the ConstrainsReferences of the problem frame model. An output edge's statement, domain, and phenomena (from the ProPAn model) correspond to the statement, domain, and phenomena of the problem frame model that the corresponding Constrains References references. The data flow analysis graph can be generated fully automatically by the ProPAn tool. The generation starts by adding for each RefersToReference pointing to the data subject, a corresponding input edge. Then inductively for each statement referenced by an input edge, output edges are generated for all ConstrainsReferences of the statement, and for

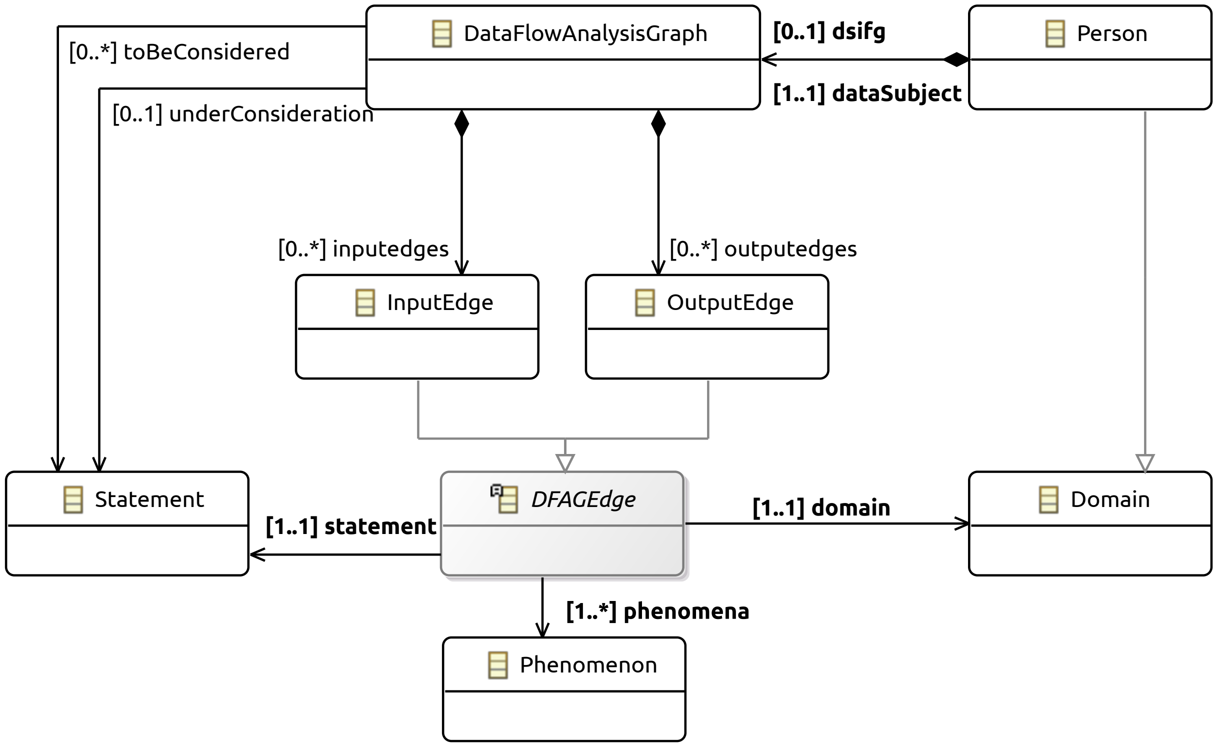


Figure 11.2.: *Metamodel for the data flow analysis graph*

Listing 11.1: *OCL invariant specifying the elements of a data flow analysis graph*

```
context DataFlowAnalysisGraph
def: inputEdgesContained(Domain domain) : Boolean =
  domain.domain.statementreferences→select(sr|
    sr.oclIsTypeOf(RefersToReference))→forAll(rtr|
      self.inputedges→one(ie|
        ie.statement.statement = rtr.statement and
        ie.domain.domain = rtr.domain and
        ie.phenomena.phenomenon = rtr.contains))
def: outputEdgesContained(Statement statement) : Boolean =
  statement.statement.statementreferences→select(sr|
    sr.oclIsTypeOf(ConstrainsReference))→forAll(cr|
      self.outputedges→one(oe|
        oe.statement.statement = rtr.statement and
        oe.domain.domain = rtr.domain and
        oe.phenomena.phenomenon = rtr.contains))
inv: inputEdgesContained(self.dataSubject) and
  self.inputedges→forAll(ie|outputEdgesContained(ie.statement)) and
  self.outputedges→forAll(oe|inputEdgesContained(oe.statement))
```

each domain referenced by an output edge, input edges are generated for all RefersToReferences referencing the domain. This is formalized by the OCL invariant shown in Listing 11.1. The global definitions inputEdgesContained and outputEdgesContained are used to check whether all input edges for a domain and all output edges for a statement are contained in the data flow analysis graph, respectively. These definitions are used in the invariant to specify the three previously mentioned generation rules.

To store the state of the data flow analysis, the class DataFlowAnalysisGraph has a reference to the set of statements that still need toBeConsidered, and a reference to the statement that is currently underConsideration. Initially, the statements to be considered are those that refer to the data subject, because the data subject serves as the starting point for the data flow analysis. The attribute underConsideration is initially not set. I describe in Section 11.3 how these references are updated during the data flow analysis.

11.2.2. Initial Available Data Relations

To model which personal data are available at the different domains of the system, I introduce the relation PersonalDataAvailableAt. In addition to the availability of personal data at a domain, I also propose to document the linkability of the available personal data to each other at the respective domain. This is realized by the relation LinkAvailableAt. The metamodel for these relations is shown in Figure 11.3. PersonalDataAvailableAt and LinkAvailableAt objects belong to exactly one Domain, namely the domain at which the personal data and link are available. A PersonalDataAvailableAt object references one Data object (the personal data to be available at the domain), while a LinkAvailableAt object references one Link that relates two Data objects, which are linkable at the domain. Note that I consider links between data objects as symmetric, which means that the relation LinkAvailableAt is symmetric. This models that the information about the link between two data objects is available at a domain, which does not necessarily mean that one of these data objects contains itself a link to the other data object. The attribute linkability of the class LinkAvailableAt specifies the degree of linkability between the two data objects given by the link. The linkability value single means that the data instances that belong to each other can be linked to each other. The values smallGroup, mediumGroup, and largeGroup specify that it is only possible to link the data instances to small, medium, and large sets of candidates of data instances they are possibly related to. Note that the literal anonymous is forbidden, instead no relation should be modeled. Note also that the meaning of the literals smallGroup, mediumGroup, and largeGroup has to be defined for each system-to-be and possibly for each data subject in the system-to-be. Furthermore, the classes PersonalDataAvailableAt and LinkAvailableAt inherit from the abstract class *AvailableAt* attributes that allow to document more details on the availability of the data and link, respectively.

How long the data or link are available at the domain is captured by the attribute duration. The possible values are, forAction, i.e., the data or link are only as long available as it is necessary for the purposes for which the data or link are available at the domain, untilDeleted, i.e., the data or link are longer available than needed for the purpose, but deleted after a specific duration in compliance with regulations, and unlimited, i.e., there are no constraints on the retention of the data or link at the domain.

As domains represent classes of entities with similar properties and not a single individual, I use the attribute availability to document at which instances of a domain the data or link may be available. The possible values are individual, i.e., the personal data or link are only available at the individual entity that receives the personal data or link, authorized, i.e., the data or link may be available at authorized instances of the domain, and all, i.e., the personal data or link are possibly available to all instances of the domain. Note that the enumeration AvailabilityDegree also contains the literal none. However, I do not allow to use this literal, because instead no available at relation should be created.

The amount of data records or links available at the domain is described by the attribute amount. The possible values are single, i.e., only one data record or link is available at the domain, small, medium, large, i.e., a small, medium, or large set of data records is available at the domain or linkable to each other, and all, i.e., all records that are processed by the system are available at the domain or all records available at the domain are linkable to each other. Note that the meaning of the literals small, medium, and large has to be defined for each system-to-be and possibly also for each data object.

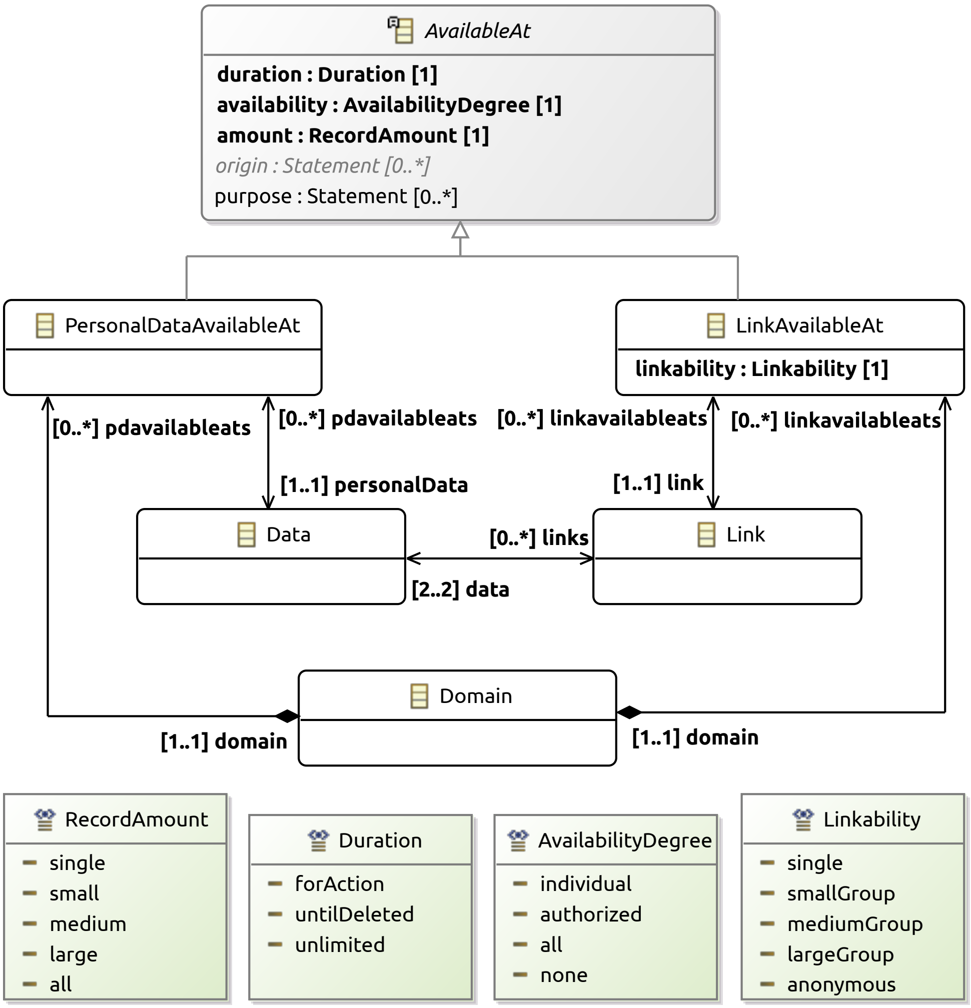


Figure 11.3.: *Metamodel for the relation PersonalDataAvailableAt*

The attribute origin allows to document from which statements it was identified that the personal data or link are available at the domain, and the attribute purpose documents due to which statements the personal data or link available at the domain flow to another domain. These two values are set automatically during the data flow analysis by the ProPAn tool.

Initially, I assume that all personal data identified in the previous step are available at the respective data subjects. Hence, the ProPAn tool automatically generates for each RelatedTo relation (see Figure 10.2 on page 157) a PersonalDataAvailableAt relation with the data subject as domain. The attributes personalData and origin are set to the personal data and origin of the RelatedTo relation, the duration is set to unlimited, availability to individual, and amount to all. This models that all personal data of an individual data subject are available at him or her for an unconstrained time period. Additionally, for each pair of personal data available at the data subject, the ProPAn tool generates a LinkAvailableAt instance with the linkability single. The attributes duration, availability, amount, and origin are also set to unlimited, individual, all, and the intersection of the origins of the data objects' PersonalDataAvailableAt relations, respectively. That means, the data subject knows that the personal data about him or her are related to him- or herself. Note that no LinkAvailableAt relation needs to be created if a Contains relation (see Figure 10.2 on page 157) exists between the respective data. This is, because if a data object contains another data object, then it is known that these belong to each other (corresponds to the linkability single), this link has the same availability and duration as the data objects, and the

Listing 11.2: *OCL specification of the operation initializeDataFlowAnalysis*

```
context Person::initializeDataFlowAnalysis()
pre: true
post:
self.relatedtos→forAll(rt|
  self.pdavailableats→one(pda|
    pda.personalData = rt.personalData and
    pda.duration = Duration::unlimited and
    pda.availability = AvailabilityDegree::individual and
    pda.amount = RecordAmount::all and
    pda.origin = rt.origin and
    pda.purpose = Set{})) and
self.pdavailableats→forAll(pda1,pda2|
  (pda1.contains.contained→incldues(pda2) or
    pda2.contains.contained→includes(pda1)) xor
  self.linkavailableats→one(la|
    la.link.data = Set{pda1.personalData, pda2.personalData} and
    la.duration = Duration::unlimited and
    la.availability = AvailabilityDegree::individual and
    la.amount = RecordAmount::all and
    la.origin = pda1.origin→intersection(pda2.origin) and
    la.purpose = Set{}))
```

amount of links between the data is all. The initialization for a specific data subject (instance of class Person) is performed by the operation initializeDataFlowAnalysis that is formally specified in Listing 11.2.

The PersonalDataAvailableAt and LinkAvailableAt relations of a domain are visualized in a so-called *available data diagram* (see Figure 11.5). Instances of the class PersonalDataAvailableAt are presented as rectangle with the name of the personal data in the first row, and the attributes of the instance as following rows. The instances of the class LinkAvailableAt are visualized as associations between the PersonalDataAvailableAt instances representing the data connected by the link of the LinkAvailableAt instance. The attributes of the LinkAvailableAt instance are presented in a rectangle with rounded corners that is linked by a dashed line to the association. In addition to the newly introduced elements, an available data diagram shows each Contains relation that connects two data objects for which PersonalDataAvailableAt instances exist in the diagram as an aggregation between these instances.

11.2.3. Application to EHS

For the sake of simplicity, I only show the data flow analysis for the data subject patient. The generated data flow analysis graph for the patient is shown in Figure 11.4. The statements A1, F1, and R6 that all refer to the data subject Patient are highlighted in the graph to express that these statements have to be considered (see reference toBeConsidered in Figure 11.2). The analysis team has to select one of these as the first step of the data flow analysis.

Figure 11.5 shows the initial available data diagram for the patient, which is automatically generated by the ProPAn tool. It contains for each RelatedTo object with the patient as dataSubject (see the personal data diagram in Figure 10.5 on page 161) a PersonalDataAvailableAt object with the previously described initial values. It also shows the contains relation that I added in the previous step to specify that the vitalSigns are contained in the healthStatus, and due to which no LinkAvailableAt instance needs to be created between the vitalSigns and healthStatus. Between all other available data, LinkAvailableAt instances are created with the attribute values as described before.

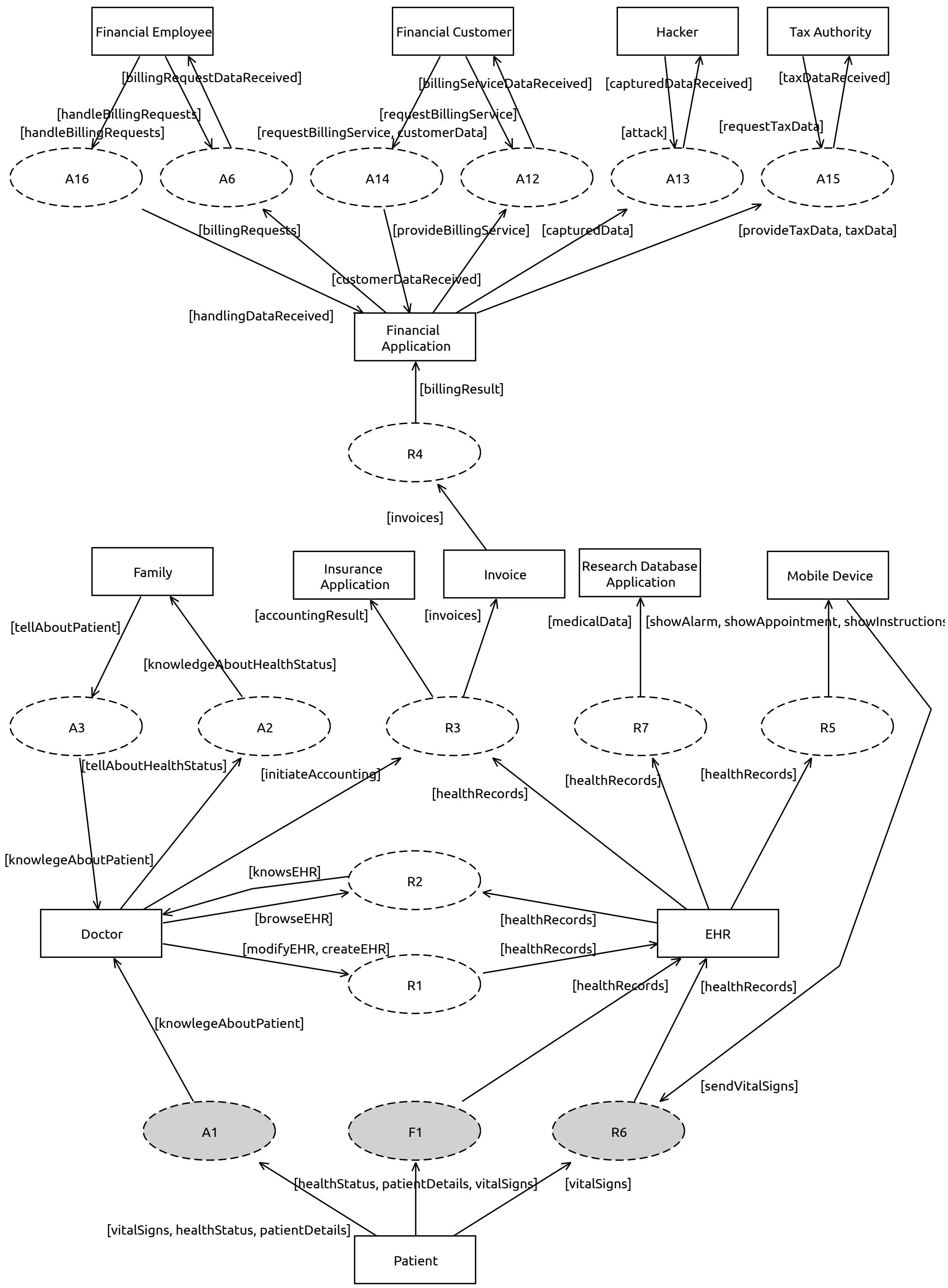


Figure 11.4.: *Data flow analysis graph for the patient with initial marking*

11.3. Identify Personal Data Flow from a Statement

In this section, I explain how the substep Identify personal data flow from a statement is performed. I explain the selection of the statement that shall be considered in Section 11.3.1. The identification of personal data flows based on the selected statement is described in Section 11.3.2. How the set of statements to be considered is modified after the identification of personal data flows is discussed in Section 11.3.3. In Section 11.3.4, I apply the substep Identify personal data flow from a statement on the EHS example.

11.3.1. Select Statement to be Considered

The first task of the substep Identify personal data flow from a statement is to choose one statement of the statements that still need to be considered. As stated before, these are highlighted in the visualization of the data flow analysis graph (see Figure 11.4). A heuristic to select the next statement under consideration is to select a statement for which most personal data flows to its input domains were already elicited. For example, requirement R3 (see Figure 11.4) should be considered when all data flows to the input domains Doctor and EHR where considered. However, the selection of the statement to be considered should not influence the final result of the data flow analysis, because a statement has to be re-considered if new flows of personal data to one of its input domains are identified during the analysis (see also Section 11.3.3).

11.3.2. Identify Personal Data Flows

Having selected a statement to be considered, the analysis team has to investigate which personal data available at the input domains of the statements flow to the output domains of the statement, and hence, are available there due to the statement. To support the analysis team, the ProPAn tool provides a view showing only the statement under consideration and its input and output domains, called *statement consideration view*. This view shows for each input and output domain its available data diagram, the statement under consideration, and the corresponding input and output edges. Figure 11.7 shows an example of a statement consideration view for the considered statement R6. The diagram shows that there is only personal data available at the Patient and no personal data are available at the Mobile Device and EHR. The task of the analysis team is now to specify which personal data available at the input domains flow to the output domains.

I differentiate three kinds of data flows:

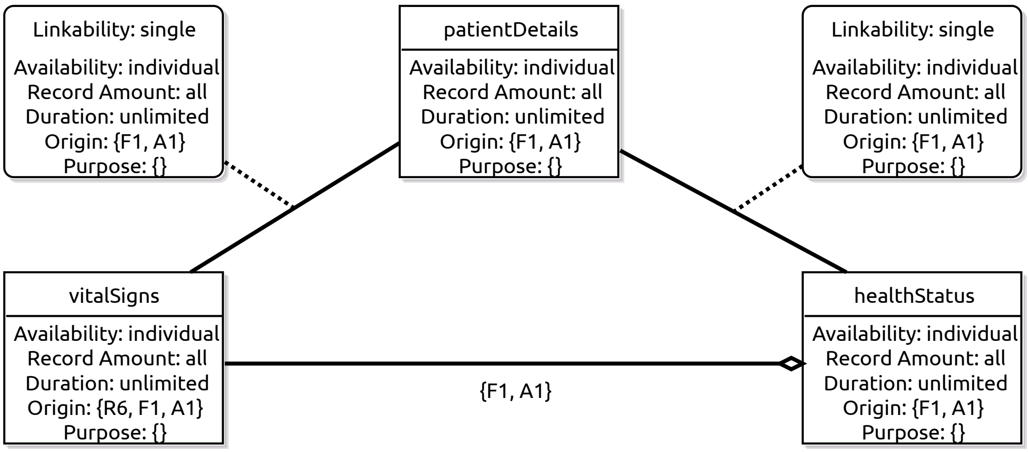


Figure 11.5.: *Initial available data diagram for the domain* Patient

1. personal data available at an input domain may flow to an output domain,
2. personal data that are contained in personal data available at the input domains flow to an output domain,
3. personal data that can be derived from personal data available at the input domains flow to an output domain.

Note that case 2. can only occur when the contains relationship is newly identified, because otherwise the data would already be available at the respective input domain. This is ensured by the ProPAn tool that adds respective PersonalDataAvailableAt instances for all domains at which the data that contains the newly identified personal data are available.

In the following, I first provide general guidelines for the identification of personal data flows in Section 11.3.2.1. Then I describe how flows of available, contained, and derived personal data are modeled in the ProPAn model and how the modeling is supported by the ProPAn tool in Sections 11.3.2.2, 11.3.2.3, and 11.3.2.4, respectively.

11.3.2.1. Identification Questions

To support the analysis team to identify personal data flows from the input domains to the output domains, I provide questionnaires that ask for the personal data that need to be provided to the output domains due to the statement under consideration (see Table 11.1) and the personal data that need to be linkable at the output domains (see Table 11.2). I formulated the questions to ask for the minimal amount, availability, and retention period of the personal data, and the minimal linkability of the personal data that shall be available at the output domain. This shall help the analysis team to follow the privacy principles *use limitation*, *storage limitation*, *collection limitation*, *data minimization*, and *fairness* (see Section 2.4.3). That is, they shall only model those flows of personal data that are necessary to achieve the functional requirements. Note that it has to be guaranteed by the machine and its environment that only those identified minimal flows of personal data happen. In the step Privacy Requirements Identification (see Chapter 12), the documented data flows are used to derive privacy requirements that document that only these minimal flows are allowed to happen.

Table 11.1 has to be filled out for each combination of personal data and output domain to which the personal data shall flow. The first question asks for the name of the personal data flowing to the output domain. Note that the answer type is String, because the flowing data may be contained in or derived from the personal data available at the input domains and hence, not yet be represented by a Data object in the ProPAn model. Exactly one of the questions DF1, DF2, and DF3 should be answered positively. That is, the data flow has to belong to one of the three above listed kinds of data flows. The subquestions of DF2 and DF3 have only to be answered if the data flow belongs to the respective kind.

If DF2 is positively answered, the personal data in which the flowing personal data are contained is elicited with question DF2.1.

If DF3 is positively answered, then questions DF3.1-DF3.3 have to be answered. Question DF3.1 asks for the personal data from which the flowing personal data can be derived, DF3.2 for the domains that are able to derive the data, and DF3.3 whether the machine involved in the statement is able to derive the personal data. Potentially, all domains at which the needed data are available are able to derive the flowing personal data. Whether it is possible to derive the data for an input domain needs to be documented to know whether the derived data are also available at the input domain. The derivation may require specific computational resources or algorithms that need not to be available at all domains. In specific cases, it is possible that at none of the input domains all personal data that are necessary to derive the flowing personal data are available. In these cases and in the cases where the input domains are not able to

Table 11.1.: *Questionnaire for the identification of a personal data flow*

No.	Question	Answer type
DF	Which personal data necessarily need to be provided to the output domain?	String
DF1	Are the personal data available at an input domain?	Boolean
DF2	Are the personal data contained in personal data available at an input domain?	Boolean
DF2.1	In which available personal data are the flowing personal data contained?	Data
DF3	Can the personal data be derived from personal data available at the input domains?	Boolean
DF3.1	From which available personal data can the flowing personal data be derived?	Set(Data)
DF3.2	Which input domains are able to derive the personal data?	Set(Domain)
DF3.3	Does the machine involved in the statement derive the personal data?	Boolean
DF4	Are the flowing data already modeled as personal data of the data subject?	Boolean
DF4.1	To which degree are the flowing personal data linkable to the data subject?	Linkability
DF4.2	Are the flowing personal data sensitive personal data for the data subject?	Boolean
DF5	What is the minimal amount of personal data records that need to be available at the output domain?	RecordAmount
DF6	Which is the smallest group of instances of the output domain to which the personal data need to be available?	AvailabilityDegree
DF7	How long is it necessary to retain the personal data at the output domain?	Duration

Table 11.2.: *Questionnaire for the identification of a link between personal data at an output domain*

No.	Question	Answer type
LF	Which personal data need to be linkable at the output domain?	Tuple(pd1:Data, pd2:Data)
LF1	To which degree have the personal data to be linkable to each other?	Linkability
LF2	What is the minimal amount of links between personal data records that need to be available at the output domain?	RecordAmount
LF3	Which is the smallest group of instances of the output domain to which the links between the personal data need to be available?	AvailabilityDegree
LF4	How long is it necessary to retain the links between the personal data at the output domain?	Duration

derive the data, the data are derived by the machine responsible to address the statement under consideration. If neither the machine, nor an input domain are able to derive the personal data, then the validity of the statement has to be questioned.

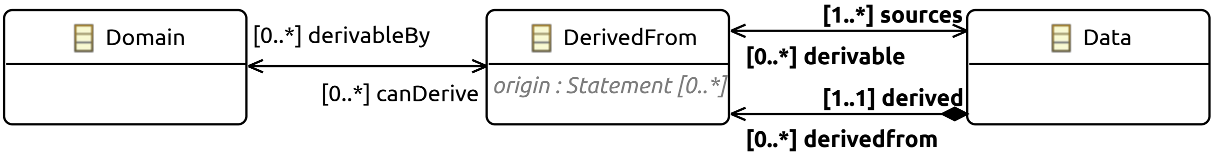


Figure 11.6.: *Metamodel for the* DerivedFrom *relation*

In the case of a flow of contained or derived personal data, the flowing data need to be declared as new personal data of the data subject that is currently considered by creating a respective RelatedTo relation. The attributes needed (see Figure 10.2 on page 157) are elicited by questions DF4.1 and DF4.2. Note that the attribute collection is set by the ProPAn tool. This is explained later in Sections 11.3.2.2-11.3.2.4. Question DF5 elicits the minimal amount of data records that have to be available at the domain, DF6 the most restrictive group of instances of the domain to which the flowing personal data shall be available, and DF7 the minimal time the personal data have to be available at the domain.

Figure 11.6 shows the metamodel for the DerivedFrom relation that allows to model that a Data object can be derived from a non-empty set of Data objects. The domains (including the machines) that are able to derive the personal data are also linked to the DerivedFrom relation. Similar to the previously introduced relations, the DerivedFrom relation has an attribute origin to document from which statements it was identified. I propose to use instances of the class DerivedFrom to document the answers to questions DF3.1-DF3.3 in the ProPAn model (more details are provided in Section 11.3.2.4).

Table 11.2 has to be filled out if the analysis team identifies that a pair of personal data is linkable to each other at an output domain due to the statement under consideration. Question LF1 elicits the needed degree of linkability between the personal data, LF2 the minimal amount of data records that have to be linkable to each other, LF3 the most restrictive group of instances of the domain to which the link shall be available, and LF4 the minimal time the link has to be available at the domain.

11.3.2.2. Add Flow of Personal Data

The simplest case is that personal data available at an input domain flows to an output domain. In this case, an instance of the class PersonalDataAvailableAt needs to be created for the output domain, the personal data, and the answers to questions DF5-DF7. The operation for creating this instance is specified in Listing 11.4. In this specification and the following, the additional attributes inputDomains and outputDomains for the class DataFlowAnalysisGraph that return the input and output domains for the statement under consideration, respectively, are used (see Listing 11.3).

The preconditions of the operation addPersonalDataFlow are that the provided outputDomain is an output domain of the currently considered statement, and that the flowing personal data are available at one of the input domains.

The postcondition states that an object of type PersonalDataAvailableAt was created with the provided attribute values, and the statement under consideration as origin (lines 8-13). Additionally, the statement under consideration is added as purpose to all PersonalDataAvailableAt instances for the flowing personal data at the input domains (lines 14-16).

11.3.2.3. Add Flow of Contained Personal Data

In the case that contained personal data (positive answer to question DF2) shall flow to an output domain, then the operation addContainedPersonalDataFlow can be used to add this flow

Listing 11.3: *OCL specification of two additional attributes of the class DataFlowAnalysisGraph*

```
context DataFlowAnalysisGraph
def: inputDomains : Set(Domain) =
  self.inputedges→select(ie|ie.statement = self.underConsideration).domain
def: outputDomains : Set(Domain) =
  self.outputedges→select(oe|oe.statement = self.underConsideration).domain
```

Listing 11.4: *OCL specification of the operation addPersonalDataFlow*

```
context DataFlowAnalysisGraph::addPersonalDataFlow(outputDomain : Domain,
  name : String, amount : RecordAmount, availability : AvailabilityDegree,
  duration : Duration)
pre:
outputDomains→includes(outputDomain) and
inputDomains.pdavailableats→exists(pa|pa.personalData.name = name)
post:
outputDomain.pdavailableats→exists(pa|
  pa.personalData.name = name and
  pa.amount = amount and
  pa.availability = availability and
  pa.duration = duration and
  pa.origin→includes(self.underConsideration)) and
inputDomains.pdavailableats→forAll(pa|
  pa.personalData.name = name implies
  pa.purpose→includes(self.underConsideration))
```

to the ProPAn model. This operation is specified in Listing 11.5.

The precondition requires that the outputDomain has to be an output domain of the currently considered statement, and that the personal data that shall contain the flowing personal data (parameter container elicited with question DF2.1) are available at an input domain.

The operation addContainedPersonalDataFlow shall create an instance of the class PersonalDataAvailableAt for the output domain, the contained personal data, the attribute values given by the answers to questions DF5-DF6, and the current statement as origin (lines 9-14).

For all input domains at which the containing personal data are available, PersonalDataAvailableAt objects shall be created to document that also the contained personal data are available at them. The attribute values for amount, availability, and duration are set to the values of the AvailableAt relation of the containing personal data, the purpose is set to the set containing the statement under consideration, and the origin is kept unchanged, because the contained personal data do not flow to the input domain due to the statement. (lines 15-22).

Additionally, a Contains object (see Figure 10.2 on page 157) needs to be created to document that the flowing personal data are contained in the container. This contains relation also documents the reason why the contained personal data are available at the input domains. (lines 23-26).

If the contained personal data are newly identified to be personal data (positive answer to question DF4), i.e., no instance of the class RelatedTo (see Figure 10.2 on page 157) exists to document that the data are personal data of the data subject that is currently considered, then such an instance needs to be created using the attributes values for linkability and sensitive elicited by questions DF4.1 and DF4.2, the same collection method as the container, and the currently considered statement as origin (lines 27-35).

Listing 11.5: *OCL specification of the operation* addContainedPersonalDataFlow

```
context DataFlowAnalysisGraph::addContainedPersonalDataFlow(
  outputDomain : Domain, name : String, container : Data,
  amount : RecordAmount, availability : AvailabilityDegree,
  duration : Duration, linkability : Linkability, sensitive : Boolean)
pre:
outputDomains→includes(outputDomain) and
inputDomains.pdavailableats→exists(pa|pa.personalData = container)
post:
outputDomain.pdavailableats→exists(pa|
  pa.personalData.name = name and
  pa.amount = amount and
  pa.availability = availability and
  pa.duration = duration and
  pa.origin→includes(self.underConsideration)) and
inputDomains.pdavailableats→forAll(paContainer|
  pa.personalData = container implies
  pa.domain.pdavailableats→exists(paContained|
    paContained.personalData.name = name and
    paContained.amount = paContainer.amount and
    paContained.availability = paContainer.availability and
    paContained.duration = paContainer.duration and
    paContained.purpose→includes(self.underConsideration))) and
container.contains→exists(c|
  c.contained.oclIsTypeOf(Data) and
  c.contained.name = name and
  c.origin→includes(self.underConsideration)) and
(not self.dataSubject.relatedtos@pre→exists(rt|
  rt.personalData.name = name) implies
  self.dataSubject.relatedtos→exists(rt|
    rt.personalData.name = name and
    rt.linkability = linkability and
    rt.sensitive = sensitive and
    rt.collection = container.relatedTos→any(rtContainer|
      rtContainer.dataSubject = self.dataSubject).collection and
    rt.origin→includes(self.underConsideration)))
```

11.3.2.4. Add Flow of Derived Personal Data

In the case that derived personal data (positive answer to question DF3) shall flow to the output domain, then the operation addDerivedPersonalDataFlow can be used to add this flow to the ProPAn model. This operation is specified in Listing 11.6.

The precondition requires that the outputDomain has to be an output domain of the currently considered statement, and that all data which are needed to derive the flowing personal data (elicited by question DF3.1) are available at the input domains (lines 8-9). Note that not all data have to be available at a single input domain, it is also allowed that the needed data are distributed over the input domains and that the machine performs the derivation based on the provided inputs. Furthermore, all domains that shall be able to derive the personal data, have to satisfy one of the following conditions. First, the domain is an input domain and all personal data necessary to derive the flowing personal data are available at it (lines 11-13). Second, the domain is a machine that is part of the same problem diagram as the statement under consideration (lines 14-17). Note that in the second case, the machine is responsible for satisfying the statement and hence, the statement has to be a functional requirement.

As result of the operation addDerivedPersonalDataFlow, an instance of the class PersonalDataAvailableAt shall be created for the output domain, the contained personal data, the attribute values given by the answers to questions DF5-DF6, and the current statement as origin (lines 19-24).

For all domains that shall be able to derive the flowing personal data, PersonalDataAvailableAt objects shall be created to model that the derived personal data are available at them. The attribute values for amount, availability, and duration are set to the values that also apply for the output domain (questions DF5-DF7), the purpose is set to the set containing the statement under consideration, and the origin is kept unchanged, because the derived personal data do not flow to the input domain due to the statement (lines 25-30).

Additionally, a DerivedFrom object needs to be created to document that the flowing personal data are derived from the sources by the domains derivableBy. This contains relation also documents the reason why the derived personal data are available at the domains that can derive the personal data (lines 31-35).

If the derived personal data are newly identified (positive answer to question DF4), i.e., no instance of the class RelatedTo (see Figure 10.2) exists to document that the data are personal data of the data subject that is currently considered, then such an instance needs to be created using the attributes values for linkability and sensitive elicited by questions DF4.1 and DF4.2, the collection method derived, and the currently considered statement as origin (lines 36-43).

11.3.2.5. Add Link Between Personal data

For each pair of personal data available at an output domain that shall be linkable due to the statement under consideration, a respective LinkAvailableAt object is created by the operation addPersonalDataLink. This operation is specified in Listing 11.7 and consumes the answers to the questions listed in Table 11.2 as parameters.

To create a link at an output domain, the personal data pd1 and pd2 have to be available at it with the same availability as the link shall be created for. For example, it makes no sense to create a link with availability all when the data shall only be available to individuals and not to all instances of the domain.

The postcondition of the operation addPersonalDataLink is that an instance of the class LinkAvailableAt is created for the output domain, a link containing the personal data pd1 and pd2, the linkability, amount, availability and duration specified by the respective parameters and elicited by questions LF1-LF4, and the statement under consideration as origin (lines 11-16). Additionally, the purpose of all LinkAvailableAt relations for the same personal data shall be updated with the currently considered statement (lines 17-19).

11.3.3. Determine New Statements to be Considered

After the analysis team identified and modeled all flows of personal data from the input domains to the output domains of the statement under consideration, it has to be determined whether new statements have to be added to the set of statements toBeConsidered (see Figure 11.2). For this, it has to be checked at which output domains new personal data are available due to the statement under consideration. All statements that have an output domain with new personal data available to it as input domain, are added to the set of statements that need to be considered. The currently considered statement is then removed from the statements to be considered. This task can be automated by the operation updateStatementsToBeConsidered specified in Listing 11.8.

11.3.4. Application to EHS

For the sake of simplicity, I only show one example for each kind of personal data flow in the EHS example. Figure 11.7 shows the view for the considered requirement R6 as first step of the

Listing 11.6: *OCL specification of the operation* addDerivedPersonalDataFlow

```
context DataFlowAnalysisGraph::addDerivedPersonalDataFlow(
  outputDomain : Domain, name : String, sources : Set(Data),
  derivableBy : Set(Domain), amount : RecordAmount,
  availability : AvailabilityDegree, duration : Duration,
  linkability : Linkability, sensitive : Boolean)
pre:
outputDomains→includes(outputDomain) and
sources→forAll(source|
  inputDomains.pdavailableats→exists(pa|pa.personalData = source)) and
derivableBy→forAll(db|
  (self.inputDomains→includes(db) and
  sources→forAll(source|
    db.pdavailableats→exists(pa|pa.personalData = source))) or
  ProblemDiagram.allInstances()→select(pd|
    pd.statements→includes(
      self.underConsideration.statement)).domains→select(d|
      d.oclIsTypeOf(Machine))→includes(db.domain))
post:
outputDomain.pdavailableats→exists(pa|
  pa.personalData.name = name and
  pa.amount = amount and
  pa.availability = availability and
  pa.duration = duration and
  pa.origin→includes(self.underConsideration)) and
derivableBy.pdavailableats→exists(paDerived|
    paDerived.personalData.name = name and
    paDerived.amount = amount and
    paDerived.availability = availability and
    paDerived.duration = duration and
    paDerived.purpose→includes(self.underConsideration)) and
DerivedFrom.allInstances()→exists(df|
  df.sources = sources and
  df.derived.name = name and
  df.derivableBy = derivableBy and
  df.origin→includes(self.underConsideration)) and
(not self.dataSubject.relatedtos@pre→exists(rt|
  rt.personalData.name = name) implies
  self.dataSubject.relatedtos→exists(rt|
    rt.personalData.name = name and
    rt.linkability = linkability and
    rt.sensitive = sensitive and
    rt.collection = CollectionMethod::derived and
    rt.origin→includes(self.underConsideration)))
```

data flow analysis. The figure shows that initially the personal data vitalSigns, patientDetails, and healthStatus are available at the input domain Patient (cf. Figure 11.5).

To give a meaning to the literals of the enumerations RecordAmount and Linkability in the context of the EHS, I define the literals small/smallGroup, medium/mediumGroup, and large/largeGroup to 2-10 records/patients, 11-100 records/patients, and more than 100 records/patients, respectively.

Table 11.3 on page 189 shows the filled out questionnaire for the identification of personal data flows considering statement R6 and the output domain EHR. The answers show that the vitalSigns available at the Patient shall flow to the domain EHR. The vital signs shall be available at all possible instances of the domain EHR, all vital signs processed by the system-to-be shall be stored at it, and it shall be available there until the vital signs have to be deleted. Note that

Listing 11.7: *OCL specification of the operation addPersonalDataLink*

```
context DataFlowAnalysisGraph::addPersonalDataLink(outputDomain : Domain,
  pd1 : Data, pd2 : Data, linkability : Linkability,
  amount : RecordAmount, availability : AvailabilityDegree,
  duration : Duration)
pre:
outputDomains→includes(outputDomain) and
outputDomain.pdavailableats→exists(pa1,pa2|
  pa1.personalData = pd1 and pa1.availability = availability and
  pa2.personalData = pd2 and pa2.availability = availability)
post:
outputDomain.linkavailableats→exists(la|
  la.link.data = Set{pd1,pd2} and
  la.amount = amount and
  la.availability = availability and
  la.duration = duration and
  la.origin→includes(self.underConsideration)) and
inputDomains.linkavailableats→forAll(la|
  la.link.data = Set{pd1,pd2} implies
  la.purpose→includes(self.underConsideration))
```

Listing 11.8: *OCL specification of the operation updateStatementsToBeConsidered*

```
context DataFlowAnalysisGraph::updateStatementsToBeConsidered()
pre: not self.underConsideration.oclIsUndefined()
post:
self.underConsideration.oclIsUndefined() and
self@pre.outputDomains→select(outputDomain|
  outputDomain.pdavailableats→excludingAll(
    outputDomain.pdavailableats@pre).notEmpty())→forAll(outputDomain|
      self.toBeConsidered→includesAll(
        self.inputedges→select(ie| ie.domain = outputDomain).statement)) and
self.toBeConsidered→excludes(self@pre.underConsideration)
```

questions DF2 and DF3, including their subquestions, do not have to be answered, because DF1 is positively answered and a data flow only belongs to one of the three previously mentioned categories. The result of the execution of the operation addPersonalDataFlow (see Listing 11.4) using the respective parameters is shown in Figure 11.8.

After the consideration of R6, new personal data are available at the domain EHR and hence, all statements for which the EHR is an input domain, are added to the statements to be considered. These are R2, R3, R5 and R7 (cf. Figure 11.4), while R6 is removed from the set of statements that have to be considered (cf. Listing 11.8).

As next statement, I consider F1. Figure 11.9 shows the view for the considered fact F1. Due to F1, the patientDetails, vitalSigns, and healthStatus are all available at the domain EHR leading to answers to questionnaire Table 11.1 similar to Table 11.3. Additionally, I identified using the questionnaire shown in Table 11.2 that the personal data patientDetails shall be linkable to the personal data healthStatus. Note that consequently the personal data vitalSigns can also be linked to the patientDetails, because the vitalSigns are contained in the healthStatus. The answers to the questionnaire for Tuple{pd1 = healthStatus, pd2 = patientDetails} are given in Table 11.4 on page 189. The result of applying the operations addPersonalDataFlow and addPersonalDataLink for the mentioned personal data and the output domain EHR is shown in Figure 11.10 on page 192. No new statements are added to the statements to be considered, because all statements for which the EHR is an input domain are already in the set of statements

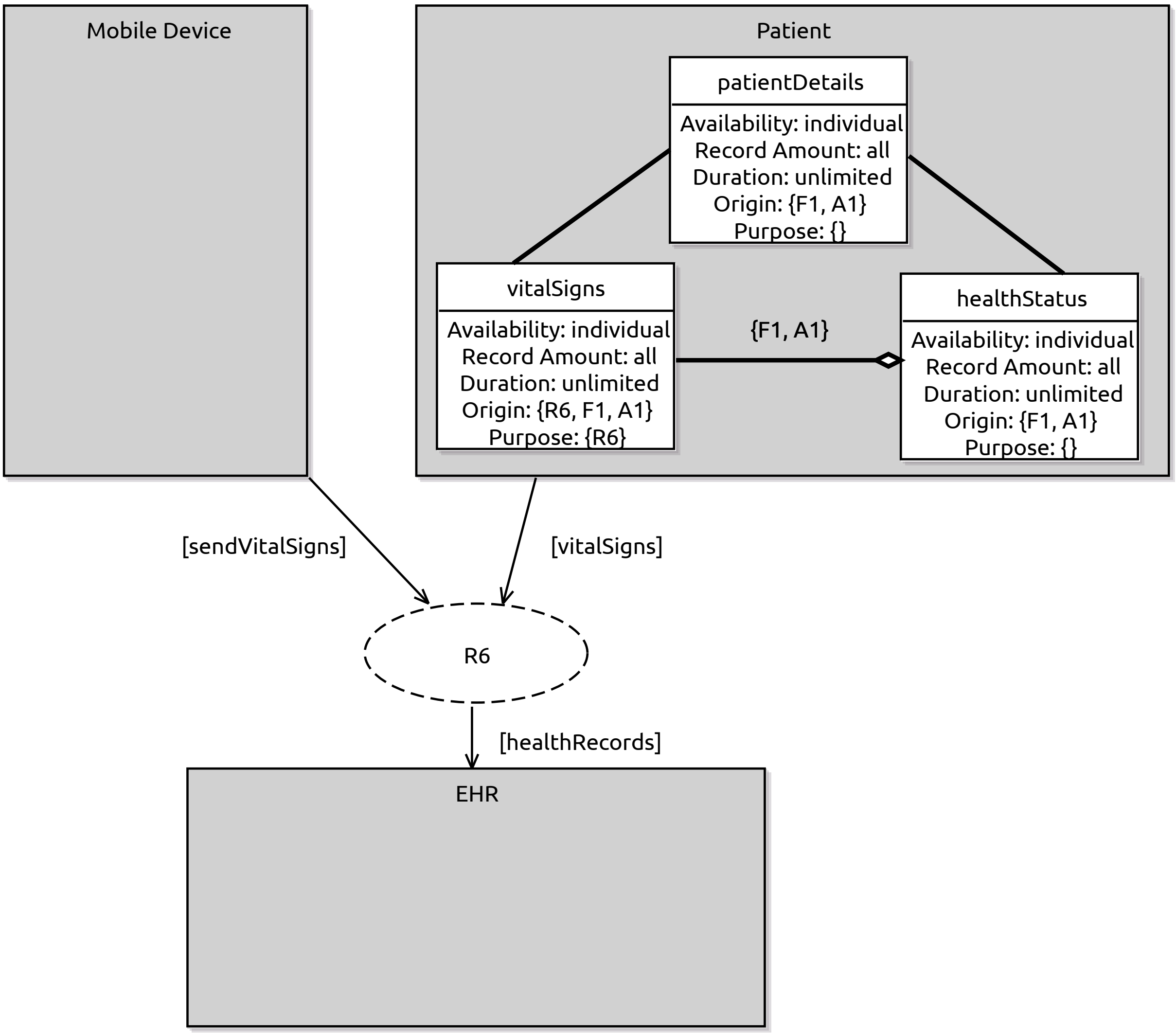


Figure 11.7.: *View for the considered requirement R6 and the personal data available at the input and output domains before a flow was added*

to be considered.

To keep the application to the EHS short, I omit a detailed consideration of the statements A1, A2, A3, R1, and R2 (cf. Figure 11.4). As next statement, I consider the functional requirement R3. Figure 11.11 on page 193 shows the initial statement consideration diagram for R3. It shows which personal data are available and linkable at the input domains EHR and Doctor. As none of the personal data available at the EHR and Doctor necessarily need to flow to the output domains Insurance Application and Invoice, I needed to identify the contained and derived personal data that are necessary to be available at these domains.

I identified that the personal data treatments and diagnoses contained in the personal data healthStatus, and the personal data insuranceNumber contained in the personal data patientDetails flow to the Insurance Application. Additionally, the personal data treatments, diagnoses, and insuranceNumber shall all be linkable to each other with the linkability single. That is, the information which treatments, diagnoses, and insuranceNumber belong to each other is available at the Insurance Application. Table 11.5 on page 195 shows the answers to the questionnaire for the identification of personal data flows for the personal data insuranceNumber flowing to the Insurance Application. The insuranceNumber is identified to be contained in the patientDetails, to identify a single patient, to be non-sensitive personal data, to be available in a large amount

Table 11.3.: *Answers to the questionnaire for the identification of personal data flows due to R6*

No.	Question	Answer
DF	Which personal data necessarily need to be provided to the output domain?	vitalSigns
DF1	Are the personal data available at an input domain?	true
DF2	Are the personal data contained in personal data available at an input domain?	-
DF2.1	In which available personal data are the flowing personal data contained?	-
DF3	Can the personal data be derived from personal data available at the input domains?	-
DF3.1	From which available personal data can the flowing personal data be derived?	-
DF3.2	Which input domains are able to derive the personal data?	-
DF3.3	Does the machine involved in the statement derive the personal data?	-
DF4	Are the flowing data already modeled as personal data of the data subject?	true
DF4.1	To which degree are the flowing personal data linkable to the data subject?	-
DF4.2	Are the flowing personal data sensitive personal data for the data subject?	-
DF5	What is the minimal amount of personal data records that needs to be available at the output domain?	all
DF6	Which is the smallest group of instances of the output domain to which the personal data need to be available?	all
DF7	How long is it necessary to retain the personal data at the output domain?	untilDeleted

Table 11.4.: *Answers to the questionnaire for the identification of a link between personal data at an output domain due to F1*

No.	Question	Answer
LF	Which personal data need to be linkable at the output domain?	Tuple{pd1=healthStatus, pd2=patientDetails}
LF1	To which degree have the personal data to be linkable to each other?	single
LF2	What is the minimal amount of links between personal data records that need to be available at the output domain?	all
LF3	Which is the smallest group of instances of the output domain to which the links between the personal data need to be available?	all
LF4	How long is it necessary to retain the links between the personal data at the output domain?	untilDeleted

(more than 100 records) at all instances of Insurance Application and the insuranceNumber shall be available there until it is necessary to delete these personal data. I omit the answers for the other contained personal data flowing to an output domain and the links between the personal

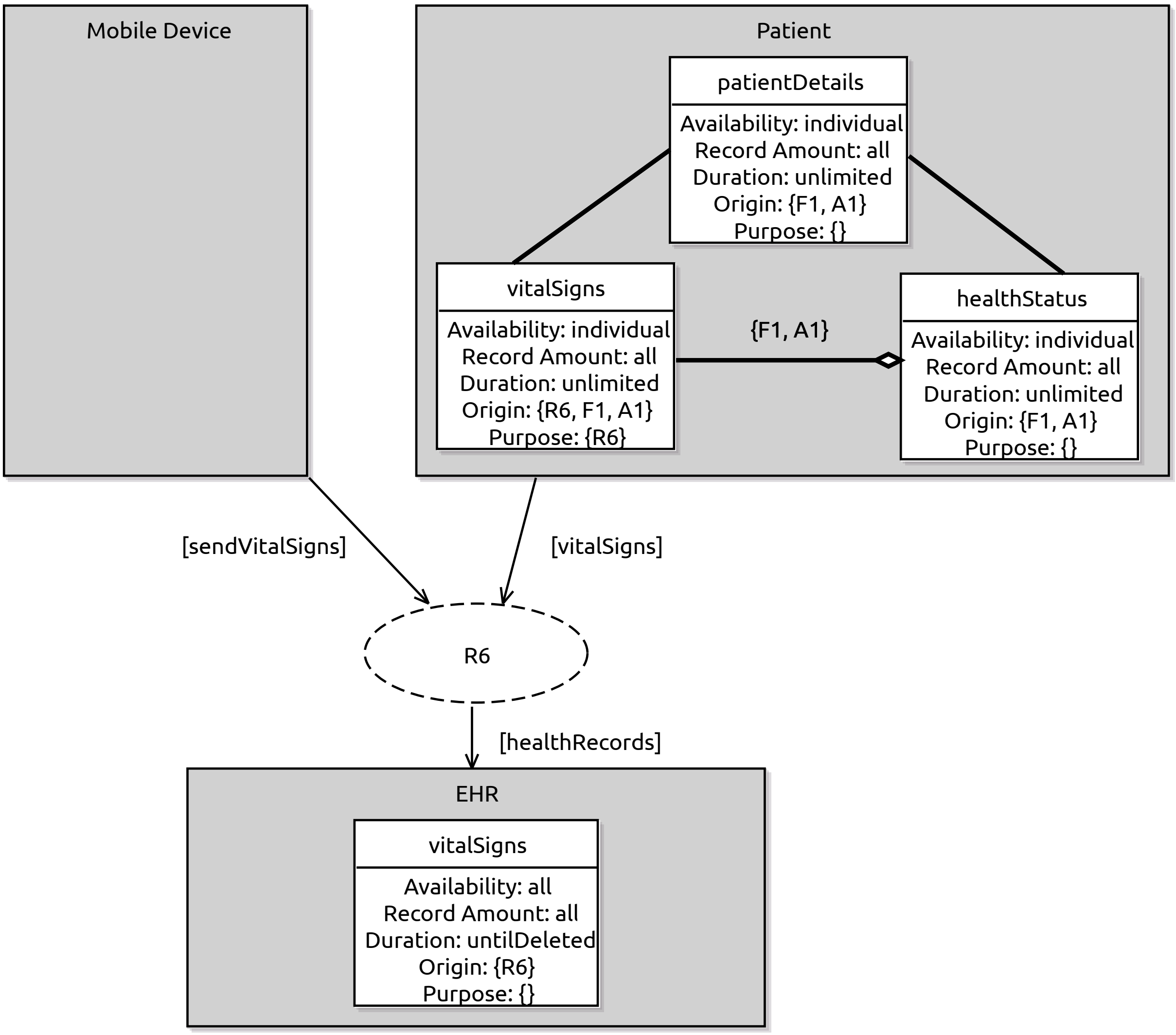


Figure 11.8.: *View for the considered requirement R6 and the personal data available at the input and output domains after a flow was added*

data available at the output domains for the sake of simplicity.

For the output domain Invoice, I identified that the personal data treatments contained in the personal data healthStatus, the personal data patientBillingDetails contained in the personal data patientDetails, and the personal data treatmentCosts derivable from the personal data healthStatus and patientDetails by the input domain Doctor flow to it. Additionally, the personal data treatments, treatmentCosts, and patientBillingDetails shall all be linkable to each other with the linkability single. That is, the information which treatments, treatmentCosts, and patientBillingDetails belong to each other is available at the lexical domain Invoice. Table 11.6 on page 196 shows the answers to the questionnaire for the identification of personal data flows for the derived personal data treatmentCosts. The treatmentCosts themselves can only be linked to a largeGroup of potential patients they may be related to, they are considered as sensitive personal data, and they shall be available at all instances of the domain Invoice in a large amount untilDeleted. I omit the answers for the other contained personal data flowing to an output domain and the links between the personal data available at the output domains for the sake of simplicity.

The result of the consideration of requirement R3 is shown in Figure 11.12 on page 194. It shows which personal data with which attributes shall be available at the output domains

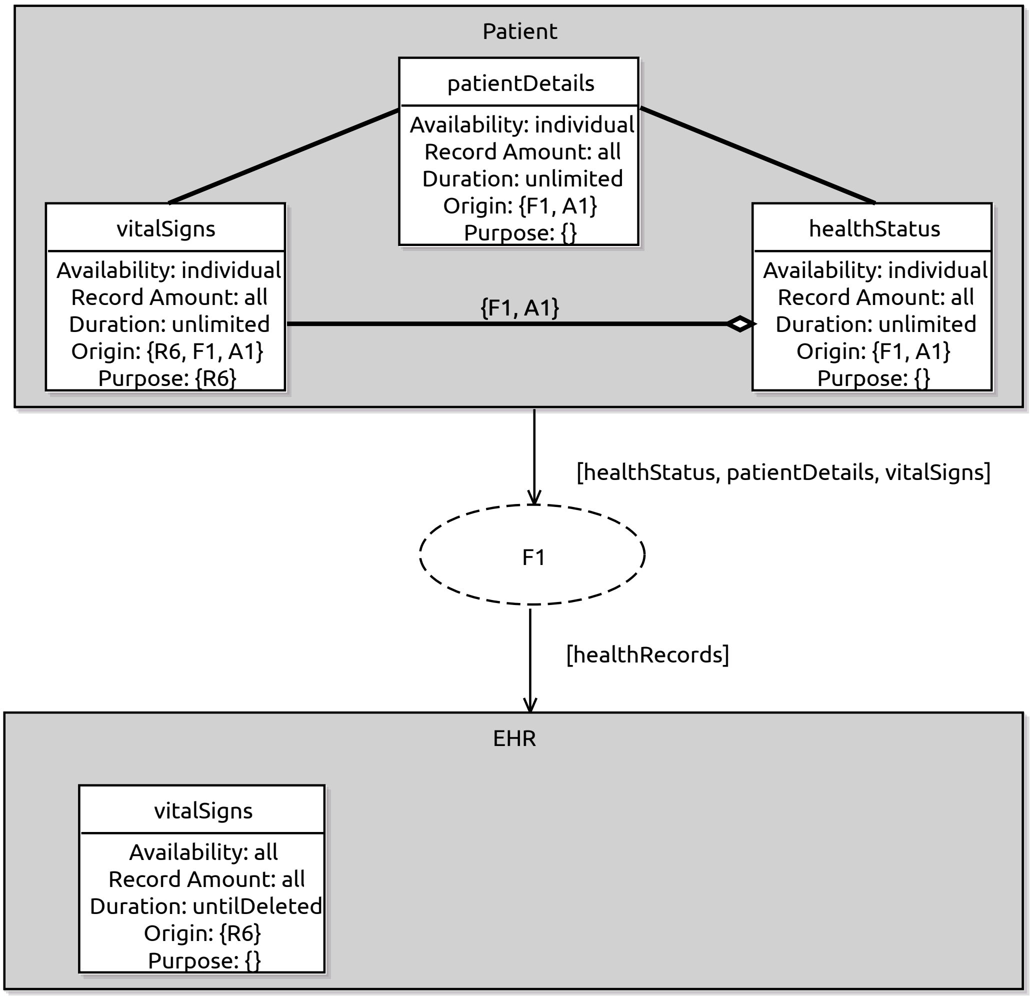


Figure 11.9.: *View for the considered fact F1 and the personal data available at the input and output domains before a flow was added*

Insurance Application and Invoice. Additionally, Figure 11.12 shows that the newly identified contained personal data patientBillingDetails, insuranceNumber, diagnoses, and treatments are available at the input domains EHR and Doctor, because at these the personal data patientDetails and healthStatus are available that contain the aforementioned personal data. Furthermore, the personal data treatmentCosts are modeled to be available at the input domain Doctor, because doctors shall be able to derive the treatmentCosts from the personal data patientDetails and healthStatus (visualized by a dependency starting from the derived data and pointing to the sources).

The substep Identify personal data flow from a statement is iteratively applied until the set of statements to be considered is empty. The identification of flows of the data subject's personal data for the other statements is done analogously to the shown considerations of the statements R6, F1, and R3.

The final personal data diagram for the data subject Patient is shown in Figure 11.13 on page 197. In comparison to the initial personal data flow diagram (see Figure 10.5 on page 161) created during the step Privacy Threshold Analysis (see Chapter 10), the final personal data diagram is extended with the personal data contained in and derived from the initially identified personal data during the data flow analysis. As for the initially identified personal data, the final personal data diagram also shows whether the contained or derived personal data are sensitive

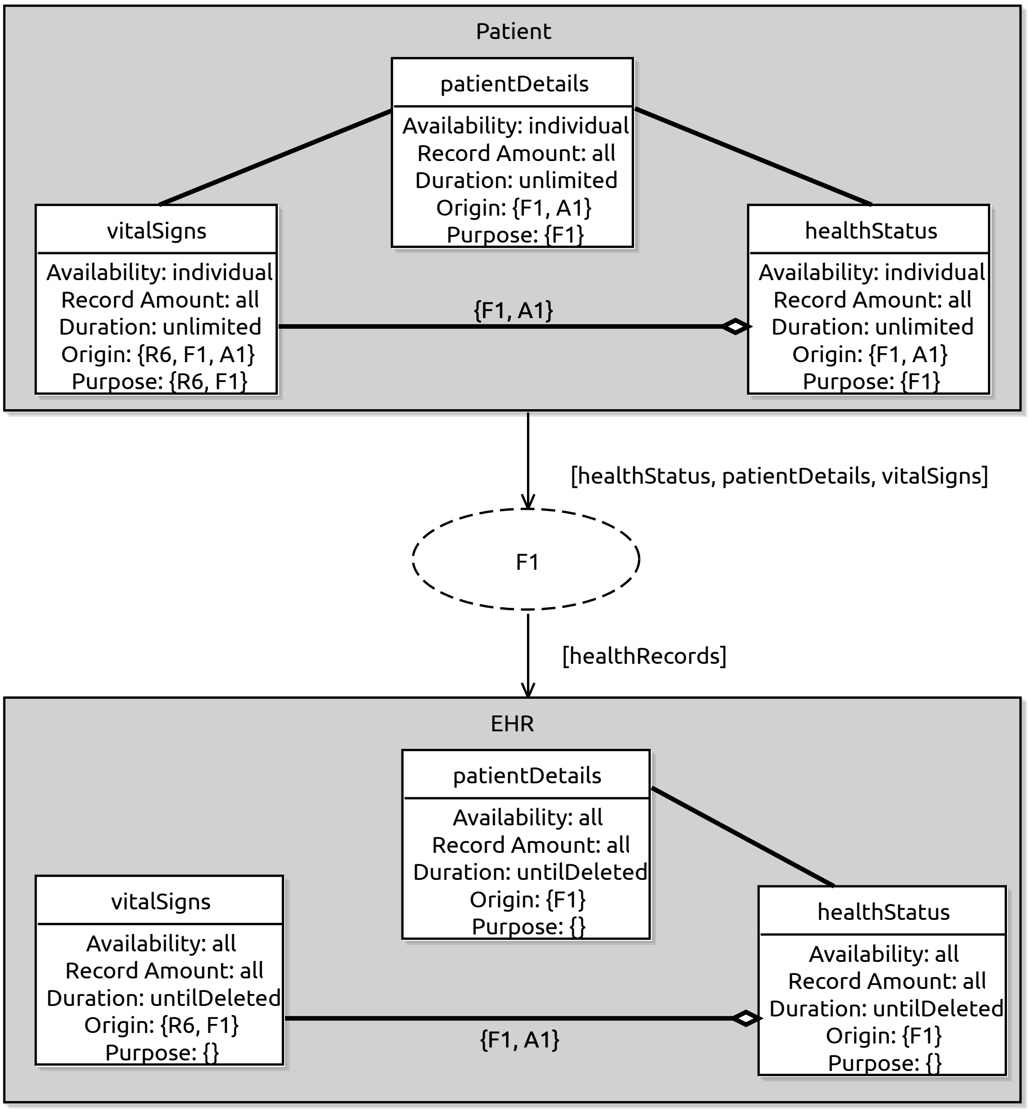


Figure 11.10.: *View for the considered fact F1 and the personal data available at the input and output domains after the flows were added*

personal data, with which degree the data are linkable to the data subject, how the personal data are collected from the data subject, and from which statement the personal data were identified.

The available data diagram for the lexical domain EHR after the personal data flow analysis for the data subject Patient is shown in Figure 11.14 on page 198. Such a diagram can be created for each domain in the ProPAn model and it visualizes which personal data are available at the domain and which of the personal data can be linked to each other at the domain. At the lexical domain EHR, the personal data patientDetails and healthStatus of the data subject patient are available, and all personal data of the patient contained in these two (cf. Figure 11.13). The personal data treatmentCosts and clinicalResearchData are not available at the lexical domain EHR, because the EHR is not able to derive these data, even though the needed data are available at it. The available data diagram in Figure 11.14 also shows that the personal data patientDetails and healthStatus are linkable to each other with linkability single at the domain EHR. That is, the information which patientDetails belong to which healthStatus is available at the EHR. Consequently, the personal data contained in the patientDetails and healthStatus can also be

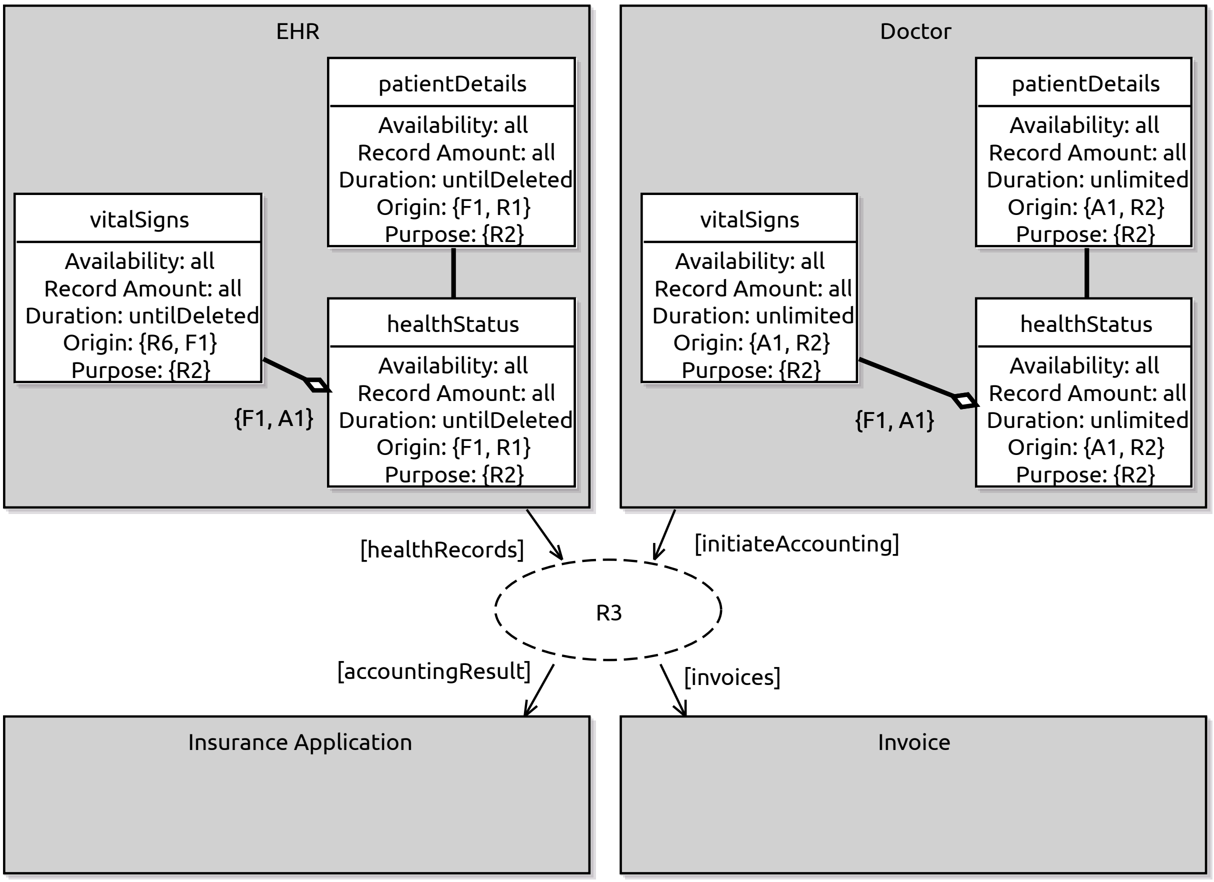


Figure 11.11.: *View for the considered requirement R3 and the personal data available at the input and output domains before a flow was added*

linked to each other.

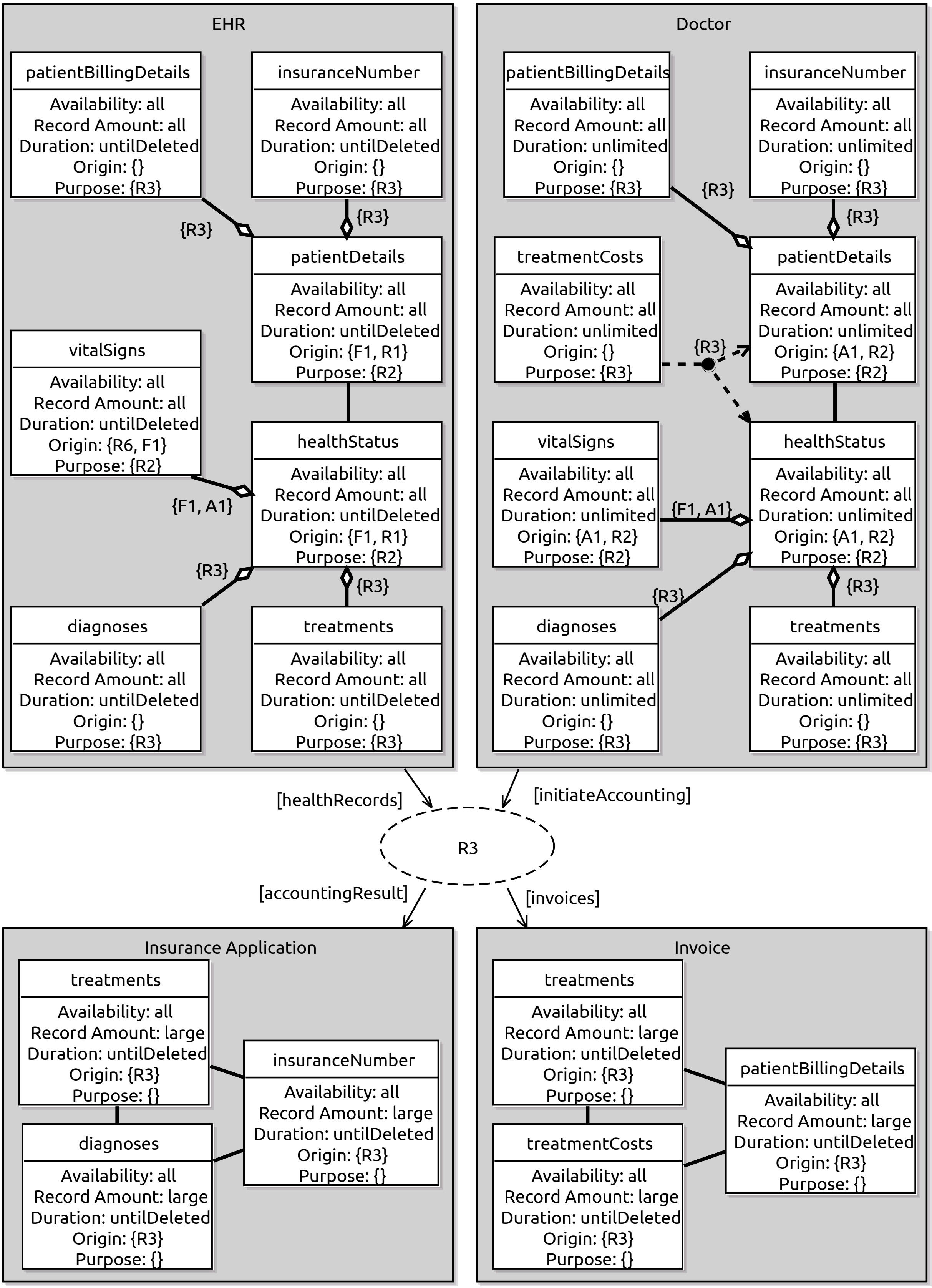


Figure 11.12.: *View for the considered requirement R3 and the personal data available at the input and output domains after the flows were added*

Table 11.5.: *Answers to the questionnaire for the identification of personal data flows due to R3 for the domain Insurance Application and the personal data insuranceNumber*

No.	Question	Answer
DF	Which personal data necessarily need to be provided to the output domain?	insuranceNumber
DF1	Are the personal data available at an input domain?	false
DF2	Are the personal data contained in personal data available at an input domain?	true
DF2.1	In which available personal data are the flowing personal data contained?	patientDetails
DF3	Can the personal data be derived from personal data available at the input domains?	-
DF3.1	From which available personal data can the flowing personal data be derived?	-
DF3.2	Which input domains are able to derive the personal data?	-
DF3.3	Does the machine involved in the statement derive the personal data?	-
DF4	Are the flowing data already modeled as personal data of the data subject?	false
DF4.1	To which degree are the flowing personal data linkable to the data subject?	single
DF4.2	Are the flowing personal data sensitive personal data for the data subject?	false
DF5	What is the minimal amount of personal data records that needs to be available at the output domain?	large
DF6	Which is the smallest group of instances of the output domain to which the personal data need to be available?	all
DF7	How long is it necessary to retain the personal data at the output domain?	untilDeleted

Table 11.6.: *Answers to the questionnaire for the identification of personal data flows due to R3 for the domain Invoice and the personal data treatmentCosts*

No.	Question	Answer
DF	Which personal data necessarily need to be provided to the output domain?	treatmentCosts
DF1	Are the personal data available at an input domain?	false
DF2	Are the personal data contained in personal data available at an input domain?	false
DF2.1	In which available personal data are the flowing personal data contained?	-
DF3	Can the personal data be derived from personal data available at the input domains?	true
DF3.1	From which available personal data can the flowing personal data be derived?	Set{healthStatus, patientDetails}
DF3.2	Which input domains are able to derive the personal data?	Set{Doctor}
DF3.3	Does the machine involved in the statement derive the personal data?	false
DF4	Are the flowing data already modeled as personal data of the data subject?	false
DF4.1	To which degree are the flowing personal data linkable to the data subject?	largeGroup
DF4.2	Are the flowing personal data sensitive personal data for the data subject?	true
DF5	What is the minimal amount of personal data records that needs to be available at the output domain?	large
DF6	Which is the smallest group of instances of the output domain to which the personal data need to be available?	all
DF7	How long is it necessary to retain the personal data at the output domain?	untilDeleted

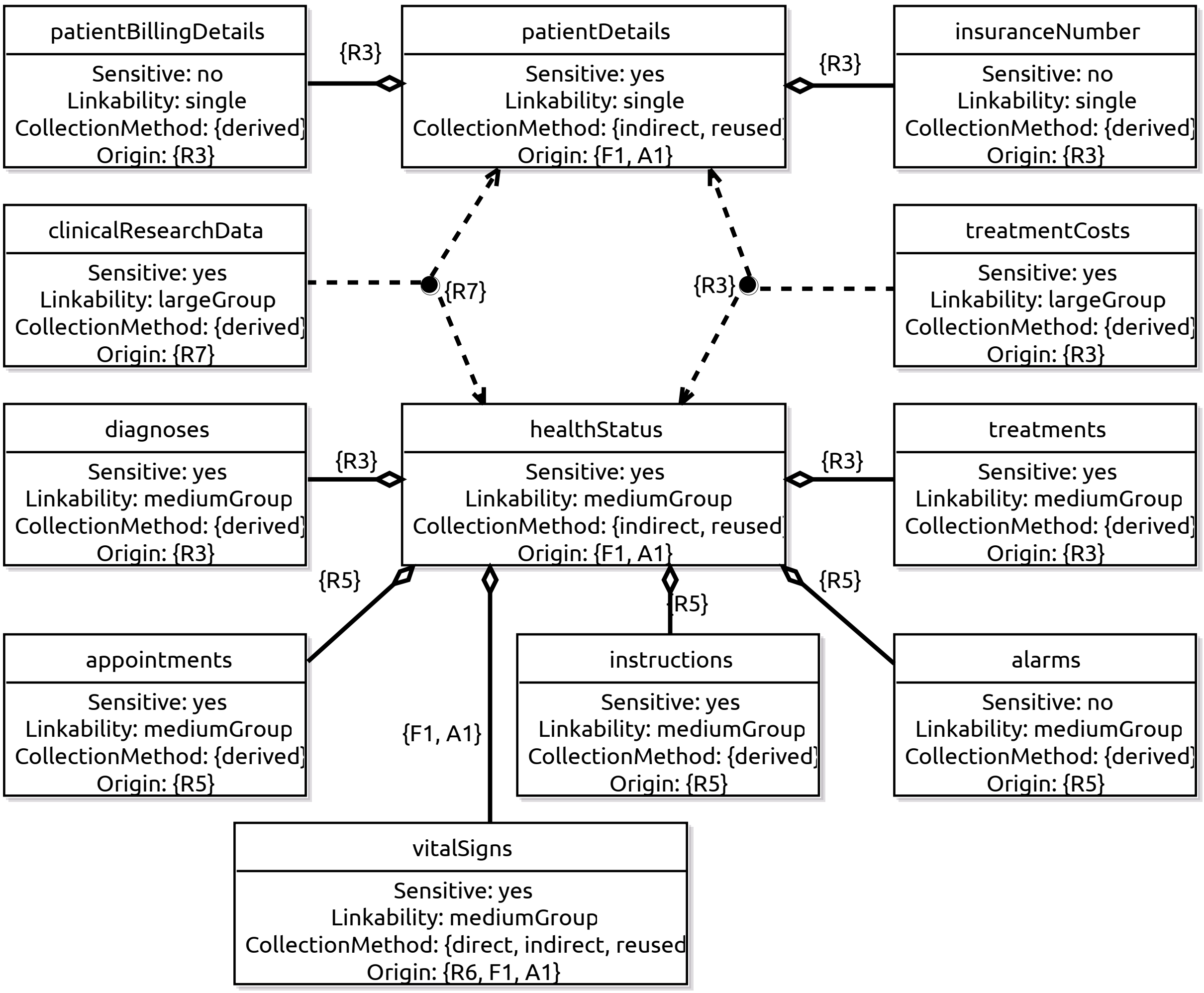


Figure 11.13.: *The final personal data diagram for the data subject* Patient *showing all personal data identified during the privacy threshold analysis and the personal data flow analysis*

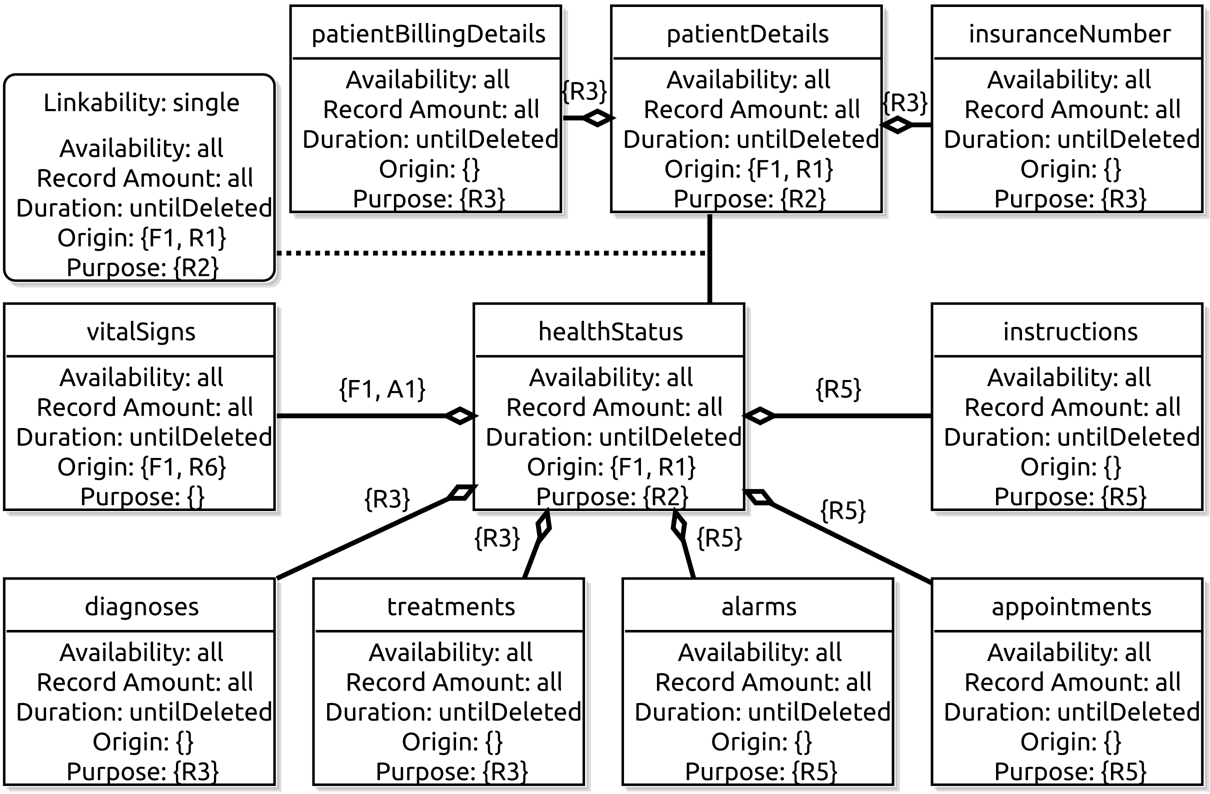


Figure 11.14.: *The available data diagram for the domain EHR showing all personal data identified to be available at it during personal data flow analysis for the data subject Patient*

11.4. Generate Final Data Flow Graphs

Based on the elicited flows of personal data and the information about the availability of the personal data at the domains of the system, the ProPAn tool can generate a so-called *final data flow graph* for each data subject. These graphs refine the initial data flow graphs that were generated in the step Privacy Threshold Analysis (see Section 10.4) and represent the intended and expected flows of personal data due to the statements contained in the problem frame model.

In Section 11.4.1, I show the extension of the already introduced metamodel for data flow graphs. This extension adds edges that allow to represent flows of derived personal data that are not available at any input domain of the statements from which the flow was identified. I explain how the ProPAn tool generates the final data flow graphs in Section 11.4.2. Section 11.4.3 presents the final data flow graph for the data subject patient in the EHS example.

11.4.1. Metamodel Extension for Data Flow Graphs

The final data flow graphs are, as the initial data flow graphs, instances of the class DataFlowGraph. This means, that a Person object may have two data flow graphs assigned to it, an initial and a final one. I already introduced the metamodel for the initial data flow graphs in Section 10.4. In comparison to an initial data flow graph (see Figure 10.7 on page 164), a final data flow graph (see Figure 11.15) may also contain instances of the class DerivedFlowEdge. A DerivedFlowEdge is created if personal data that are derived from other personal data flow to an output domain, and the derived personal data are not derivable by any of the input domains, i.e., the derived personal data flowing to the output domain are not available at one of the input domains. In these cases the machine is responsible to derive the flowing personal data. I make this explicit using DerivedFlowEdge instances in the final data flow graphs.

A DerivedFlowEdge has in addition to a DataFlowEdge the DerivedFrom relations associated to it that describe from which personal data and by which domain the flowing personal data can be derived.

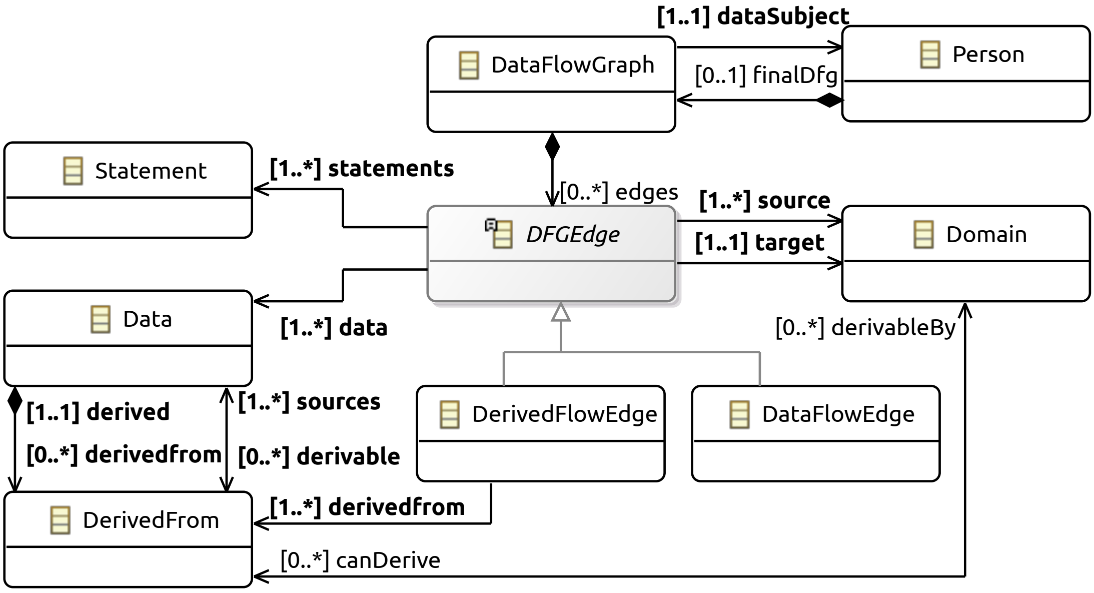


Figure 11.15.: *The metamodel for the final data flow graph*

Listing 11.9: *OCL specification of the operation* generateFinalDataFlowGraph

```
1  context Person::generateFinalDataFlowGraph()
2  pre: true
3  post:
4  self.finalDfg.edges→forAll(e1,e2|
5    (e1.source = e2.source and e1.target = e2.target and e1 <> e2) implies
6      (e1.oclIsTypeOf(DerivedFlowEdge) and e2.oclIsTypeOf(DataFlowEdge) or
7       e2.oclIsTypeOf(DerivedFlowEdge) and e1.oclIsTypeOf(DataFlowEdge))) and
8  self.relatedtos.personalData.pdavailableats→forAll(paSource,paTarget|
9    (paSource.domain <> paTarget.domain and
10     paSource.personalData = paTarget.personalData and
11     paTarget.origin→intersection(paSource.purpose)→notEmpty()) implies
12   (self.finalDfg.edges→exists(e|
13     e.oclIsTypeOf(DataFlowEdge) and
14     e.source = paSource.domain and e.target = paTarget.domain and
15     e.data→includes(paSource.data) and
16     e.statements→includesAll(
17       paTarget.origin→intersection(paSource.purpose))))) and
18 self.relatedtos.personalData.pdavailableats→forAll(pa|
19   (not self.finalDfg.edges→exists(e|
20     e.oclIsTypeOf(DataFlowEdge) and
21     e.target = pa.domain and
22     e.data→includes(pa.personalData) and
23     e.statements→includesAll(pa.origin))) implies
24   (pa.personalData.derivedfrom→forAll(df|
25     df.origin→intersection(pa.origin)→notEmpty() implies
26     self.finalDfg.edges→exists(e|
27       e.oclIsTypeOf(DerivedFlowEdge) and
28       e.target = pa.domain and
29       e.source = df.sources.pdavailableats.domains→select(d|
30         e.statements→exists(s|
31           s.statement.refersTo→includes(d.domain))) and
32       e.data→includes(pa.personalData) and
33       e.statements→includesAll(df.origin→intersection(pa.origin)) and
34       e.derivedFrom→includes(df)))))
```

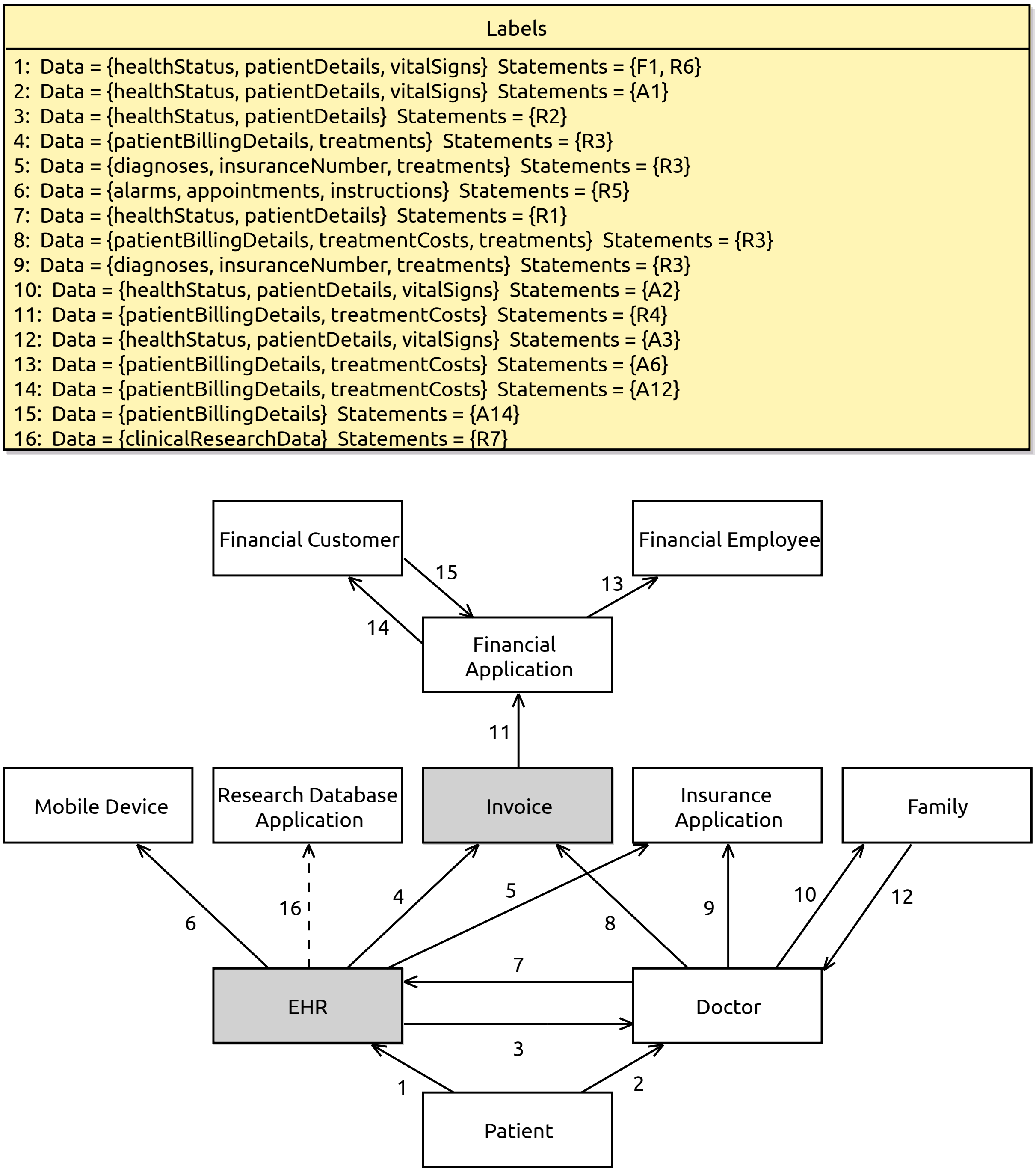
11.4.2. Generation of a Final Data Flow Graph

The generation of the final data flow graph is performed by the operation generateFinalDataFlowGraph that is specified in Listing 11.9. The postcondition of the operation generateFinalDataFlowGraph consists of three conditions that all have to be satisfied.

First, for each type of edge (data flow or derived flow edge) and pair of domains, at most one edge shall exist that connects the two domains (lines 4-8). That means, an edge from a source domain to a target domain aggregates all data flows from the source to the target.

Second, a data flow edge shall exist between domains *source* and *target* if these are not the same, personal data *pd* are available at both domains, and the intersection of the purpose why the *pd* are available at *source* and the origin from which it was identified that *pd* are available at *target* is not empty. The personal data *pd* and the intersection of the purpose and origin shall be contained in the attributes data and statements, respectively (lines 8-17).

Third, there may be personal data available at a domain for which no data flow edge exists (lines 18-23), because the personal data are derived from other personal data and are not derivable by one of the input domains of the statement from which the flow of personal data was identified. For all instances of the class DerivedFrom specifying that such personal data can be derived from other personal data, a derived data flow edge shall exist. The target of the derived data flow edge is the domain at which the derived personal data are available and the source are



Labels

1: Data = {healthStatus, patientDetails, vitalSigns} Statements = {F1, R6}
2: Data = {healthStatus, patientDetails, vitalSigns} Statements = {A1}
3: Data = {healthStatus, patientDetails} Statements = {R2}
4: Data = {patientBillingDetails, treatments} Statements = {R3}
5: Data = {diagnoses, insuranceNumber, treatments} Statements = {R3}
6: Data = {alarms, appointments, instructions} Statements = {R5}
7: Data = {healthStatus, patientDetails} Statements = {R1}
8: Data = {patientBillingDetails, treatmentCosts, treatments} Statements = {R3}
9: Data = {diagnoses, insuranceNumber, treatments} Statements = {R3}
10: Data = {healthStatus, patientDetails, vitalSigns} Statements = {A2}
11: Data = {patientBillingDetails, treatmentCosts} Statements = {R4}
12: Data = {healthStatus, patientDetails, vitalSigns} Statements = {A3}
13: Data = {patientBillingDetails, treatmentCosts} Statements = {A6}
14: Data = {patientBillingDetails, treatmentCosts} Statements = {A12}
15: Data = {patientBillingDetails} Statements = {A14}
16: Data = {clinicalResearchData} Statements = {R7}

Figure 11.16.: *The final data flow graph for the data subject* Patient *generated by the ProPAn tool*

all domains at which personal data needed to derive the personal data are available and that are input domains of a statement from which the derived flow was identified (lines 24-34).

11.4.3. Application to EHS

The final data flow graph for the data subject Patient is shown in Figure 11.16. The graph visualizes the elicited flows of personal data between the domains of the system. Data flow edges and derived flow edges are presented as solid and dashed arrows starting from their source and pointing to their target, respectively. For the sake of readability, the edges are annotated with numbers, which can be found in the Labels box. This box provides for each number the personal data which flow between the source and target domains of the edge and from which statements these flows were identified. For example, I identified from statements F1 and R6 (cf. Section 11.3.4) that the personal data healthStatus, patientDetails, and vitalSigns flow from the data subject Patient to the lexical domain EHR (data flow edge with label 1). Figure 11.16 contains only one derived data flow (edge with label 16) representing the flow of the personal data clinicalResearchData from the lexical domain EHR to the domain Research Database Application due to requirement R7. The personal data clinicalResearchData are derived by the machine from the personal data healthStatus and patientDetails to satisfy requirement R7 (cf. Figure 11.13), but the personal data clinicalResearchData are not available at the lexical domain EHR (cf. Figure 11.14). Note that the personal data treatmentCosts are also derived personal data, but this flow is represented by a data flow edge, because the treatmentCosts are derivable by the domain Doctor. Hence, the treatmentCosts are available at the domain Doctor and the flow of the treatmentCosts to the lexical domain Invoice is presented by the data flow edge with label 8.

In comparison to the initial data flow graph for the data subject Patient (see Figure 10.9 on page 166), the final data flow graph only contains the flows of personal data between domains that were elicited as necessary to satisfy the requirements and that are expected to exist due to the domain knowledge. For example, I identified during the data flow analysis that no personal data shall flow to the indirect counterstakeholder Hacker, and that no personal data of the patient are expected to flow from the Financial Employee to the Financial Application.

11.5. Comparison to the State of the Art

In this section, I discuss briefly the state of the art methods that I introduced in Chapter 3 and that explicitly elicit how a system processes personal data, including the identification of personal data flows in a system. Several methods have steps in which the processing of personal data is explicitly documented or modeled (Antignac et al., 2016; Yee, 2017; Oliver, 2016; De and Le Métayer, 2016; Ahmadian and Jürjens, 2016; van Blarkom et al., 2003; Crespo et al., 2015; Drgon et al., 2016).

Antignac et al. (2016), Yee (2017), and Oliver (2016) propose to add annotations to DFDs that indicate the processing of personal data. However, the authors do not provide support to identify where and how personal data are processed. My final data flow graphs that refine the initial data flow graphs are comparable to the annotated DFDs proposed by Antignac et al., Yee, and Oliver.

De and Le Métayer (2016) collect the personal data processed by a system using a template summarizing seven attributes (form, precision, volume, purpose, retention, visibility, and intervenability). All these attributes, except intervenability, are also collected in this step of the ProPAn method. In contrast to De and Le Métayer, I elicit more fine grained information, i.e., for each domain at which the personal data are available. In the step Privacy Requirements Identification (see Chapter 12) the attribute intervenability is reflected in the form of privacy requirements. De and Le Métayer provide no support for the elicitation of the personal data.

Ahmadian and Jürjens (2016) propose to annotate UML diagrams with stereotypes to indicate the processing of personal data. In comparison to ProPAn, the authors do not provide support to systematically elicit how the personal data are processed.

Crespo et al. (2015) and Drgon et al. (2016) describe high-level privacy engineering methods and hence, they do not provide detailed support for identification of the personal data that are processed by the system. However, steps to assess the processing behavior of a system concerning personal data are contained in their methods.

I can conclude that the systematic and tool supported elicitation and modeling of personal data flows through a system, and the availability of personal data and their linkability at domains based on its functional requirements and domain knowledge about the machine's environment are novel contributions to the state of the art. Several state of the art methods use data flow diagrams (DFDs) as system models and starting point for the privacy analysis (Antignac et al., 2016; Yee, 2017; Oliver, 2016; De and Le Métayer, 2016; Knirsch et al., 2015; Deng et al., 2011). However, the authors do not provide much guidance on the creation of the DFDs. In this chapter, I have shown how final data flow graphs that are similar to DFDs can systematically be created based on a given requirements model. This is an additional contribution of the ProPAn method to the state of the art. A novelty of the proposed data flow graphs and all other artifacts created using the ProPAn method is that explicit links to the requirements and system model are created and maintained that allow to trace these artifacts to the requirements and domain knowledge they were identified from. These traceability links make the privacy analysis results more transparent and comprehensible.

11.6. Conclusions

In this chapter, I have shown that a requirements model in problem frame notation enhanced with domain knowledge diagrams elicited during the step Privacy Context Elicitation (see Chapter 9) can be used to systematically identify how a system processes personal data. The analysis starts with considering which personal data identified during the step Privacy Threshold Analysis (see Chapter 10) flow from the data subject to other domains. Then the analysis team iteratively assesses to which domains the personal data further flow. During this analysis the information

1. how the personal data are collected from the data subjects,
2. in which amount the personal data are available at the domains of the system,
3. how long the personal data are retained at the domains,
4. due to which statements the personal data are available at a domain,
5. for which purpose the personal data need to be available at the domain, and
6. which personal data available at a domain need to be linkable to each other

is elicited guided by the *refers to* and *constrains* references of the statements in the problem frame model. Furthermore, I support to model that only contained or derived personal data flow between domains and to refine the personal data processed by the system in this way. The presented final data flow graphs visualize the elicited data flows and are similar to DFDs. In future work, it may be investigated whether these can be used as input for other privacy requirements engineering methods (see Section 11.5).

The next step of the ProPAn method is the Privacy Requirements Identification (see Chapter 12). This step uses the information elicited in the step Data Flow Analysis described in this chapter to derive the privacy requirements that the system-to-be has to satisfy.

Chapter 12

Computer-aided Identification of Privacy Requirements

In this chapter, I provide the details on the step Privacy Requirements Identification of the ProPAn method. This step is concerned with the derivation of privacy requirements from the identified personal data, their availability at the domains of the system, and their flow through the system, which the analysis team elicited and documented in the steps Privacy Threshold Analysis (see Chapter 10) and Data Flow Analysis (see Chapter 11). Similar to the identification of personal data flows, the privacy requirements identification also follows the privacy principles *use limitation*, *storage limitation*, *collection limitation*, *data minimization*, and *fairness* (see Section 2.4.3). That is, in this step privacy requirements are generated that reflect the elicited intended and expected availability of personal data at the domains of the system. These can then be adjusted by the analysis team, and automatically validated for consistency by the ProPAn tool.

The presented generation of the privacy requirements is based on the papers (Meis and Heisel, 2016b, 2017b) of which I am the main author. Maritta Heisel provided valuable feedback to the content of the papers that improved these.

In Section 12.1, I provide an introduction to this chapter. The automatic generation of the privacy requirements is presented in Section 12.2. How the analysis team may adjust the generated privacy requirements is explained in Section 12.3. I discuss how the adjusted requirements are validated for consistency by the ProPAn tool in Section 12.4. In Section 12.5, I compare the step Privacy Requirements Identification with the state of the art methods. This chapter is concluded in Section 12.6.

12.1. Introduction

Identifying the privacy requirements for a system-to-be is one of the main goals of a privacy analysis. These privacy requirements need to be derived from the functional description of the system-to-be, and are the starting point for the identification and assessment of possible threats to them, and the selection of technologies that have to be integrated into the system-to-be to address them.

The review of the state of the art in privacy requirements engineering presented in Chapter 3 has shown that only few methods support the systematic identification of privacy requirements from a system model (cf. Section 12.5). Most methods assume that the analysis team can provide the privacy requirements of the system-to-be as an input. Additionally, no method considers privacy requirements that are as fine-grained as the ones proposed in my privacy requirements taxonomy (see Chapter 7).

In this chapter, I describe a systematic and computer-aided method to generate, adjust, and validate privacy requirements from the information about the availability of personal data at the domains of the system, and their flow through the system elicited during the previous step of the

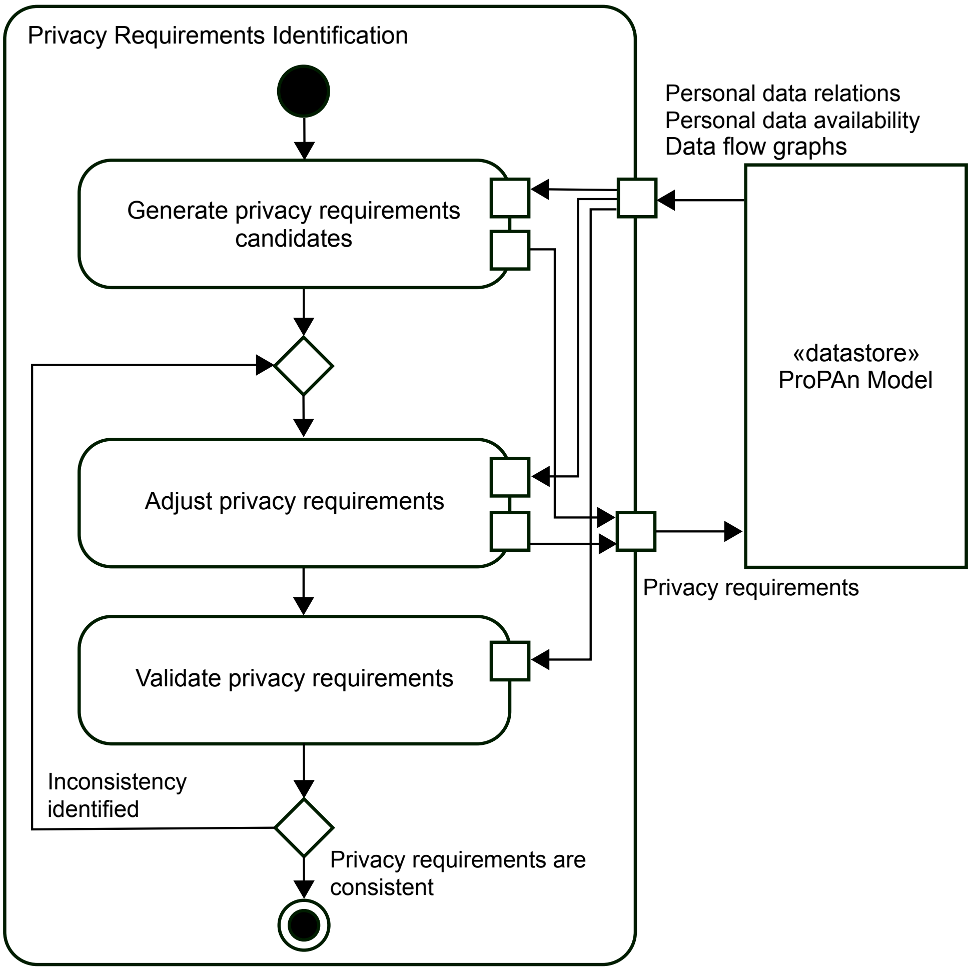

Privacy Requirements Identification
Generate privacy requirements candidates
Adjust privacy requirements
Validate privacy requirements
Inconsistency identified
Privacy requirements are consistent
Personal data relations
Personal data availability
Data flow graphs
«datastore»
ProPAn Model
Privacy requirements

Figure 12.1.: *Detailed view on the step Privacy Requirements Identification of the ProPAn method*

ProPAn method (see Chapter 11). My privacy requirements taxonomy introduced in Chapter 7 defines the privacy requirements that are generated during this step. Figure 12.1 provides a detailed view on the substeps of the step Privacy Requirements Identification. In the first substep called Generate privacy requirements, the ProPAn tool automatically generates privacy requirements based on the documented personal data, availability of the personal data at the domains of the system, and the data flow graphs of the identified data subjects. During the substep Adjust privacy requirements, the analysis team can adjust the generated privacy requirements by adding additional information to complete them, or by weakening or strengthening them. The adjusted privacy requirements are then automatically validated by the ProPAn tool in the step Validate privacy requirements. If inconsistencies are identified in privacy requirements, between them, or between the privacy requirements and the documented information in the ProPAn model, then the analysis team needs to adjust the privacy requirements again. Otherwise, the step Privacy Requirements Identification is finished.

12.2. Generate Privacy Requirements

To automatically generate privacy requirements candidates, I make use of the artifacts elicited in the step Data Flow Analysis and documented in the ProPAn model. I assume that these artifacts reflect the intended processing of personal data that is introduced by the system-to-be and that occurs in its environment. This means, I generate the privacy requirements on the assumption that only the intended processing may be performed and the end-users shall be informed about this processing and shall be able to intervene in this processing. In this context, I aim at eliciting all relevant privacy requirements based on my privacy requirements taxonomy introduced in Chapter 7 also taking into account the EU General Data Protection Regulation (European Commission, 2016). The analysis team can then decide in the next substep of my method (see Section 12.3) to remove, change, and complete the generated privacy requirements if the assumptions made were too strong or too weak.

The ways how I identify the privacy requirements differs for the different kinds of privacy requirements. I present in the following sections the generation rules for the different privacy requirements followed by the (partial) application of these rules on the EHS. The generation of privacy requirements related to the protection goal security is explained in Section 12.2.1. Section 12.2.2 introduces how privacy requirements related to the protection goal unlinkability are generated. I present the generation of transparency requirements in Section 12.2.3, and the generation of intervenability requirements in Section 12.2.4.

12.2.1. Generating Requirements Related to the Protection Goal Security

I introduced the taxonomy of security-related privacy requirements in Section 7.2.2. The generation of data confidentiality requirements is explained in Section 12.2.1.1, and the generation of integrity and availability requirements in Section 12.2.1.2.

12.2.1.1. Generating Data Confidentiality Requirements

During the Data Flow Analysis (see Chapter 11), the analysis team elicits which personal data needs to be available at the domains of the system following the privacy principles *use limitation*, *storage limitation*, *collection limitation*, *data minimization*, and *fairness* (see Section 2.4.3). I use this elicited information to derive data confidentiality requirements for each combination of data subject and person in the ProPAn model at which personal data of the data subject are available. These persons are considered as counterstakeholders. For the personal data that are not available at a counterstakeholder, I generate undetectability requirements to model that the counterstakeholder shall not even know the existence of the personal data. This generation of undetectability requirements is explained in Section 12.2.2.1. The analysis team may decide to weaken the generated undetectability requirements to data confidentiality in substep Adjust privacy requirements (see Section 12.3). I generate for the personal data available at a person data confidentiality requirements that describe that only this availability of the personal data is allowed.

To keep the number of requirements that are generated small, I create for each triple of data subject, counterstakeholder, and availability degree (cf. Figure 7.5 on page 106) at most one data confidentiality requirement containing all personal data of the data subject that shall be kept confidential from the counterstakeholder, while possibly being available to specific instances of the counterstakeholder (specified by the attribute availability). These specific instances are those at which the personal data are modeled to be available. The generation is performed using the operation generateDataConfidentialityRequirements specified in Listing 12.2. To simplify the specification, I introduce two additional operations for the class Domain (see Listing 12.1). The first operation returns all personal data available at the domain that are personal data of the provided dataSubject. The second operation returns only the personal data available at the domain that are personal data of the provided dataSubject and that are available at the domain with availability degree degree.

Application to EHS For the sake of simplicity, I only consider the data subject patient and the counterstakeholder financial employee for the generation of the confidentiality requirements–

Listing 12.1: *OCL specification of additional attributes for the class* Domain

```
context Domain
def: availablePersonalData(dataSubject : Person) : Set(Data) =
  dataSubject.relatedtos.personalData→intersection(
    self.pdavailableats.personalData)
def: availablePersonalData(dataSubject : Person,
    degree : AvailabilityDegree) : Set(Data) =
  dataSubject.relatedtos.personalData→intersection(
    self.pdavailableats→select(pd|pd.available = degree).personalData)
```

Listing 12.2: *OCL specification of additional attributes for the class* Domain

```
context Person::generateDataConfidentialityRequirements()
pre: true
post:
Person.allInstances()→forAll(counterstakeholder|
  Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
      AvailabilityDegree::all}→forAll(ad|
    self.privacyrequirements→select(pr|
      pr.oclIsTypeOf(DataConfidentialityRequirement))→one(sd|
        sd.dataSubject = self and
        sd.counterstakeholders = Set{counterstakeholder} and
        sd.personalData = counterstakeholder.
          availablePersonalData(self, ad).personalData and
        sd.availability = ad and
        sd.repudiation = Repudiation::none)))
```

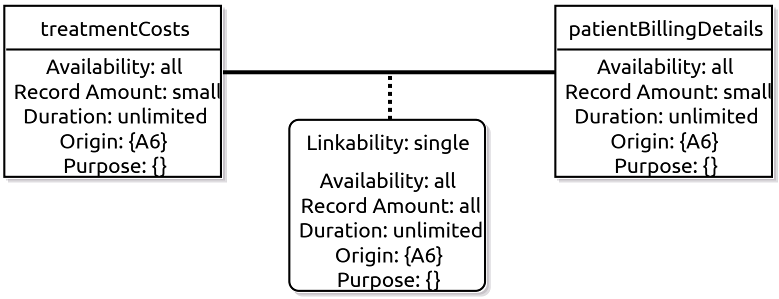
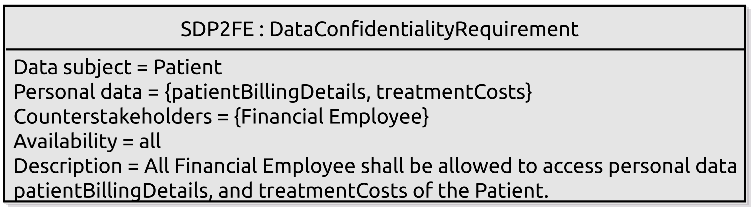


Figure 12.2.: *Available data diagram for the financial employee*

including data confidentiality, data unlinkability, anonymity, and undetectability requirements (cf. Figures 7.2 and 7.5 on pages 101 and 106).

The financial employee's available data diagram is presented in Figure 12.2. The diagram shows that the personal data treatmentCosts and patientBillingDetails shall be available to all financial employees. For these personal data available to the financial employees, the data confidentiality requirement shown in Figure 12.3 is generated. It documents that the treatment costs and patient billing details are allowed to be accessible for all financial employees. The attribute Description presents the textual description of the data confidentiality requirement. It is generated based on the text template for data confidentiality requirements (see Listing 7.5 on page 103).



SDP2FE : DataConfidentialityRequirement

Data subject = Patient
Personal data = {patientBillingDetails, treatmentCosts}
Counterstakeholders = {Financial Employee}
Availability = all
Description = All Financial Employee shall be allowed to access personal data patientBillingDetails, and treatmentCosts of the Patient.

Figure 12.3.: *Data confidentiality requirement for the data subject patient and the counterstakeholder financial employee*

Listing 12.3: *OCL specification of the additional attribute* processedPersonalData *of the class* Person

```
context Person
def: processedPersonalData : Set(Data) =
  self.relatedtos.personalData.pdavailableats→select(pd|
    pd.domain.domain.designed or
    pd.origin.statement→exists(s|s.oclIsTypeOf(Requirement))).personalData
```

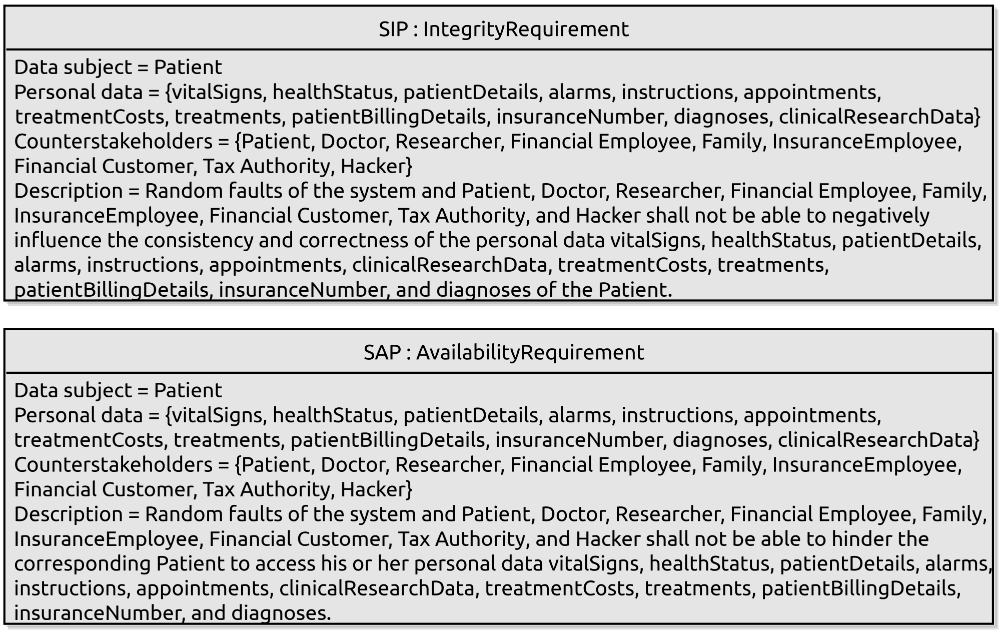
Listing 12.4: *OCL specification of the operation* generateAvailabilityIntegrityRequirements

```
context Person::generateAvailabilityIntegrityRequirements()
pre: true
post:
self.privacyrequirements→select(pr|
      pr.oclIsTypeOf(IntegrityRequirement))→one(ir|
        ir.dataSubject = self and
        ir.counterstakeholders = Person.allInstances() and
        ir.personalData = self.processedPersonalData) and
self.privacyrequirements→select(pr|
      pr.oclIsTypeOf(AvailabilityRequirement))→one(ar|
        ar.dataSubject = self and
        ar.counterstakeholders = Person.allInstances() and
        ar.personalData = self.processedPersonalData)
```

12.2.1.2. Generating Integrity and Availability Requirements

The generation of the general integrity and availability requirements of my taxonomy (see Figure 7.2 on page 101) is straightforward. I instantiate for each data subject whose personal data are stored by the machine or that are made available at a domain of the system due to a functional requirement (computed by the operation processedPersonalData specified in Listing 12.3) one integrity requirement and one availability requirement. All personal data that were identified to be available at a designed domain are associated to these two requirements. The collection of counterstakeholders is set to all persons in the ProPAn model. This models that no human in the system shall negatively influence the availability and integrity of the data subject's personal data. During the next substep of my method, the analysis team may decide to refine the set of counterstakeholders. The operation generateAvailabilityIntegrityRequirements creates availability and integrity requirements for a data subject as described above (see Listing 12.4).

Application to EHS The personal data diagram of the patient is shown in Figure 11.13 on page 197 and the available data diagram of the designed domain EHR in Figure 11.14 on page 198. The integrity and availability requirements generated for the patient (see Figure 12.4) reference all the personal data available at the designed domain EHR and additionally the treatment costs that are available at the designed domain invoice.



SIP : IntegrityRequirement

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, diagnoses, clinicalResearchData}
Counterstakeholders = {Patient, Doctor, Researcher, Financial Employee, Family, InsuranceEmployee, Financial Customer, Tax Authority, Hacker}
Description = Random faults of the system and Patient, Doctor, Researcher, Financial Employee, Family, InsuranceEmployee, Financial Customer, Tax Authority, and Hacker shall not be able to negatively influence the consistency and correctness of the personal data vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, and diagnoses of the Patient.

SAP : AvailabilityRequirement

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, diagnoses, clinicalResearchData}
Counterstakeholders = {Patient, Doctor, Researcher, Financial Employee, Family, InsuranceEmployee, Financial Customer, Tax Authority, Hacker}
Description = Random faults of the system and Patient, Doctor, Researcher, Financial Employee, Family, InsuranceEmployee, Financial Customer, Tax Authority, and Hacker shall not be able to hinder the corresponding Patient to access his or her personal data vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, and diagnoses.

Figure 12.4.: *Integrity and availability requirement for the data subject patient*

12.2.2. Generating Requirements for the Protection Goal Unlinkability

ProPAn's taxonomy of unlinkability requirements is introduced in Section 7.2.3. The metamodel for unlinkability requirements is shown in Figure 7.5 on page 106.

For the automatic generation, I only create instances of the classes UndetectabilityRequirement (see Section 12.2.2.1), DataUnlinkabilityRequirement (see Section 12.2.2.2), and AnonymityRequirement (see Section 12.2.2.3). In the step Adjust privacy requirements, an anonymity requirement may be refined to a pseudonymity requirement, and an undetectability requirement to a data confidentiality requirement (see Section 12.3).

12.2.2.1. Generating Undetectability Requirements

If personal data of a data subject are not available at a counterstakeholder and also not part of any personal data available at the counterstakeholder, then I assume that these personal data shall be undetectable for the counterstakeholder. Note that an undetectability requirement may be too strong for this case, because the counterstakeholder may be allowed to know that specific personal data exist, but may not be allowed to know the content of them. Hence, the analysis team may weaken an undetectability requirement in the next substep of the ProPAn method (see Section 12.3) to a data confidentiality requirement.

To keep the number of requirements that are generated small, I create for each triple of data subject, counterstakeholder, and availability degree (cf. Figure 7.5 on page 106) at most one undetectability requirement containing all personal data of the data subject that shall be undetectable for the counterstakeholder, while possibly being detectable for specific instances of the counterstakeholder. The generation is performed using the operation generateUndetectabilityRequirements specified in Listing 12.5.

The postcondition of the operation generateUndetectabilityRequirements consists of two parts. First, for all personal data p of the data subject ds that are available to the counterstakeholder c with availability degree ad, an undetectability requirement shall exist that states that the personal data p of the data subject ds shall be undetectable for counterstakeholder c, while specific instances of c (given by the availability degree ad) shall be able to detect p (lines 4-14). Second,

Listing 12.5: *OCL specification of the operation* generateUndetectabilityRequirements

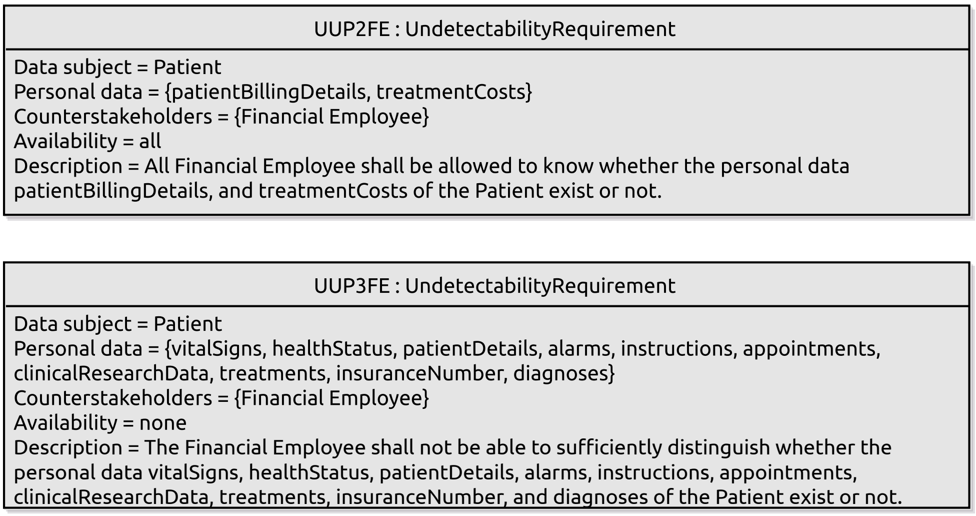
```
1  context Person::generateUndetectabilityRequirements()
2  pre: true
3  post:
4  Person.allInstances()→forAll(counterstakeholder|
5    Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
         AvailabilityDegree::all}→forAll(ad|
6      counterstakeholder.
7        availablePersonalData(self, ad)→notEmpty() implies
8      self.privacyrequirements→select(pr|
9        pr.oclIsTypeOf(UndetectabilityRequirement))→one(ur|
10         ur.dataSubject = self and
11         ur.counterstakeholders = Set{counterstakeholder} and
12         ur.personalData = counterstakeholder.
13           availablePersonalData(self, ad) and
14         ur.availability = ad and
15         ur.repudiation = Repudiation::none))) and
16 Person.allInstances()→forAll(counterstakeholder|
17     self.relatedtos.personalData−excludingAll(
18       counterstakeholder.availablePersonalData(self).personalData)
19         →notEmpty() implies
20     self.privacyrequirements→select(pr|
21       pr.oclIsTypeOf(UndetectabilityRequirement))→one(ur|
22         ur.dataSubject = self and
23         ur.counterstakeholders = Set{counterstakeholder} and
24         ur.personalData = self.relatedtos.personalData−excludingAll(
25           counterstakeholder.availablePersonalData(self).personalData) and
26         ur.availability = AvailabilityDegree::none and
27         ur.repudiation = Repudiation::none)))
```

for all personal data *p* of the data subject *ds* that are not available at the counterstakeholder *c*, an undetectability requirement shall exist that states that the personal data *p* of the data subject *ds* shall be undetectable for all possible instances of the counterstakeholder *c*, i.e., with availability degree none (lines 15-23). Note that the attribute repudiation is set to none, i.e., no repudiation requirement is initially specified. The analysis team may decide in the substep Adjust privacy requirements that another repudiation type may be required (cf. Section 12.3).

Application to EHS As stated before, I limit the application to the EHS to the data subject patient and counterstakeholder financial employee. Figure 12.2 shows which personal data of patients are available to the biddable domain financial employee. The personal data treatmentCosts and patientBillingDetails shall be available to all financial employees, while all other personal data of patients (cf. Figure 11.13) shall not be available to financial employees. Hence, two undetectability requirements are generated for the counterstakeholder financial employee and the data subject patient. These two (including their textual descriptions in the field Description) are shown in Figure 12.5.

12.2.2.2. Generating Data Unlinkability Requirements

During the data flow analysis, the analysis team elicited not only which personal data shall be available at which domains of the system, but also which data shall be linkable to each other at the domains using LinkAvailableAt relations. These relations have a transitive nature. That is, if personal data *a* are linkable to personal data *b* and *b* linkable to personal data *c*, then this implies a link between *a* and *c*. These transitive relations do not have to be modeled by the analysis team manually, they are automatically computed when needed by the ProPAn tool. During this



UUP2FE : UndetectabilityRequirement

Data subject = Patient
Personal data = {patientBillingDetails, treatmentCosts}
Counterstakeholders = {Financial Employee}
Availability = all
Description = All Financial Employee shall be allowed to know whether the personal data patientBillingDetails, and treatmentCosts of the Patient exist or not.

UUP3FE : UndetectabilityRequirement

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatments, insuranceNumber, diagnoses}
Counterstakeholders = {Financial Employee}
Availability = none
Description = The Financial Employee shall not be able to sufficiently distinguish whether the personal data vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatments, insuranceNumber, and diagnoses of the Patient exist or not.

Figure 12.5.: *Undetectability requirements for the data subject patient and the counterstakeholder financial employee*

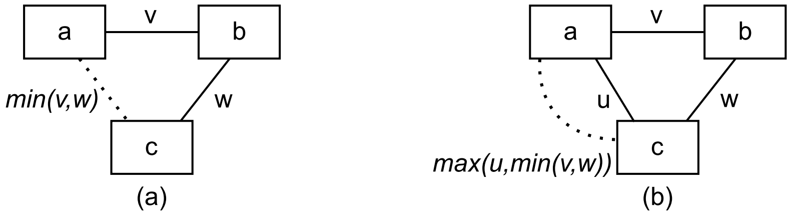


Figure 12.6.: *Cases describing how the linkability attribute of the linkable relations are determined for the transitive closure of them. (**a**) Linkable relation exists between a and b, but not between a and c; (**b**) Linkable relations exist between a, b, and c*

automatic computation, I consider all contains relations between personal data available at a domain also as linkable relations with linkability single and availability all (cf. Section 11.2).

To decide which linkability a derived link between a and c has, I distinguish two cases. The cases are visualized in Figure 12.6. First (case (a)), if not yet a link between a and c is identified during the closure computation, then the linkability between a and c is the minimum of the existing linkability v between a and b, and the existing linkability w between b and c. Second (case (b)), if there is already a link between a and c with linkability u, then I replace the existing linkability u only if the minimum of v and w has a greater linkability than u. Minimal and maximal linkability are defined by the total ordering anonymous $<$ largeGroup $<$ mediumGroup $<$ smallGroup $<$ single. The operators min and max are specified in Listing 12.6. For example, if $u =$ anonymous, $v =$ single, and $w =$ mediumGroup, then $min(v, w) =$ mediumGroup, and $max(u, min(v, w)) =$ mediumGroup. If no linkable relation is documented between personal data a and personal data b that both are available at a domain, then I consider a and b to be linkable to each other with linkability *anonymous* in the closure. That is, it is not known at the domain which personal data a are related to which personal data b.

For the closure computation, also the attribute availability of the LinkAvailableAt relations has to be considered. A transitive link is only created if both LinkAvailableAt relations have the same availability. The operation linkableClosure is used to compute the transitive closure over the LinkAvailableAt relations for a domain and a given data subject (see Listing 12.7). The closure operation returns a set of tuples that consist of the link between two Data objects (represented by a set), a linkability, and a availability. The closure contains for all personal data $p1$ and $p2$ of the data subject such that $p1$ contains $p2$, and both are available at the

Listing 12.6: *Min and max operations for the enumeration* Linkability

```
context Linkability
def: max(l : Linkability) : Linkability =
  if Set{self, l}→includes(Linkability::single) then
    Linkability::single
  else
    if Set{self, l}→includes(Linkability::smallGroup) then
      Linkability::smallGroup
    else
      if Set{self, l}→includes(Linkability::mediumGroup) then
        Linkability::mediumGroup
      else
        if Set{self, l}→includes(Linkability::largeGroup) then
          Linkability::largeGroup
        else
          Linkability::anonymous
        endif
      endif
    endif
  endif
def: min(l : Linkability) : Linkability =
  if (self.max(l) = self) then l else self endif
```

considered domain with the same degree of availability, a tuple for these personal data with linkability single and the availability of the personal data $p1$ and $p2$ (lines 4-11). The closure also contains for each LinkAvailableAt instance of the domain about personal data of the data subject a tuple representing it (lines 12-17). As all personal data available at a domain can be linked to each other with the linkability anonymous, for all pairs of personal data available at the considered domain with the same availability, the closure contains a respective tuple (lines 18-23). Finally, for all tuples $t1$ and $t2$ that have the same availability, and the links of the tuples can be connected to a new link, i.e., the intersection of the links contains one common element, a tuple shall be in the closure that connects the data of the links and has the minimal linkability (see Listing 12.6) of the tuples $t1$ and $t2$ (lines 24-30). Note that linkableClosure for a data subject may contain tuples with the same link and availability, but different linkabilities. For the generation of data unlinkability requirements, I only consider those tuples with the maximal linkability. The additional operation linkableClosureMax returns only those tuples with maximal linkability (see Listing 12.7 and Figure 12.6 case (b)).

Based on the computed closure of the linkable relation, the ProPAn tool can generate data unlinkability requirements. For each combination of data subject, counterstakeholder, availability degree, and linkability at most one data unlinkability requirement is created. The ProPAn tool decides for each pair of the data subject *ds*' personal data that are available at the counterstakeholder *c* with availability *ad* with which linkability they are linkable to each other and to which data unlinkability requirement for *ds* and *c* it has to be added. In this way, I set the attribute links (see Figure 7.5 on page 106). The attribute personalData is set to the set of all personal data that are contained in a link of the attribute links of the same requirement.

The operation generateDataUnlinkabilityRequirements (see Listing 12.8) generates for a data subject the data unlinkability requirements based on the operation linkableClosureMax (see Listing 12.7) as described above.

Application to EHS In the available data diagram of the financial employee (see Figure 12.2) it is modeled that the treatmentCosts and patientBillingDetails are linkable to each other with linkability single for all financial employees. That is, a financial employee is able to know which

Listing 12.7: *OCL specification of the operation linkableClosure*

```
context Domain::linkableClosure(dataSubject:Person) :
    Set(Tuple(link:Set(Data), l:Linkability, a:Availability))
pre: true
post:
Contains.allInstances()→forAll(c|
  Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
      AvailabilityDegree::all}→forAll(ad|
    availablePersonalData(dataSubject, ad)→includesAll(
      Set{c.contained, c.containing}) implies
    result→includes(Tuple{
      link=Set{c.contained, c.containing},
      l=Linkability::single,
      a=ad})))) and
self.linkavailableats→forAll(la|
  dataSubject.relatedtos.personalData→includesAll(la.link.data) implies
  result→includes(Tuple{
      link=la.link.data,
      l=la.linkability,
      a=la.availability})) and
Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
    AvailabilityDegree::all}→forAll(ad|
  self.availablePersonalData(dataSubject, ad)→forAll(d1, d2|
    result→includes(Tuple{
      link=Set{d1, d2},
      l=Linkability::anonymous,
      a=ad})))) and
result→forAll(t1,t2|
  (t1.availability = t2.availability and
   t1.link.data→intersection(t2.link.data)→size() = 1) implies
  result→includes(Tuple{
      link=t1.link.data→symmetricDifference(t2.link.data),
      l=t1.l.min(t2.l),
      a=t1.a}))

context Domain
def: linkableClosureMax(dataSubjectPerson) : Set(Tuple(link:Set(Data),
    l:Linkability, a:Availability)) =
  linkableClosure(dataSubject)→select(t|
    linkableClosure→forAll(t2|
      (t.link = t2.link and t.a = t2.a) implies
        t.l.max(t2.l) = t.l))
```

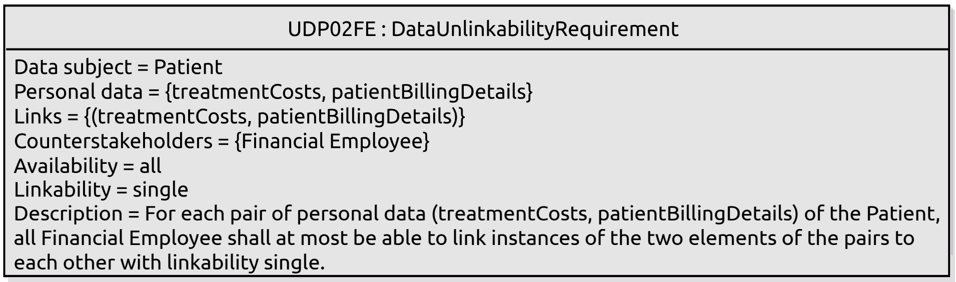
treatment costs belong to which patient billing details. Thus, I only obtain one data unlinkability requirement, namely for the linkability single and availability all, because there is no pair of personal data available at the financial employee with a different availability or linkability. This data unlinkability requirement is shown in Figure 12.7.

12.2.2.3. Generating Anonymity Requirements

To generate the anonymity requirements for a data subject ds and a counterstakeholder c, it has to be derived which personal data a can be related to the ds by c with which linkability. For this, I use the RelatedTo relation of ds' personal data diagram and the previously discussed closure of the linkable relation available at c (computed by the operation linkableClosureMax specified in Listing 12.7). The RelatedTo relation (cf. Figure 10.2 on page 157) describes for every personal data a of a data subject ds with which linkability these personal data a can be

Listing 12.8: *OCL specification of the operation generateDataUnlinkabilityRequirements*

```
context Person::generateDataUnlinkabilityRequirements()
pre: true
post:
Person.allInstances()→forAll(counterstakeholder|
  Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
      AvailabilityDegree::all}→forAll(ad|
  Set{Linkability::single, Linkability::smallGroup, Linkability::mediumGroup,
      Linkability::largeGroup, Linkability::anonymous}→forAll(l|
    linkableClosureMax(self)→select(t|t.a = ad and t.l = l).notEmpty() implies
    self.privacyrequirements→select(pr|
      pr.oclIsTypeOf(DataUnlinkabilityRequirement))→one(dur|
        dur.dataSubject = self and
        dur.counterstakeholders = Set{counterstakeholder} and
        dur.personalData = dur.links.data and
        linkableClosureMax(self)→select(t|t.a = ad and t.l = l)→forAll(t|
          dur.links→exists(link|link.data = t.link)) and
        dur.availability = ad and
        dur.linkability = l and
        dur.repudiation = Repudiation::none))))
```



UDP02FE : DataUnlinkabilityRequirement

Data subject = Patient
Personal data = {treatmentCosts, patientBillingDetails}
Links = {(treatmentCosts, patientBillingDetails)}
Counterstakeholders = {Financial Employee}
Availability = all
Linkability = single
Description = For each pair of personal data (treatmentCosts, patientBillingDetails) of the Patient, all Financial Employee shall at most be able to link instances of the two elements of the pairs to each other with linkability single.

Figure 12.7.: *Data unlinkability requirement for the data subject patient and the counterstakeholder financial employee*

related to *ds* (by any domain). However, it is possible that a higher linkability of *a* to *ds* exists at a domain. This case is illustrated in Figure 12.8. If personal data *a* are linkable to personal data *b* with linkability *v* at a domain, and *b* can be related with linkability *w* to *ds* at that domain, then this implies that *a* can be linked to *ds* with at least the linkability $min(v, w)$ at the domain. Similar as for the closure of the linkable relation, I only take into account the derived linkability $min(v, w)$ if it has a greater linkability than the existing linkability u. The described combination of the linkable and related-to relations introduces a new relation relatedClosure that provides for a domain (including counterstakeholders), data subject, and personal data of the data subject that are available to the counterstakeholder the linkability with which the domain can relate the personal data to the data subject. The relatedClosure relation is generated on demand by the ProPAn tool, and is used to instantiate anonymity requirements. The additional operation relatedClosureMax returns only those tuples of relatedClosure with maximal linkability, similar to the operation linkableClosureMax for the operation linkableClosure.

The two operations relatedClosure and relatedClosureMax are specified in Listing 12.9. The first part of the postcondition of relatedClosure adds all personal data available at the domain to the relation with the linkability that is documented for the personal data by the RelatedTo relation for the respective data subject (lines 5-10). In the second part, the linkable closure is used to add the derived linkability of personal data to the relation as described above and illustrated in Figure 12.8 (lines 11-16).

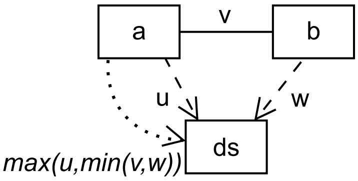


Figure 12.8.: *Case describing how the linkability attribute of the related to relations are determined for the transitive closure of the linkable relations. a and b are related to ds and a linkable relation between a and b exists.*

Listing 12.9: *OCL specification of the operation* relatedClosure

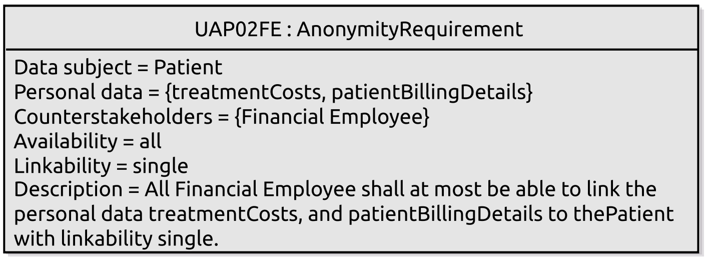
```
context Domain::relatedClosure(dataSubject:Person) : Set(Tuple(d:Data,
    l:Linkability, a:Availability))
pre: true
post:
Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
    AvailabilityDegree::all}→forAll(ad|
  availablePersonalData(dataSubject, ad).relatedtos→select(rt|
      rt.dataSubject = dataSubject)→forAll(rt|
        result→includes(Tuple{
          d=rt.personalData,
          l=rt.linkability,
          a=ad})) and
  linkableClosureMax(dataSubject)→select(lt|lt.a = ad)→forAll(lt|
    result→select(rt|t.link→includes(rt.d))→forAll(rt|
      result→includes(Tuple{
        d=lt.link→excluding(rt.d)→any(true) and
        l=rt.l.min(lt.l) and
        a=ad}))))

context Domain
def: relatedClosureMax(dataSubject:Person) : Set(Tuple(d:Data, l:Linkability,
    a:Availability)) =
  relatedClosure(dataSubject)→select(t|t.a = ad and t.l = l and
    relatedClosure→forAll(t2|
      (t.d = t2.d and t.a = t2.a) implies t.l.max(t2.l) = t.l))
```

Listing 12.10 shows the specification of the operation generateAnonymityRequirements. This operation creates for each combination of data subject, counterstakeholder, availability, and linkability an anonymity requirement if personal data exist that are linkable to the data subject with the given linkability by counterstakeholders with the given availability (lines 4-7). If this is the case, one anonymity requirement is created for the combination of data subject, counterstakeholder, linkability, and availability that references all personal data that are available at the counterstakeholder with the given availability and that are linkable to the data subject with the given linkability based on the operation relatedClosureMax (lines 8-16).

Application to EHS From the generation of the data unlinkability requirements, it is known that the personal data treatment costs and patient billing details of the patient that are available at the financial employee are linkable to each other with linkability single. The patient's personal data diagram (see Figure 11.13 on page 197) shows that the personal data patientBillingDetails can be related to the single patient they belong to and the treatmentCosts only to a largeGroup of patients. The operation relatedClosureMax reveals that also the treatmentCosts can be linked to the single patient they belong to, because the treatmentCosts are linkable to the patientBillingDetails with linkability single and the patientBillingDetails allow to identify the patient they belong

Listing 12.10: *OCL specification of the operation generateAnonymityRequirements*

```
context Person::generateAnonymityRequirements()
pre: true
post:
Person.allInstances()→forAll(counterstakeholder|
  Set{AvailabilityDegree::individual, AvailabilityDegree::authorized,
      AvailabilityDegree::all}→forAll(ad|
  Set{Linkability::single, Linkability::smallGroup, Linkability::mediumGroup,
      Linkability::largeGroup, Linkability::anonymous}→forAll(l|
    relatedClosureMax→select(t|t.a = ad and t.l = l)→notEmpty() implies
    self.privacyrequirements→select(pr|
      pr.oclIsTypeOf(AnonymityRequirement))→one(ar|
        ar.dataSubject = self and
        ar.counterstakeholders = Set{counterstakeholder} and
        ar.personalData =
          relatedClosureMax→select(t|t.a = ad and t.l = l).d
        ar.availability = ad and
        ar.linkability = l and
        ar.repudiation = Repudiation::none))))
```



UAP02FE : AnonymityRequirement

Data subject = Patient
Personal data = {treatmentCosts, patientBillingDetails}
Counterstakeholders = {Financial Employee}
Availability = all
Linkability = single
Description = All Financial Employee shall at most be able to link the personal data treatmentCosts, and patientBillingDetails to thePatient with linkability single.

Figure 12.9.: *Anonymity requirement for the data subject patient and the counterstakeholder financial employee*

to. Consequently, the ProPAn tool only generates one anonymity requirement for the patient and the financial employee. This anonymity requirement is shown in Figure 12.9.

12.2.3. Generating Requirements Related to the Protection Goal Transparency

My considered taxonomy of transparency requirements is presented in Figure 7.6 on page 112. The basis to derive the transparency requirements for the system-to-be are the data flow graphs generated in the step Data Flow Analysis (see Chapter 11). These graphs visualize the intended and expected flow of personal data through the system. I already used the initial data flow graphs generated in the step Privacy Threshold Analysis (see Chapter 10) to identify which personal data are potentially collected and stored by the machine, and flow to other domains due to the machine. The analysis team uses this information in the substep Decide on privacy threshold (see Section 10.5) to determine whether a detailed privacy analysis is necessary or not.

I reuse some of the operations introduced in Section 10.5 to generate collection, flow, and storage information requirements (see Sections 12.2.3.1, 12.2.3.2, and 12.2.3.3, respectively). The generation of exceptional information requirements is explained in Section 12.2.3.4 and the generation of presentation requirements for the transparency requirements in Section 12.2.3.5.

12.2.3.1. Generating Collection Information Requirements

Data collection happens at those places in the system-to-be where personal data of a data subject flow from a given domain to a designed domain, which represents a part of the machine

Listing 12.11: *OCL specification of the operation* generateCollectionInformationRequirements

```
context Person::generateCollectionInformationRequirements()
pre: true
post:
self.finalDfg.collectionEdges.data→forAll(d|
  self.privacyrequirements→select(pr|
    pr.oclIsTypeOf(CollectionInformationRequirement))→one(tc|
      tc.dataSubject = self and
      tc.counterstakeholders = Set{} and
      tc.personalData = Set{d} and
      tc.origin =
        d.pdavailableats→select(pd|pd.domain.domain.designed).origin and
      tc.purpose =
        d.pdavailableats→select(pd|pd.domain.domain.designed).purpose and
      tc.method = self.relatedtos→any(rt|rt.personalData = d).collection))
```

(cf. Section 2.2). This is, because every personal data that are collected by the system-to-be flow from the machine's environment (a given domain) to a part of it (a designed domain).

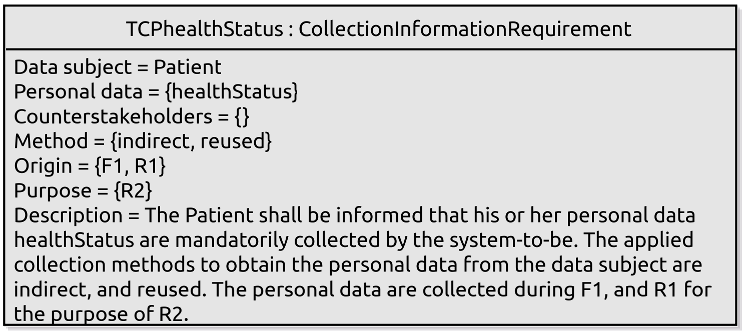
For each personal data p and data subject ds for whom p represents personal data, I propose to instantiate a collection information requirement with the attributes dataSubject $= ds$ and personalData $= \{p\}$ if p flows from a given domain to a designed domain. Additionally, the ProPAn tool automatically sets the attributes origin, purpose, and method. As origin, I set the statements due to which p flow to the designed domains, and as purpose, I set the statements due to which p flow from the designed domain further to other domains (both documented by the respective PersonalDataAvailableAt relation as origin and purpose of p). The attribute method is instantiated using the documented collection methods from the RelatedTo relations presented in the personal data diagram of ds. All other attributes have to be adjusted by the analysis team in next step of my method.

The operation generateCollectionInformationRequirements creates the collection information requirements for a data subject as described above. The operation is specified in Listing 12.11 and uses in its specification the additional attribute collectionEdges for data flow graphs (see Listing 10.11 on page 168)

Application to EHS The data that are collected from patients can be derived from the in-going edges of the designed domains EHR and Invoice in the patient's final data flow graph shown in Figure 11.16 on page 201. The edges 1, 7, and 8 describe which personal data are collected by the machine. Figure 12.10 shows the generated collection information requirement for the data subject patient and the personal data healthStatus. In Figure 12.10, only the instantiated attributes are shown, all other attributes have to be set by the analysis team in the step Adjust privacy requirements. The collection methods for the health status were already collected during the step Privacy Threshold Analysis and documented by a RelatedTo relation in the patient's personal data diagram (see Figure 11.13 on page 197).

12.2.3.2. Generating Flow Information Requirements

Data subjects shall be informed about all data flows that are introduced by the machine, while they do not have to be informed about the data flows inside the machine, because they only have to be informed about the behavior of the machine as a whole. Additionally, data subjects do not have to be informed about data flows that happen independently of the machine, because these flows are out of the scope of the machine. Hence, I only consider the documented data flows to given domains that are caused by the machine. I distinguish two kinds of data flows


TCPhealthStatus : CollectionInformationRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {}
Method = {indirect, reused}
Origin = {F1, R1}
Purpose = {R2}
Description = The Patient shall be informed that his or her personal data healthStatus are mandatorily collected by the system-to-be. The applied collection methods to obtain the personal data from the data subject are indirect, and reused. The personal data are collected during F1, and R1 for the purpose of R2.

Figure 12.10.: *Collection information requirement for the data subject patient and the personal data health status*

Listing 12.12: *OCL specification of the operation* generateFlowInformationRequirements

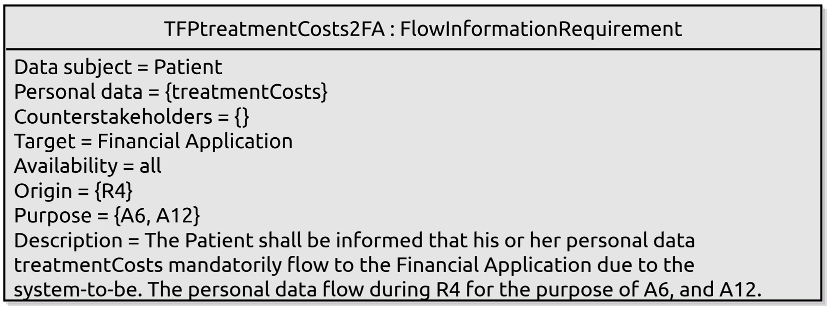
```
context Person::generateFlowInformationRequirements()
pre: true
post:
self.finalDfg.flowEdges→forAll(fe | fe.data→forAll(d |
  d.pdavailableats→select(pd | pd.domain = fe.target).availability→forAll(ad |
    self.privacyrequirements→select(pr |
      pr.oclIsTypeOf(FlowInformationRequirement))→one(tf |
        tf.dataSubject = self and
        tf.counterstakeholders = Set{} and
        tf.personalData = Set{d} and
        tf.origin = d.pdavailableats→select(pd |
            pd.domain = fe.target and pd.availability = ad).origin and
        tf.purpose = d.pdavailableats→select(pd |
            pd.domain = fe.target and pd.availability = ad).purpose and
        tf.target = fe.target and
        tf.availability = ad))))
```

introduced by the machine.

First, the machine may introduce flows of personal data from designed domains to given domains. Second, there might be data flows between given domains that originate from a functional requirement. This can be the case if the machine is only responsible to forward personal data p of a data subject ds from a given domain d_1 to another given domain d_2 without collecting or storing the data. I instantiate a flow information requirement for each combination of given domain d_2 and data subject ds if personal data p of ds flow to d_2 from a designed domain, or due to a functional requirement from a given domain.

In both cases of data flows, I propose to generate a flow information requirement with dataSubject $= ds$, personalData $= \{p\}$, and target $= d_2$. Additionally, the ProPAn tool automatically sets the attributes origin and purpose to the set of statements for which it was documented that the personal data p have to be available at d_2, and due to which statements p flow to d_2, respectively. The attribute availability is set to the availability with which the personal data p are available at the target d_2. If p are available with different availabilities at the target, then for each availability a respective flow information requirement is instantiated. The other attributes have to be set manually by the analysis team in the substep Adjust privacy requirements.

The operation generateFlowInformationRequirements creates the flow information requirements for a data subject as described above. The operation is specified in Listing 12.12 and uses in its specification the additional attribute flowEdges for data flow graphs (see Listing 10.11 on page 168).



TFPtreatmentCosts2FA : FlowInformationRequirement

Data subject = Patient
Personal data = {treatmentCosts}
Counterstakeholders = {}
Target = Financial Application
Availability = all
Origin = {R4}
Purpose = {A6, A12}
Description = The Patient shall be informed that his or her personal data treatmentCosts mandatorily flow to the Financial Application due to the system-to-be. The personal data flow during R4 for the purpose of A6, and A12.

Figure 12.11.: *Flow information requirement for the data subject patient, the target financial application, and the personal data treatment costs*

Listing 12.13: *Max operation for the enumeration* Duration

```
context Duration
def: max(d : Duration) : Duration =
  if Set{self, d}->includes(Duration::unlimited) then
    Duration::unlimited
  else
    if Set{self, d}->includes(Duration::untilDeleted) then
      Duration::untilDeleted
    else
      Duration::forAction
    endif
  endif
```

Application to EHS In the EHS scenario, there is no example for a flow of personal data between two given domains that originates from a functional requirement, but the final data flow graph for the patient (see Figure 11.16) shows several data flows from the designed domains EHR and Invoice to given domains (edges 3, 5, 6, 11, and 16). For the data subject patient, the target financial application, and the personal data treatment costs, the ProPAn tool generates the flow information requirement shown in Figure 12.11 based on edge 11.

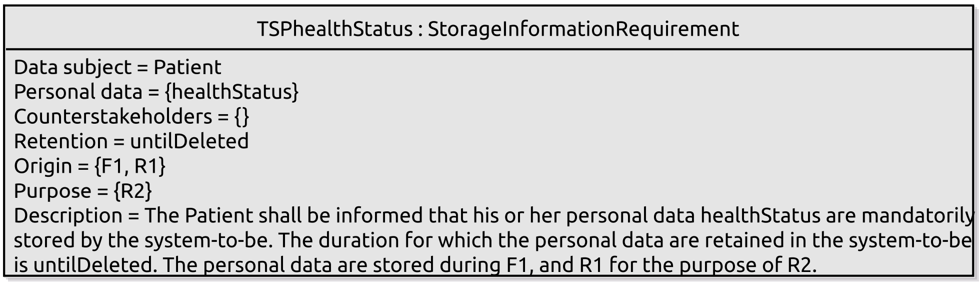
12.2.3.3. Generating Storage Information Requirements

All personal data that are available at a designed domain, are stored by the machine for at least the time that it is necessary to be there to satisfy the functional requirements, which is documented by the attribute duration of the corresponding PersonalDataAvailableAt object (see Figure 11.3 on page 176). I propose to instantiate for each pair of data subject ds and personal data p of ds a storage information requirement if p are available at a designed domain. For a storage information requirement with dataSubject $= ds$ and personalData $= \{p\}$, I set retention to the maximal duration with which they are available at a designed domain. The maximal duration is determined by the total ordering forAction $<$ untilDeleted $<$ unlimited (see also Listing 12.13). Additionally, the ProPAn tool can automatically derive the origin and purpose for which p are stored by the machine from the available data diagrams of the designed domains.

The operation generateStorageInformationRequirements creates the storage information requirements for a data subject as described above. The operation is specified in Listing 12.14 and uses the PersonalDataAvailableAt relations of the designed domains documented in the previous step Data Flow Analysis. To determine the maximal duration for which the personal data are available at a designed domain, the operation max of the class Duration is used (see Listing 12.13).

Listing 12.14: *OCL specification of the operation* generateStorageInformationRequirements

```
context Person::generateStorageInformationRequirements()
pre: true
post:
self.relatedtos.personalData.pdavailableats→select(pd|
  pd.domain.domain.designed).personalData→forAll(d|
    let pdavailableatsDesigned : Set(PersonalDataAvailableAt) =
      d.pdavailableats→select(pd|pd.domain.domain.designed)
    in
    self.privacyrequirements→select(pr|
      pr.oclIsTypeOf(StorageInformationRequirement))→one(ts|
        ts.dataSubject = self and
        ts.counterstakeholders = Set{} and
        ts.personalData = Set{d} and
        ts.origin =
          pdavailableatsDesigned.origin and
        ts.purpose =
          pdavailableatsDesigned.purpose and
        ts.retention =
          pdavailableatsDesigned.duration→any(d|
            pdavailableatsDesigned.duration→forAll(d2|
              d.max(d2) = d))))
```


TSPhealthStatus : StorageInformationRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {}
Retention = untilDeleted
Origin = {F1, R1}
Purpose = {R2}
Description = The Patient shall be informed that his or her personal data healthStatus are mandatorily stored by the system-to-be. The duration for which the personal data are retained in the system-to-be is untilDeleted. The personal data are stored during F1, and R1 for the purpose of R2.

Figure 12.12.: *Storage information requirement for the data subject patient, and the personal data health status*

Application to EHS The personal data available to the designed domains EHR and Invoice are stored by the machine of the EHS. The available data diagram for the EHR is shown in Figure 11.14 on page 198 and a preliminary available data diagram for the Invoice is shown in Figure 11.12 on page 194 as part of the statement consideration diagram for R3 during the step Data Flow Analysis. For all these personal data, the ProPAn tool generates storage information requirements. As an example, the storage information requirement for the personal data healthStatus is shown in Figure 12.12.

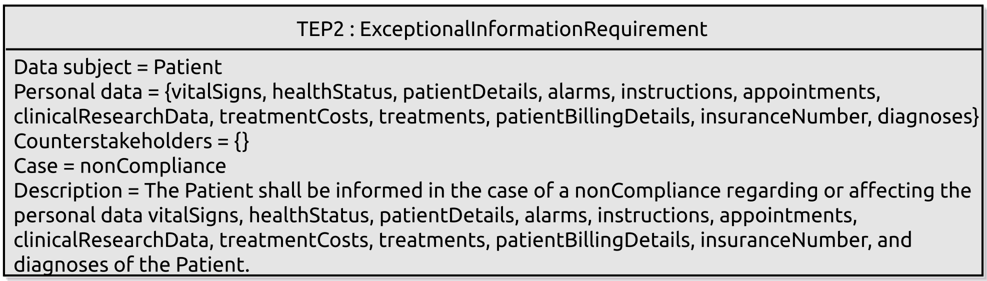
12.2.3.4. Generating Exceptional Information Requirements

The need for exceptional information requirements strongly depends on the legal requirements on the system-to-be. I consider for the generation of exceptional information requirements the GDPR (European Commission, 2016) as source of legal requirements.

Article 33 of the GDPR is about the notification of a personal data breach to the supervisory authority and Article 34 prescribes the communication of a personal data breach to the data subject. Hence, I generate an exceptional information requirement for each data subject with all personal data of him or her that are processed by the machine with the attribute case set to dataBreach (cf. Figure 7.6 on page 112).

Listing 12.15: *OCL specification of the operation* generateExceptionalInformationRequirements

```
context Person::generateExceptionalInformationRequirements()
pre: true
post:
Set{ExceptionalCase::dataBreach, ExceptionalCase::systemChange,
    ExceptionalCase::nonCompliance,
    ExceptionalCase::authorityRequest}→forAll(ec|
  self.privacyrequirements→select(pr|
    pr.oclIsTypeOf(ExceptionalInformationRequirement))→one(te|
      te.dataSubject = self and
      te.counterstakeholders = Set{} and
      te.personalData = self.processedPersonalData and
      te.case = ec))
```



TEP2 : ExceptionalInformationRequirement

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, diagnoses}
Counterstakeholders = {}
Case = nonCompliance
Description = The Patient shall be informed in the case of a nonCompliance regarding or affecting the personal data vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, and diagnoses of the Patient.

Figure 12.13.: *Exceptional information requirements for the data subject patient and the case* nonCompliance

Article 58 of the GDPR describes the powers of supervisory authorities. These powers imply the need for two additional exceptional information requirements for each data subject and all of his or her personal data that are processed by the machine. The first exceptional information requirement has the type authorityRequest and expresses that the controller has to provide all information to the authorities that these need in order to perform their duties. The second exceptional information requirement has the type nonCompliance and represents that authorities have to be informed if personal data are processed in breach with the EU General Data Protection Regulation.

Furthermore, data subjects and authorities shall be informed about changes in the system that imply a change in the processing of personal data. Hence, I propose to instantiate an exceptional information requirement with the attribute case set to systemChange for each data subject and all of his or her personal data.

The operation generateExceptionalInformationRequirements specified in Listing 12.15 creates for a data subject the four needed exceptional information requirements. The personal data processed by the machine are derived by the additional attribute processedPersonalData (see Listing 12.3).

Application to EHS Figure 12.13 shows the exceptional information requirement TEP2 for the data subject patient and exceptional case nonCompliance generated by the ProPAn tool. The ProPAn tool also generated for the other exceptional cases respective requirements, e.g., the exceptional information requirement TEP0 with the exceptional case dataBreach.

Listing 12.16: *OCL specification of the operation* generatePresentationRequirements

```
context Person::generatePresentationRequirements()
pre: true
post:
self.privacyrequirements→select(pr|
  pr.oclIsKindOf(TransparencyRequirement))→forAll(t|
    not t.presentationrequirement.oclIsUndefined())
```

12.2.3.5. Generating Presentation Requirements

The metamodel for transparency requirements (see Figure 7.6 on page 112) requires that each transparency requirement has a presentation requirement that specifies how its contained information shall be presented to the data subjects (and authorities). Hence, the ProPAn tool generates for each transparency requirement in the ProPAn model a presentation requirement using the operation generatePresentationRequirements (see Listing 12.16). The attributes of these requirements have to be set by the analysis team in the step Adjust privacy requirements.

12.2.4. Generating Requirements Related to the Protection Goal Intervenability

In this section, I describe how I automatically generate data subject and authority intervention requirements (see Figure 7.7 on page 116) based on the processing and exceptional information requirements in the model, and the intervention needs described by the EU General Data Protection Regulation.

12.2.4.1. Generating Data Subject Intervention Requirements

In my taxonomy of intervenability requirements (see Figure 7.7), each data subject intervention requirement is related to a processing information requirement. Hence, the processing information requirements that are generated as described in Section 12.2.3 are the basis for the generation of data subject intervention requirements. To determine which kinds of intervention requirements are needed, I consider the GDPR.

The GDPR promotes an opt-in scheme for the processing of personal data (cf. Articles 6, 7, and 9 of the GDPR). Hence, I add to all processing information requirements consent to the documented processing grounds and generate two intervenability requirements for each processing information requirement. First, I generate one with the intervention type doNotConsent and effect noProcessing. The time for this intervention requirement depends on the kind of processing information requirement. For collection information requirements, time is set to beforeCollection. For flow information requirements, time is set to beforeCollection if the personal data flowing are collected from the data subject, i.e., a collection information requirement for the corresponding personal data exists, and to beforeTransmission if the flowing personal data are not collected by the machine, e.g., the personal data are derived from other personal data. For storage information requirements, I distinguish the same two cases. If the stored personal data were collected by the machine, then I set time to beforeCollection, and else I set time to atRecording. Second, I generate a data subject intervention requirement with type withdrawConsent, effect erasure, and time anyTime for each processing information requirement.

The right of access by the data subject (Article 15) implies that the data subject shall be able to request access to his or her personal data stored by the software or sent to a processor or third party. Hence, I generate for each storage and flow information requirement a data subject intervention requirement with type review, effect access, and time anyTime.

The right to rectification (Article 16) allows data subjects to challenge the accuracy and

Processing Information Requirement	Requirements with Intervention Type
Collection, Storage, Flow	doNotConsent, withdrawConsent, object
Storage, Flow	review, challengeAccuracy, challengeCompleteness
Collection	requestDataCopy

Table 12.1.: *Overview of the generated data subject intervention requirements for processing information requirements.*

completeness of their personal data stored by the software or sent to processors or third parties at any time. Hence, I instantiate two additional data subject intervention requirements for each storage and flow information requirement. First, I generate one with type challengeAccuracy, effect correction, and time anyTime, and second, one with type challengeCompleteness, effect amendment, and time anyTime.

The rights to restriction of processing (Article 18) and to object (Article 21) describe that, at any time, data subjects can object to the processing of their personal data and the processing of these personal data shall then be restricted accordingly. Additionally, Article 17 states that in the case of objection to the processing of personal data, the respective personal data shall be erased if there are no overriding legitimate grounds for the processing. Hence, I generate an additional data subject intervention requirement for each processing information requirement with type object, time anyTime, and the effects restrictedProcessing and erasure.

The right to data portability (Article 20) allows data subjects to request *"the personal data concerning him or her, which he or she has provided to a controller, in a structured, commonly used and machine-readable format [..]"* (European Commission, 2016). Hence, I generate a data subject intervention requirement with type requestDataCopy, effect dataCopy and time anyTime for each collection information requirement. Note that the right to data portability only references the personal data provided by the data subject and not data that results from further processing of the provided data. Because of this, I only generate the intervention requirements for collection information requirements.

In Table 12.1, I summarize for which processing information requirements which types of data subject intervention requirements are generated.

The operation generateDataSubjectInterventionRequirements generates the data subject intervention requirements as described above and links them as controlOptions to the respective processing information requirements (see Listing 12.17).

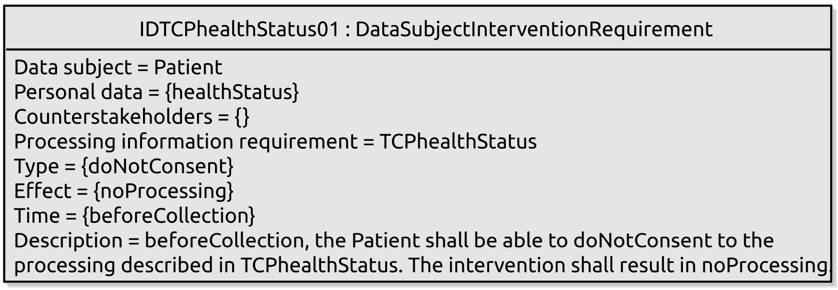
Application to EHS Example Among others, the ProPAn tool generated a collection information requirement for the patient's personal data healthStatus (see Figure 12.10). Figure 12.14 shows one of the four data subject intervention requirements that are generated for it. As described above, the ProPAn tool adds to the collection information requirement consent to the processing grounds, and the four data subject intervention requirements with the types requestDataCopy, doNotConsent, withdrawConsent, and object and the corresponding effects and times to the controlOptions. Analogously, data subject intervention requirements are generated for all other processing information requirements. For storage and flow information requirements, instead of an intervention requirement with type requestDataCopy, a requirement with type review is created. Additionally, intervention requirements with types challengeAccuracy and challengeCompleteness are generated for storage and flow information requirements (cf. Table 12.1).

12.2.4.2. Generating Authority Intervention Requirements

Authority intervention requirements are linked to exceptional information requirements (cf. Figure 7.7 on page 116), and hence, I propose to create authority intervention requirements based

Listing 12.17: *OCL specification of the operation* generateDataSubjectInterventionRequirements

```
context Person::generateDataSubjectInterventionRequirements()
pre: true
post:
self.privacyrequirements→select(pr|
  pr.oclIsKindOf(ProcessingInformationRequirement))→forAll(tp|
    tp.grounds→includes(ProcessingGrounds::consent) and
    tp.controlOptions→forAll(id|
      id.dataSubject = self and
      id.counterstakeholders = Set{} and
      id.personalData = tp.personalData) and
    tp.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::doNotConsent} and
      id.effect = Set{InterventionEffect::noProcessing} and
      id.time =
        if tp.oclIsTypeOf(CollectionInformationRequirement) then
          Set{ActionTime::beforeCollection}
        else
          if self.privacyrequirements→select(pr|
            pr.oclIsKindOf(CollectionInformationRequirement))→exists(tc|
              tc.personalData→includesAll(tp.personalData)) then
            Set{ActionTime::beforeCollection}
          else
            if tp.oclIsTypeOf(FlowInformationRequirement) then
              Set{ActionTime::beforeTransmission}
            else
              Set{ActionTime::atRecording}
            endif
          endif
        endif) and
    tp.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::withdrawConsent} and
      id.effect = Set{InterventionEffect::erasure} and
      id.time = Set{ActionTime::anyTime}) and
    tp.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::object} and
      id.effect = Set{InterventionEffect::restrictedProcessing,
          InterventionEffect::erasure} and
      id.time = Set{ActionTime::anyTime})) and
self.privacyrequirements→select(pr|
  pr.oclIsTypeOf(StorageInformationRequirement) or
  pr.oclIsTypeOf(FlowInformationRequirement))→forAll(tp|
    tp.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::review} and
      id.effect = Set{InterventionEffect::access} and
      id.time = Set{ActionTime::anyTime}) and
    tp.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::challengeAccuracy} and
      id.effect = Set{InterventionEffect::correction} and
      id.time = Set{ActionTime::anyTime}) and
    tp.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::challengeCompleteness} and
      id.effect = Set{InterventionEffect::amendment} and
      id.time = Set{ActionTime::anyTime})) and
self.privacyrequirements→select(pr|
  pr.oclIsTypeOf(CollectionInformationRequirement))→forAll(tc|
    tc.controlOptions→one(id|
      id.type = Set{DataSubjectIntervention::requestDataCopy} and
      id.effect = Set{InterventionEffect::dataCopy} and
      id.time = Set{ActionTime::anyTime}))
```



IDTCPhealthStatus01 : DataSubjectInterventionRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {}
Processing information requirement = TCPhealthStatus
Type = {doNotConsent}
Effect = {noProcessing}
Time = {beforeCollection}
Description = beforeCollection, the Patient shall be able to doNotConsent to the processing described in TCPhealthStatus. The intervention shall result in noProcessing.

Figure 12.14.: *Data subject intervention requirement of type* doNotConsent *for the collection information requirement for the data subject patient and the personal data health status*

on the previously generated exceptional information requirements.

Articles 33 and 34 prescribe the communication of personal data breaches to supervisory authorities and data subjects, but the GDPR does not mention intervention options for authorities that are directly related to the occurrence of data breaches. Hence, I do not create related authority intervention requirements for the exceptional information requirements with case dataBreach. I also do not propose to generate authority intervention requirements for exceptional information requirements with case systemChange, as the GDPR does not explicitly state intervention options for authorities to system changes.

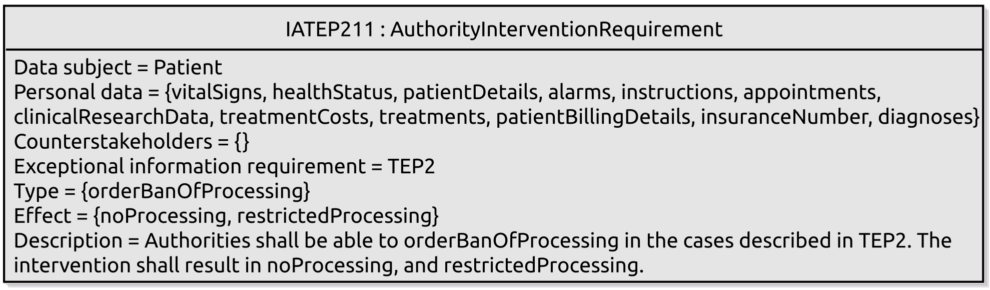
Based on Article 58, I proposed to generate exceptional information requirements with the types authorityRequest and nonCompliance in Section 12.2.3. Associated to the exceptional information requirements with case authorityRequest, I generate an authority intervention requirement with type obtainAccess, which represents that authorities can request access to the processed personal data (effect access). For the exceptional information requirements with the type nonCompliance, Article 58 implies to generate an authority intervention requirement of each authority intervention type (except obtainAccess) and possible intervention effect (see Table 6.3 on page 92). Hence, the ProPAn tool generates authority intervention requirements for all of the valid combinations of types (except obtainAccess) and effects shown in Table 6.3.

The operation generateAuthorityInterventionRequirements generates for a data subject the authority intervention requirements as described above (Listing 12.18).

Application to EHS Example For the EHS example, the ProPAn tool generated for each exceptional case an exceptional information requirement for the data subject patient and all of his or her personal data processed by the machine. The exceptional information requirement with case authorityRequest has an authority intervention requirement with type obtainAccess and the exceptional information requirement with case nonCompliance has four authority intervention requirements associated as described above. The authority intervention requirement of type orderBanOfProcessing for the exceptional information requirement shown in Figure 12.13 is shown in Figure 12.15.

Listing 12.18: *OCL specification of the operation* generateAuthorityInterventionRequirements

```
context Person::generateAuthorityInterventionRequirements()
pre: true
post:
self.privacyrequirements→select(pr|
  pr.oclIsKindOf(ExceptionalInformationRequirement))→forAll(te|
    te.authorityinterventionrequirements→forAll(ia|
      ia.dataSubject = self and
      ia.counterstakeholders = Set{} and
      ia.personalData = te.personalData)) and
self.privacyrequirements→select(pr|
  pr.oclIsKindOf(ExceptionalInformationRequirement) and
  pr.case = ExceptionalCase::authorityRequest)→forAll(te|
    te.authorityinterventionrequirements→one(ia|
      ia.type = Set{AuthorityIntervention::obtainAccess} and
      ia.effect = Set{InterventionEffect::access})) and
self.privacyrequirements→select(pr|
  pr.oclIsKindOf(ExceptionalInformationRequirement) and
  pr.case = ExceptionalCase::nonCompliance)→forAll(te|
    te.authorityinterventionrequirements→one(ia|
      ia.type = Set{AuthorityIntervention::suspendDataFlows} and
      ia.effect = Set{InterventionEffect::suspendedDataFlows}) and
    te.authorityinterventionrequirements→one(ia|
      ia.type = Set{AuthorityIntervention::orderBanOfProcessing} and
      ia.effect = Set{InterventionEffect::noProcessing,
          InterventionEffect::restrictedProcessing}) and
    te.authorityinterventionrequirements→one(ia|
      ia.type = Set{AuthorityIntervention::orderErasure} and
      ia.effect = Set{InterventionEffect::noProcessing,
          InterventionEffect::erasure}) and
    te.authorityinterventionrequirements→one(ia|
      ia.type = Set{AuthorityIntervention::orderRectification} and
      ia.effect = Set{InterventionEffect::correction,
          InterventionEffect::amendment}))
```



IATEP211 : AuthorityInterventionRequirement

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, diagnoses}
Counterstakeholders = {}
Exceptional information requirement = TEP2
Type = {orderBanOfProcessing}
Effect = {noProcessing, restrictedProcessing}
Description = Authorities shall be able to orderBanOfProcessing in the cases described in TEP2. The intervention shall result in noProcessing, and restrictedProcessing.

Figure 12.15.: *Generated authority intervention requirement with type* orderBanOfProcessing *for exceptional information requirement* TEP2

12.2.4.3. Generating Intervention Information Requirement

Article 12 of the GDPR describes that controllers have to inform data subjects about the status of the interventions they requested. Hence, I generate one intervention information requirement for each data subject intervention requirement, as already prescribed by the multiplicities in my proposed metamodel of intervenability requirements (see Figure 7.7 on page 116). This generation is realized by the operation generateInterventionInformationRequierments specified in Listing 12.19.

Listing 12.19: *OCL specification of the operation* generateInterventionInformationRequierments

```
context Person::generateInterventionInformationRequierments()
pre: true
post:
self.privacyrequirements→select(pr|
  pr.oclIsKindOf(DataSubjectIntervention))→forAll(id|
    id.dataSubject = pr.dataSubject and
    id.personalData = pr.personalData and
    id.counterstakeholders = pr.counterstakeholders)
```

12.3. Adjust Privacy Requirements

After the automatic generation of the privacy requirements candidates in step Generate privacy requirements, the analysis team manually inspects the generated privacy requirements and has the possibility to complete and adjust these in the step Adjust privacy requirements described in this section. In the following, I discuss the possibilities the analysis team has to complete and adjust the generated requirements based on the different kinds of privacy requirements. The adjustment of privacy requirements related to the protection goal security is explained in Section 12.3.1. Section 12.3.2 introduces how privacy requirements related to the protection goal unlinkability can be adjusted. I present how the analysis team may adjust transparency requirements in Section 12.3.3 and intervenability requirements in Section 12.3.4.

12.3.1. Adjusting Requirements Related to the Protection Goal Security

I explain the possible adjustments to data confidentiality requirements in Section 12.3.1.1, and those for integrity and availability requirements in Section 12.3.1.2.

12.3.1.1. Adjusting Data Confidentiality Requirements

The generated data confidentiality requirements just reflect the elicited availability of personal data at the domains of the system elicited during the step Data Flow Analysis. Hence, a modification of the generated data confidentiality requirements possibly implies an inconsistency to the elicited personal data flows. This is the case when the modification leads to more restrictive data confidentiality requirements, that do not allow flows of personal data that were previously elicited. Modifications that weaken data confidentiality requirements, i.e., allowing more personal data to be available to the counterstakeholders or with a higher availability, are allowed to be performed by the analysis team. The adjusted data confidentiality requirements are checked for such inconsistencies in the step Validate privacy requirements (see Section 12.4).

The analysis team may also decide to change the attribute repudiation of a data confidentiality requirement if the counterstakeholders shall be able to prove that the personal data available to them are correct to nonRepudiation, or the data subject shall be able to plausibly deny that the personal data are correct to plausibleDeniability.

Application to the Running Example The attribute repudiation of the data confidentiality requirement shown in Figure 12.3 can, e.g., be adjusted to nonRepudiation to specify that financial employees shall be able to prove that the patient billing details and treatment costs provided to them are correct.

12.3.1.2. Adjusting Integrity and Availability Requirements

For the generated availability and integrity requirements, the analysis team may reduce the number of counterstakeholders that shall not be able to negatively influence the availability or integrity of the stakeholders personal data if they do not expect harm from these.

Application to the Running Example I decided not to change the generated integrity and availability requirements for the patient.

12.3.2. Adjusting Requirements Related to the Protection Goal Unlinkability

The generated unlinkability requirements do not have to be completed, because all attributes of them are already automatically set. The analysis team has only to decide whether the identified requirements really reflect the privacy protection needs, and whether a specific level of repudiation is needed. Similar to data confidentiality requirements, strengthening unlinkability requirements leads to inconsistencies to the elicited flows of personal data through the system, while a weakening unlinkability requirements does not introduce inconsistencies.

The possible adjustments to undetectability requirements is discussed in Section 12.3.2.1. How the analysis team may change anonymity and data unlinkability requirements is explained in Section 12.3.2.2.

12.3.2.1. Adjusting Undetectability Requirements

For each undetectability requirement with availability none, the analysis may decide that not all of the personal data of the data subject *ds* listed in the requirement have to be undetectable for the counterstakeholder *c*. That is, a counterstakeholder *c* may be allowed to know that specific data exist, but he or she shall not be able to know the exact content of the data. Hence, the analysis team can decide to introduce a data confidentiality requirement for *ds* and *c* with availability none. The personal data of *ds* that only have to be kept confidential and not undetectable, can then be moved from the corresponding undetectability requirement to the new data confidentiality requirement. Additionally, the moved personal data may be added to an undetectability requirement for *ds* and *c* with availability all, to make explicit that *c* is allowed to know whether the personal data exist or not. It is possible that the analysis team decides that all personal data shall only be kept confidential. Then the undetectability requirement is completely replaced by the introduced data confidentiality requirement.

For each undetectability requirement with an availability different from none, the ProPAn tool already generated a data confidentiality requirement with the same data subject, counterstakeholder, personal data, and availability. For these undetectability requirements, the analysis team may also decide that the counterstakeholders are allowed to know whether the personal data exist or not, and consequently delete the personal data from the undetectability requirements that do not need to be undetectable, or to delete the whole undetectability requirement. The personal data that do not need to be undetectable may then also be added to a respective undetectability requirement with availability all.

Application to the Running Example For the data subject patient and the counterstakeholder financial employee, the undetectability requirements shown in Figure 12.5 were generated. As it is not possible and needed to hide the information that the EHS processes the personal data listed in the undetectability requirement UUP3FE from the financial employee, UUP3FE is replaced by the data confidentiality requirement SDP3FE with the same attributes (except the description). Additionally, the personal data listed in UUP3FE are added to the undetectability requirement

UUP2FE, to make explicit that financial employees are allowed to know about the existence of these data.

12.3.2.2. Adjusting Anonymity and Data Unlinkability Requirements

For each anonymity and data unlinkability requirement, the analysis team has to consider whether linkability of the contained personal data to the data subject, or of the links between personal data were correctly derived from the ProPAn model. The analysis team can decide to weaken or strengthen the requirements by increasing or reducing the linkability with which personal data can be linked to the data subject or a pair of personal data can be linked to each other by the counterstakeholder. As mentioned before, strengthening the requirements may lead to inconsistencies between the requirements and the elicited data flows. These are identified during the step Validate privacy requirements (see Section 12.4).

Another possibility to adjust anonymity requirements is that the analysis team refines an anonymity requirement or a part of it to a pseudonymity requirement.

An anonymity requirement with linkability single can be translated into a pseudonymity requirement with kind person or role. The pseudonym kind person means that for every individual only one pseudonym exists. Hence, all personal data can be linked to the single individual they belong to if the relation between pseudonym and individual is known. A role pseudonym is used for specific roles. That is, an individual has a single pseudonym for each role. As I consider biddable domains as data subjects and biddable domains normally represent roles of individuals, a role pseudonym allows the same level of linkability as a person pseudonym in my situation. This is, because I do not distinguish further roles that a biddable domain may have.

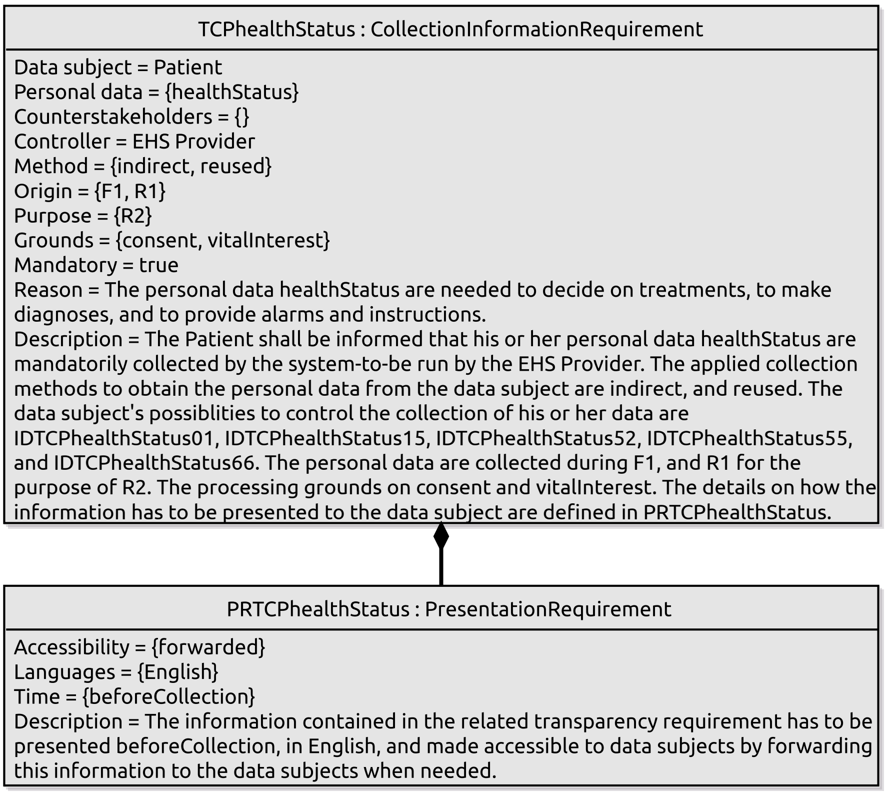
An anonymity requirement with linkability smallGroup, mediumGroup, or largeGroup may be translated into a pseudonymity requirement with kind relationship or roleRelationship. Due to the consideration of biddable domains as data subjects, relationship and roleRelationship pseudonyms are also equivalent for my case. The pseudonym kind relationship means that for every communication partner another pseudonym is used. For example, only the messages sent to the same partner can be linked to each other using a relationship pseudonym and not all communication.

An anonymity requirement with linkability anonymous may be translated into a pseudonymity requirement with kind transaction. A transaction pseudonym is only used once for one action or data that are related to an individual. Hence, the pseudonyms themselves do not provide any link to the individual they belong to. For more details on the kinds of pseudonyms see (Pfitzmann and Hansen, 2010).

Application to the Running Example I decide not to change or refine the anonymity and data unlinkability requirements for the patient and the financial employee shown in Figures 12.9 and 12.7. The personal data treatment costs and patient billing details have to be available to the financial employee with the possibility to link them to the single patient they belong to. Additionally, the available data have to be linkable to each other with linkability single. This is, because financial employees need this information to perform the billing of the patients' invoices.

12.3.3. Adjusting Requirements Related to the Protection Goal Transparency

Not all attributes of the transparency requirements were automatically filled during the generation of them. Hence, the analysis team has to complete the generated transparency requirements manually. The possible adjustments to processing information requirements are explained in Section 12.3.3.1, and those for exceptional information requirements in Section 12.3.3.2.



TCPhealthStatus : CollectionInformationRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {}
Controller = EHS Provider
Method = {indirect, reused}
Origin = {F1, R1}
Purpose = {R2}
Grounds = {consent, vitalInterest}
Mandatory = true
Reason = The personal data healthStatus are needed to decide on treatments, to make diagnoses, and to provide alarms and instructions.
Description = The Patient shall be informed that his or her personal data healthStatus are mandatorily collected by the system-to-be run by the EHS Provider. The applied collection methods to obtain the personal data from the data subject are indirect, and reused. The data subject's possiblities to control the collection of his or her data are IDTCPhealthStatus01, IDTCPhealthStatus15, IDTCPhealthStatus52, IDTCPhealthStatus55, and IDTCPhealthStatus66. The personal data are collected during F1, and R1 for the purpose of R2. The processing grounds on consent and vitalInterest. The details on how the information has to be presented to the data subject are defined in PRTCPhealthStatus.

PRTCPhealthStatus : PresentationRequirement

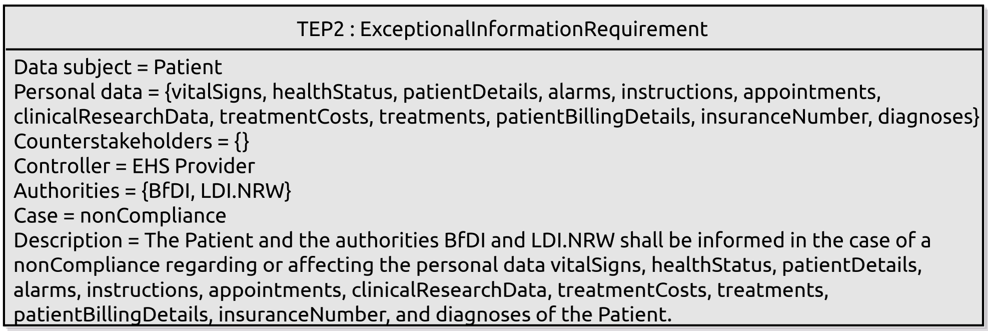
Accessibility = {forwarded}
Languages = {English}
Time = {beforeCollection}
Description = The information contained in the related transparency requirement has to be presented beforeCollection, in English, and made accessible to data subjects by forwarding this information to the data subjects when needed.

Figure 12.16.: *Adjusted collection information requirement for the data subject patient and the personal data health status*

12.3.3.1. Adjusting Processing Information Requirements

For the processing information requirements, the analysis team may revise and complete the processing grounds on which the personal data are processed, whether it is mandatory to provide the personal data to use the services provided by the machine, the reason for the processing of the personal data, who the controller of the machine is, and the measures taken by the controller to protect the processed personal data (cf. Figure 7.6 on page 112). The analysis team can also add, delete, or merge transparency requirements if needed. The consistency of these modifications to the elicited personal data flows is then checked in the step Validate privacy requirements.

Application to the Running Example The completed version of the collection information requirement TCPhealthStatus (see Figure 12.10) is shown in Figure 12.16. The associated presentation requirement states that the information contained in TCPhealthStatus has to be presented before the data are collected from the data subject in English and made accessible to data subjects by *forwarding* the information to them. Note that the attribute measures is not set, because yet no concrete measures were selected to protected the personal data, but when such mechanisms have been selected, the attribute should be set (see Chapter 17). To the processing grounds, I added vitalInterest, because the EHS shall help to maintain and improve the patients' health and I added a reason why the healthStatus needs to be collected. As controller of the EHS, I added the person EHS Provider. The related intervenability requirements (controlOptions) are not shown in Figure 12.16 for the sake of simplicity.



TEP2 : ExceptionalInformationRequirement

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, diagnoses}
Counterstakeholders = {}
Controller = EHS Provider
Authorities = {BfDI, LDI.NRW}
Case = nonCompliance
Description = The Patient and the authorities BfDI and LDI.NRW shall be informed in the case of a nonCompliance regarding or affecting the personal data vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, and diagnoses of the Patient.

Figure 12.17.: *Adjusted exceptional information requirements for the data subject patient and the case nonCompliance*

12.3.3.2. Adjusting Exceptional Information Requirements

The analysis team has to manually complete the generated exceptional information requirements. The attributes that have to be set for each exceptional information requirement are the controller of the system-to-be and the authorities that the controller has to inform. The European Commission provides a list of the data protection authorities of its member states[1]. This list can be used to determine relevant data protection authorities for the system-to-be.

There are no further possibilities to adjust the generated exceptional information requirements, because informing supervisory authorities about the processing of personal data and data breaches is mandatory if the processing shall comply with the EU Data Protection Regulation.

12.3.3.2.1. Application to EHS Example We instantiate the controller with the EHS Provider, as we also did for the collection information requirement shown in Figure 12.16. Furthermore, as relevant authorities, I added the *Bundesbeauftragte für den Datenschutz und die Informationsfreiheit* (BfDI) (the German data protection authority) and the *Landesbeauftragte für Datenschutz und Informationsfreiheit Nordrhein-Westfalen* (LDI.NRW) (the data protection authority of the state North Rhine-Westphalia), because I assume that the EHS shall be operated and used in North Rhine-Westphalia. The adjusted exceptional information requirement for the case nonCompliance is shown in Figure 12.17.

12.3.4. Adjusting Requirements Related to the Protection Goal Intervenability

The automatically generated requirements have to be adjusted manually by the analysis team, because not all attributes of the requirements can be set automatically and it is possible that some of the generated requirements are too strong. In the following, we describe the analysis team's possibilities to adjust the generated requirements in conformance with the EU General Data Protection Regulation.

12.3.4.1. Data Subject Intervention Requirement

The GDPR allows some adjustments of the generated intervention requirements for data subjects.

Article 6 of the GDPR (European Commission, 2016) states that the processing of personal data shall be based on the data subject's consent, but it provides a list of circumstances under

[1] `http://ec.europa.eu/justice/article-29/structure/data-protection-authorities/index_en.htm` (accessed on 25 May 2018)

which it is possible to process personal data without the explicit consent of the data subject. These are that the processing is necessary

1. *"for the performance of a contract to which the data subject is a party"* (processing ground contract in the metamodel shown in Figure 7.7 on page 116),
2. *"for compliance with a legal obligation"* (processing ground regulation),
3. *"to protect the vital interest of the data subject"* (processing ground vitalInterest),
4. *"for the performance of a task carried out in the public interest"* (processing ground publicInterest),
5. *"for the purposes of the legitimate interests pursued by the controller or by a third party"* (processing ground controllerInterest), and
6. *"for the establishment, exercise or defense of legal claims"* (processing ground legalClaims).

Hence, the analysis team can decide to remove the processing ground consent from a processing information requirement if the analysis team adds at least one of the other grounds. Then the analysis team has also to remove the related data subject intervention requirements with the types doNotConsent and withdrawConsent. Additionally, the analysis team may change the effect of withdrawing consent to noProcessing or restrictedProcessing according to Article 17 if the processing is also based on other grounds including contract, regulation, vitalInterest, publicInterest, controllerInterest, or legalClaims.

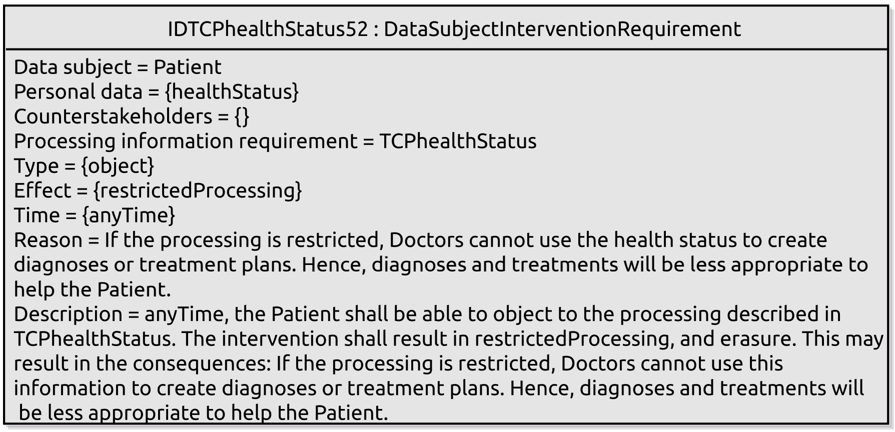
Article 11 states that if the controller processes personal data that do not allow to identify the related data subject, then the controller is not obliged to provide the data subject the rights to access, rectification, erasure, restriction of processing, and data portability, unless the data subject provides additional information that allow his or her identification. Hence, the analysis team can decide to remove data subject intervention requirements of type review, challengeAccuracy, challengeCompleteness, object, and requestDataCopy if the personal data of the related processing information requirement cannot be related to the individual data subject they belong to using the information provided by the system-to-be. The processing of aggregated data, instead of data that are linkable to the individual, is an example of a situation in which the analysis team can decide to remove data subject intervention requirements of the above mentioned types.

Article 17 allows the analysis team to remove data subject intervention requirements with type object and effect erasure if the processing grounds of the related processing information requirement include regulation, publicInterest, or legalClaims.

For all data subject intervention requirements, the analysis team has to specify the consequences that a data subject has to expect when he or she exercises his or her rights represented by the respective intervention requirement. This is already specified by the multiplicity 1 in Figure 7.7 on page 116.

Finally, the analysis team can strengthen the attribute time and decide to make the time at which a data subject can exercise his or her rights stricter. The linear ordering anyTime $<$ beforeCollection $<$ beforeUse $<$ atRecording $<$ beforeTransmission $<$ afterRecognition $<$ onRequest on the enumeration ActionTime specifies which literals are stricter then the others. The underlying semantics of this linear ordering is that an action time is smaller than another action time if it is stricter in the sense that it applies to more cases than the other (e.g., anyTime applies for all points in time) or will take place earlier in time (e.g., beforeCollection is earlier than beforeUse).

Note that if a data subject intervention requirement is removed, then also its related intervention information requirement is removed from the model, because it has no meaning without the related data subject intervention requirement.



IDTCPhealthStatus52 : DataSubjectInterventionRequirement

Data subject = Patient
Personal data = {healthStatus}
Counterstakeholders = {}
Processing information requirement = TCPhealthStatus
Type = {object}
Effect = {restrictedProcessing}
Time = {anyTime}
Reason = If the processing is restricted, Doctors cannot use the health status to create diagnoses or treatment plans. Hence, diagnoses and treatments will be less appropriate to help the Patient.
Description = anyTime, the Patient shall be able to object to the processing described in TCPhealthStatus. The intervention shall result in restrictedProcessing, and erasure. This may result in the consequences: If the processing is restricted, Doctors cannot use this information to create diagnoses or treatment plans. Hence, diagnoses and treatments will be less appropriate to help the Patient.

Figure 12.18.: *Adjusted data subject intervention requirement of type* object *for the collection information requirement* *TCPhealthStatus*

12.3.4.1.1. Application to EHS Example In the EHS example, I do not need to consider consent as a ground for the processing, because the personal data health status of patients are collected for purposes in their vital interest (cf. Figure 12.16). Hence, I could remove the data subject intervention requirements with type doNotConsent (see Figure 12.14) and withdrawConsent. Additionally, the collected health status could be important in the case of legal claims. Hence, I can also remove the effect erasure from the intervention requirement with type object, and I add legalClaims to the grounds of the collection information requirement for the personal data health status.

For the EHS machine (more precisely at the designed domain EHR), the personal data health status are linkable to the individual patient they belong to and hence, I cannot remove the remaining intervention requirement with type object and effect restrictedProcessing based on Article 11.

Furthermore, I have to provide details on the consequences that the intervention will have for data subjects (cf. Figure 7.7). The adjusted data subject intervention requirement of type object for the collection information requirement TCPhealthStatus shown in Figure 12.16 is shown in Figure 12.18.

12.3.4.2. Authority Intervention Requirement

The EU Data Protection Regulation provides no information about circumstances when the authority intervention requirements do not need to be considered. Hence, the analysis team should only modify the generated requirements after consultation of a legal expert in the field of data protection (if not already such an expert is part of the analysis team).

12.3.4.2.1. Application to EHS Example The generated authority intervention requirements are not changed, because I expect that the data protection authorities are allowed to make use of all their powers described in the GDPR.

12.3.4.3. Intervention Information Requirement

The analysis team has to set the controller of the system-to-be for each intervention information requirement. There are no further possibilities to adjust the generated intervention information requirements.

12.3.4.3.1. Application to EHS Example The controller of the intervention information requirements is also set to EHS Provider.

12.4. Validate Privacy Requirements

In this section, I discuss how the privacy requirements adjusted by the analysis team are validated by the ProPAn tool in the step Validate privacy requirements. I already introduced several validation conditions that instances of my privacy requirements taxonomy have to satisfy and that check the consistency between privacy requirements in Chapter 7. These validation conditions are expressed as OCL invariants on the ProPAn metamodel and integrated into it. In this way, the adjusted requirements can be validated using the EMF validation mechanism.

I introduce additional validation conditions to check the consistency between the adjusted privacy requirements and the availability of personal data at the domains of the system, the personal data flows documented in the ProPAn model, and the data protection needs implied by the GDPR (from which the privacy requirements were originally generated). The ProPAn tool is able to detect all inconsistencies discussed in the following and to inform the analysis team about them. The analysis team then has the possibility to correct the inconsistencies by performing the substep Adjust privacy requirements again (cf. Figure 12.1).

Note that all privacy requirements that are automatically generated by our method do not cause any errors or warnings due to the validation conditions (except the validation conditions that check whether the analysis team completed the attributes of the transparency requirements that cannot be derived from the ProPAn model). An inconsistency between the ProPAn model or the GDPR and the privacy requirements can only be introduced during the manual adjustment of the privacy requirements. Hence, the validation conditions check whether the adjusted privacy requirements are still consistent to each other, and to the ProPAn model and the data protection needs implied by the GDPR. I obtained these validation conditions by considering the possible inconsistencies that can be introduced by adjustments of the analysis team.

I distinguish two kinds of validation conditions. First, an adjustment of the analysis team introduces an inconsistency between the privacy requirements or between a privacy requirement and the ProPAn model or GDPR. In this case, the validation condition raises an error and the analysis team has to remove this inconsistency. Second, an adjustment can be consistent to the ProPAn model, but represent a weaker property than the one that we derived from the ProPAn method during the step Generate privacy requirements (see Section 12.2). In this case, we present a warning to the analysis team because they possibly incidentally weakened the privacy requirements. All validation conditions already introduced in Chapter 7 raise errors. Table A.1 on page 425 in Appendix A provides an overview of all validation conditions. Furthermore, Appendix A provides the formalization of all validation conditions as OCL invariants.

I introduce an additional general validation condition for all privacy requirements in Section 12.4.1. In Section 12.4.2, I introduce validation conditions for availability and integrity requirements. Additional validation conditions for all refinements of the class ConfidentialityRequirement are discussed in Section 12.4.3. I present the additional validation conditions for transparency and intervenability requirements in Sections 12.4.4 and 12.4.5, respectively.

12.4.1. Validating Privacy Requirements

For the adjusted privacy requirements, it has to be checked whether the analysis team completed all generated privacy requirements. The analysis team is informed which privacy requirements have unset attributes and have to be completed. The following validation condition is already encoded in the ProPAn metamodel by the multiplicities that specify which attributes have to be set.

VP2 All necessary attributes of a privacy requirement have to be set.

Application to EHS All generated privacy requirements have been completed and hence, no unset attributes are found that have to be set.

12.4.2. Validating Availability and Integrity Requirements

The analysis team may remove or add data from availability and integrity requirements during the previous step. To ensure that the analysis team did not incidentally remove personal data of a data subject that are processed by the machine, we have the following validation conditions. These validation conditions state that the machine has at least to maintain the availability and integrity of the personal data processed by it.

VSA2 All personal data of a data subject that are processed by the machine have to be contained in an availability requirement of the data subject.

VSI1 All personal data of a data subject that are processed by the machine have to be contained in an integrity requirement of the data subject.

Furthermore, I check whether the analysis team added personal data of a data subject to a corresponding availability or integrity requirement that are not processed by the machine. The machine cannot assure the integrity or availability of personal data that are not processed by it. Hence, I propose the following two validation conditions.

VSA3 All personal data contained in an availability requirement have to be processed by the machine.

VSI2 All personal data contained in an integrity requirement have to be processed by the machine.

Application to the Running Example

As I did not modify the generated availability and integrity requirements, the personal data assigned to them is exactly the personal data of the data subject that are processed by the machine.

12.4.3. Validating Confidentiality Requirements

In the previous step, the analysis team may have introduced inconsistencies between the confidentiality requirements (including unlinkability requirements; cf. Figure 7.5 on page 106) and the ProPAn model.

In the case that the analysis team strengthened an anonymity requirement (and analogously for data unlinkability requirements), then this introduces an inconsistency to the documented linkability in the ProPAn model. The analysis team may have strengthened an anonymity requirement in one of the following ways:

1. By deciding that the counterstakeholder shall only be able to link the personal data with a weaker linkability to the data subject.
2. By changing an anonymity requirement (or a part of it) to an undetectability requirement.
3. By changing an anonymity requirement (or a part of it) to a data confidentiality requirement.

Based on the available data diagram of the counterstakeholder, the analysis team has to decide whether their adjustments introduce too strong privacy requirements, or whether the functional requirements are not restrictive enough concerning the implied data flows. The following validation conditions check the described inconsistencies and raise errors when they are not satisfied.

VSD2 For each undetectability and data confidentiality requirement, the personal data referenced by it shall not be available at the referenced counterstakeholders.

VUA2 For each anonymity requirement, all personal data referenced by it have to be linkable to the data subject with a linkability weaker than or equal to the linkability mentioned in the anonymity requirement for all listed counterstakeholders.

VUD2 For each data unlinkability requirement, all pairs of personal data referenced by the requirement shall be linkable with a linkability weaker than or equal to the linkability of the data unlinkability requirement for all listed counterstakeholders.

If the analysis team deleted a confidentiality requirement that belonged to data subject ds and counterstakeholder c, then it might be the case that personal data p (or a pair of personal data) of ds do not occur in any of the other confidentiality requirements that belong to ds and c. In this case, we have no statement how p has to be protected against c. This might be intended by the analysis team, but nevertheless, the tool warns the analysis team about this situation.

VSC5 For all combinations of data subject ds and counterstakeholder c, check whether each personal data object of ds is referenced by at least one of the confidentiality requirements for ds and c.

VUD3 For all combinations of data subject ds and counterstakeholder c, check whether each link between personal data of ds that are available to counterstakeholder c is referenced by at least one data unlinkability requirement for ds and c.

Weakening unlinkability requirements does not introduce inconsistencies. For example, if counterstakeholder c is able to link personal data p with linkability mediumGroup to the data subject ds and the related anonymity requirement allows that c is able to link p to ds with linkability single, then the system-to-be satisfies this anonymity requirement. However, I decided to warn the analysis team in the case of an unlinkability requirement that is weaker than it would need to be. This is done by the following two validation conditions.

VUA3 For each anonymity requirement, all personal data referenced by it should be linkable to the data subject with a linkability stronger than or equal to the linkability mentioned in the anonymity requirement for all listed counterstakeholder.

VUD4 For each data unlinkability requirement, all pairs of personal data referenced by the requirement should be linkable with a linkability stronger than or equal to the linkability of the data unlinkability requirement for all listed counterstakeholder.

Application to the Running Example

The validation conditions are all satisfied by the adjusted confidentiality requirements, because the initially generated confidentiality requirements complied to the validation conditions and I only weakened the undetectability requirement UUP3FE to a respective confidentiality requirement.

12.4.4. Validating Transparency Requirements

The analysis team might have changed the transparency requirements inconsistently by adding personal data to transparency requirements that are not collected or stored by the machine, or do not flow to the specified domain. These inconsistencies represent errors that have to be corrected by the analysis team and are represented by the following validation conditions.

VTP2 All personal data referenced by a collection or storage information requirement have to be available at a designed domain.

VTF2 All personal data referenced by a flow information requirement have to be available at the target of the personal data flow.

The analysis team could have deleted transparency requirements or removed personal data from them. This would lead to an inconsistency between the transparency requirements and the transparency needs that can be derived from the ProPAn model as described in Section 12.2.3. The following three validation conditions check whether such inconsistencies exist.

VTC2 For all personal data flows from a given domain to a designed domain, a corresponding collection information requirement shall exist.

VTS2 For all personal data available at a designed domain, a corresponding storage information requirement shall exist.

VTF3 For all personal data flows from a designed domain to a given domain, or between two given domains due to a functional requirement, a corresponding flow information requirement shall exist.

Considering the GDPR as source for transparency and intervenability requirements, I obtain further validation conditions that check the consistency of transparency requirements and their relations to the intervention requirements. These validation conditions aim at validating the transparency requirements for compliance to the EU General Data Protection regulation. This is, I check whether only the adjustments described in Sections 12.3.3 and 12.3.4 were performed on the transparency requirements.

From Article 6 of the GDPR, I derive that the grounds of a processing information requirement shall contain consent, unless the grounds include contract, regulation, vitalInterest, publicInterest, controllerInterest, or legalClaims. Hence, I obtain the following validation condition.

VTP3 If a processing information requirement does not include consent as processing ground, it has to list contract, regulation, vital interest, public interest, controller interest, or legal claims as processing ground.

Article 6 and 7 imply that if processing is based on consent, then the data subject shall have the options to not consent and to withdraw previously given consent. This is specified by validation condition VTP1 given in Listing 7.2.5.3 on page 116. On the other hand, if the options to not consent and to withdraw previously given consent are presented to a data subject, then this processing is based on consent and should have consent as a processing ground. This leads to the following validation condition.

Listing 12.20: *Additional attribute for the class* PrivacyRequirement *representing whether personal data referenced by a privacy requirement is identifiable personal data for the controller*

```
context PrivacyRequirement
def: identifiableData : Boolean =
  Domain.allInstances()→select(d|d.domain.designed)→exists(domain|
    domain.relatedClosure(self.dataSubject)→exists(t|
      self.personalData→includes(t.d) and
      self.l = Linkability::single))
```

VTP4 If a processing information requirement has as control options data subject intervention requirements with type doNotConsent and withdrawConsent, then its processing grounds should include consent.

As discussed in the previous section, Article 11 states that controllers are not forced to provide intervention options to data subjects if the controller is not able to identify the data subject from the personal data that are processed. Hence, the following validation conditions include as a precondition that the controller is able to identify the data subject to which the concerned personal data belong. The additional operation identifiableData for the class PrivacyRequirement returns true iff personal data referenced by the privacy requirement are linkable to the data subject at a designed domain with linkability single (see Listing 12.20).

Due to Article 15, data subjects shall be able to access all their personal data that are stored and provided to others. Hence, I obtain the following validation condition.

VTP5 For each storage and flow information requirement about personal data that the controller can uniquely link to the data subject, the requirement has to have a data subject intervention requirement with type access as control option.

From Article 16, I derived the right of data subjects to challenge the accuracy and completeness of their stored personal data and personal data provided to others. This leads to the following validation condition.

VTP6 For each storage and flow information requirement about personal data that the controller can uniquely link to the data subject, the requirement has to have data subject intervention requirements with types challengeAccuracy and challengeCompleteness as control options.

The right to erasure (Article 17) says that the effect of objection to the processing of personal data shall be the erasure of these data, unless the processing of the concerned personal data is necessary for other legitimate purposes, e.g., compliance to legal obligations. Hence, I get the following validation condition.

VTP7 For each processing information requirement about personal data that the controller can uniquely link to the data subject and whose grounds include neither regulation, public interest, nor legal claims, the requirement has to have a data subject intervention requirement with type object and effect erasure as control option.

Article 18 describes the right to restrict the processing of personal data. Hence, I obtain the following validation condition.

VTP8 For each processing information requirement about personal data that the controller can uniquely link to the data subject, the requirement has to have a data subject intervention requirement with type object and effect noProcessing or restrictedProcessing as control option.

The existence of an intervention option related to the right to data portability (Article 20) is checked by the following validation condition.

VTC3 For each collection information requirement about personal data that the controller can uniquely link to the data subject and whose grounds do not include public interest, the requirement has to have a data subject intervention requirement with type requestDataCopy as control option.

Article 33 is concerned with the occurrence of data breaches and the duties of controllers in the case of such. From this Article, I derived the following validation condition.

VTE2 For each personal data of a data subject processed by the machine, an exceptional information requirement with case dataBreach has to exist for the data subject and the personal data.

From Article 58, I have deduced in Section 12.2.4.2 the intervention options of supervisory authorities. The following validation conditions check whether these intervention options are reflected in the model.

VTE3 For each personal data of a data subject processed by the machine, an exceptional information requirement with case authorityRequest has to exist for the data subject and has to include the personal data.

VTE4 For each exceptional information requirement with case authorityRequest, its authority intervention requirements have to include an authority intervention requirement with type obtainAccess.

VTE5 For each personal data of a data subject processed by the machine, an exceptional information requirement with case nonCompliance has to exist for the data subject and has to include the personal data.

VTE6 For each exceptional information requirement with case nonCompliance, its authority intervention requirements have to include authority intervention requirements with the types suspendDataFlows, orderBanOfProcessing, orderErasure, and orderRectification.

Application to the Running Example

As I only completed the transparency requirements and adjusted them according to the guidelines provided in Section 12.3, all validation conditions for transparency requirements are satisfied.

12.4.5. Validating Intervenability Requirements

To check whether the analysis team's modifications on the generated intervenability requirements still comply to the GDPR, I propose validation conditions that allow an automatic validation of the consistency of the adjusted intervenability requirements.

From Articles 6, 7, and 9, I derived constraints on the time when data subjects shall be able to exercise their intervention options. The following validation conditions make use of the linear ordering anyTime < beforeCollection < beforeUse < atRecording < beforeTransmission < afterRecognition < onRequest on the enumeration ActionTime (see Listing 12.21).

VID3 Each data subject intervention requirement that is a control option of a collection information requirement has to have a time that is smaller than or equal to beforeCollection.

Listing 12.21: *Min operation for the enumeration ActionTime*

```
context ActionTime
def: min(l : ActionTime) : ActionTime =
  if Set{self, l}→includes(ActionTime::anyTime) then
    ActionTime::anyTime
  else
    if Set{self, l}→includes(ActionTime::beforeCollection) then
      ActionTime::beforeCollection
    else
      if Set{self, l}→includes(ActionTime::beforeUse) then
        ActionTime::beforeUse
      else
        if Set{self, l}→includes(ActionTime::atRecording) then
          ActionTime::atRecording
        else
          if Set{self, l}→includes(ActionTime::beforeTransmission) then
            ActionTime::beforeTransmission
          else
            if Set{self, l}→includes(ActionTime::afterRecognition) then
              ActionTime::afterRecognition
            else
              ActionTime::onRequest
            endif
          endif
        endif
      endif
    endif
  endif
```

VID4 Each data subject intervention requirement that is a control option of a flow information requirement has to have a time that is smaller than or equal to beforeTransmission.

VID5 Each data subject intervention requirement that is a control option of a storage information requirement has to have a time that is smaller than or equal to atRecording.

The right of access by the data subject (Article 15) implies the following validation conditions that prescribe to which processing information requirements a data subject intervention requirement with type review may be associated, and that its time has to be anyTime.

VID6 Each data subject intervention requirement with type review has to be a control option of a storage or flow information requirement.

VID7 Each data subject intervention requirement with type review has to have the time anyTime.

From Article 16, I derived that data subjects can challenge the accuracy and completeness of their stored personal data and personal data provided to others at any time. Hence, I obtain the following two validation conditions.

VID8 Each data subject intervention requirement with type challengeAccuracy or challengeCompleteness has to be a control option of a storage or flow information requirement.

VID9 Each data subject intervention requirement with type challengeAccuracy or challengeCompleteness has to have the time anyTime.

From Articles 7, 18, 20, and 21, I derived a further validation condition concerning the time when data subjects shall be able to exercise their intervention options.

VID10 Each data subject intervention requirement with type withdrawConsent, object, or requestDataCopy has to have the time anyTime.

The right to erasure, also known as the right to be forgotten (Article 17), says that the effect of withdrawing consent to the processing of personal data shall be the erasure of these data, unless the processing of the concerned personal data is necessary for other legitimate purposes, e.g., compliance to legal obligations. Hence, I get the following validation condition.

VID11 For each data subject intervention requirement with type withdrawConsent that is a control option of a processing information requirement whose grounds include neither contract, regulation, vital interest, public interest, controller interest, nor legal claims, the effects have to include erasure.

12.4.5.0.1. Application to the EHS Example All instances of intervention requirements satisfy the defined validation conditions, as these were modified in accordance to the guidelines for step Adjust privacy requirements discussed in Section 12.3.

12.5. Comparison to the State of the Art

In this section, I discuss briefly the state of the art methods that I introduced in Chapter 3 and that explicitly support the elicitation of privacy requirements. The state of the art approaches consider different privacy conceptual models and hence identify diverse kinds of privacy requirements. I discuss how the different privacy conceptual models are related to my privacy requirements taxonomy in Section 7.3.

Most privacy requirements engineering methods are based on a manual identification of privacy requirements based on guidelines. Antignac et al. (2016) provide guidelines to annotate DFDs with privacy principles. Perera et al. (2016) support the identification of privacy principles relevant for an IoT application. Oetzel and Spiekermann (2014) propose to identify relevant privacy principles and refinements of these for a system-to-be based on a list of privacy principles and refinements. Kalloniatis et al. (2008) propose a list of privacy goals that the analysis team has to consider for the organizational goals of the system. Deng et al. (2011) suggest relevant privacy protection goals that have to be considered for specific elements of a data flow diagram. Islam et al. (2010) propose to derive privacy goals from laws and regulations, and to add these to a goal model. For this, they use the privacy goal taxonomy of Antón and Earp (2004). Bellotti and Sellen (1993) elicit feedback and control requirements related to the protection goals transparency and intervenability based on four categories concerning personal data processing with guiding questions. Degeling et al. (2016) do not follow a specific privacy conceptual model, but propose that privacy experts annotate a given system model with comments concerning privacy supported by predefined questions. These comments may be interpreted as privacy requirements. Oliver (2016) identifies privacy requirements from data flow diagrams that are annotated with information about the processed data, the type of data, the purpose of data processing, the usage of the data, and the data's security classification. Senarath et al. (2017) propose to elicit the privacy requirements from the end-users (data subjects) of the system-to-be by performing a user-survey.

All these methods have in common that they are not automated and do not support the validation of manually identified privacy requirements. Additionally, I have shown in Section 7.3 that my privacy taxonomy is able to express all concepts used in the other privacy conceptual models and even provides more expressive privacy requirements than the other conceptual models. The higher expressiveness originates from the attributes of the privacy requirements of my taxonomy that allow to express detailed privacy requirements in a structured format.

Mead et al. (2011) propose a questionnaire that has to be filled in by the analysis team with questions concerning the data processing. Based on the answers of the questionnaire, the PRET tool returns a list of privacy requirements deduced from privacy principles and privacy regulations. However, these privacy requirements are not tailored to a system model and hence, are more abstract than the privacy requirements generated by the ProPAn tool. Furthermore, Mead et al. do not support an adjustment or refinement of the generated privacy requirements.

I can conclude that the automatic generation of privacy requirements based on and tailored to a system model is a novel contribution to the state of the art. The allowed adjustment and automated validation of the adjusted privacy requirements is also not present in the other state of the art methods. Another novelty of the proposed identification of privacy requirements is that the privacy requirements have explicit links to the requirements specification and system model. These traceability links make the it more transparent and comprehensible why specific privacy requirements were created, and how the system-to-be shall process personal data. This also contributes to a better understanding of what the system-to-be shall do.

12.6. Conclusions

In this chapter, I have shown how instances of my privacy requirements taxonomy introduced in Chapter 7 can automatically be created and validated using the ProPAn tool. This generation and validation is based on the information elicited and documented in the ProPAn model during the previous two steps Privacy Threshold Analysis (see Chapter 10) and Data Flow Analysis, and the legal requirements implied by the EU General Data Protection Regulation. Especially, the information about

1. which data represent personal data of a data subject,
2. at which domains of the system the personal data are available, and
3. how the personal data flow through the system

are used for the generation and validation of the privacy requirements. Furthermore, I provide guidelines for the analysis team to modify the generated privacy requirements in compliance to the GDPR and consistent to the information documented in the ProPAn model. To check whether the analysis team has correctly applied these guidelines, I proposed validation conditions that allow to identify inconsistencies

1. in the privacy requirements,
2. between privacy requirements,
3. between the privacy requirements and the information documented in the ProPAn model,
4. and between the privacy requirements and the data protection needs implied by the GDPR.

In this way, I provide a systematic, tool-supported, and to a large extent automated method to derive the privacy requirements on a system-to-be based on its functional requirements. This is a novel contribution to the state of the art.

The next step of the ProPAn method is the Privacy Risk Analysis (see Chapter 13). During the Privacy Risk Analysis, the analysis team identifies threats to the documented privacy requirements and has to evaluate the privacy risks implied by these. To identify the threats, the analysis team assesses deviations of the normal behavior of the system by considering the documented functional requirements.

Chapter 13

Identification and Evaluation of Privacy Threats

In this chapter, I introduce the step Privacy Risk Analysis of the ProPAn method. In this step, the analysis team identifies threats to the privacy requirements elicited and documented in the previous step of the ProPAn method. I propose to identify possible deviations for each functional requirement and to assess whether these deviations result in privacy threats. After this *local* identification of privacy threats, the analysis team has to connect the identified threats, more precisely, their causes, consequences, and impact on the privacy requirements, to each other. In this way, the analysis team obtains a *global* view on the privacy threats for the system-to-be. Based on this global view, the analysis team can evaluate the privacy risks implied by the identified privacy threats.

The identification of privacy threats from deviations of the functional requirements is based on the paper (Meis and Heisel, 2017c) of which I am the main author. Maritta Heisel provided valuable feedback to the content of the paper that improved it.

In Section 13.1, I provide an introduction to this chapter. I explain how the functional requirements are systematically assessed to identify deviations from the normal behavior of the system that may lead to privacy threats in Section 13.2. The global assessment of the identified privacy threats is introduced in Section 13.3. The final evaluation of the privacy risks implied by the identified privacy threats is presented in Section 13.4. In Section 13.5, I compare the step Privacy Threat Analysis of the ProPAn method with the state of the art methods introduced in Chapter 3. This chapter is concluded in Section 13.6.

13.1. Introduction

Having identified the privacy requirements that a system has to address, the analysis team has mainly two options to operationalize these (cf. Chapter 3). First, the analysis team can refine the privacy requirements to functional requirements that implement these, e.g., by selecting specific privacy enhancing technologies. This option is described in more detail in Chapter 17, which introduces the step Privacy Measure Integration (see also Chapter 8). The selected technologies and non-technical measures that shall be integrated into the system may have severe consequences on the system, its functionality and properties, and the costs and effort to implement the system. Hence, the second option is to assess the possible risk of a violation of the privacy requirements. Based on the severity of the identified privacy risks, the analysis team can better decide on whether or which measures have to be integrated into the system to appropriately address the privacy requirements. That is, if it is unlikely that a privacy requirement is significantly harmed, the analysis team may accept this risk and decide not to integrate a measure mitigating the risk into the system. On the other hand, likely events that significantly harm privacy requirements are not acceptable and have to be treated.

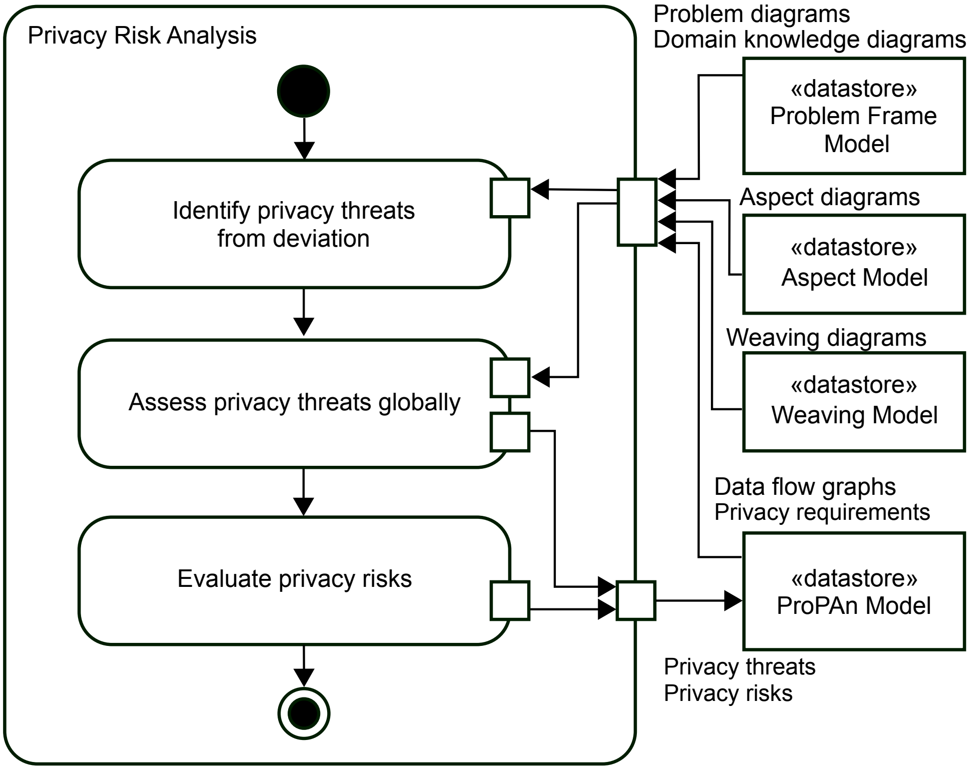


Figure 13.1.: *Detailed view on the step* Privacy Risk Analysis *of the ProPAn method*

Different legislators, e.g., the European Union in the EU General Data Protection Regulation (European Commission, 2016), prescribe that data protection or privacy impact assessments (PIAs) are performed for all kinds of projects involving the processing of personal data. The goal of a PIA is to assess the implications of the project on the data subjects' privacy. A central element of a PIA is the identification and evaluation of privacy threats to estimate the privacy risks implied by the considered project. In my thesis, I focus on software projects, and I want to assist the analysis team to identify and evaluate the privacy threats of the system as early as possible during the development process, namely in the requirements engineering phase.

Figure 13.1 shows the substeps of the step Privacy Risk Analysis. In the first substep Identify privacy threats from deviation, I propose to assess each functional requirement (documented in problem or aspect diagrams) in isolation to identify privacy threats implied by deviations from the system's behavior to achieve the functional requirement. For this substep, I adopt the Hazard and Operability (HAZOP) studies (IEC, 2001), which have successfully been used to assess the safety implications of a system, to a systematic methodology called Privacy and Operability (PRIOP) studies. In the second substep Assess privacy threats globally, the analysis team has to provide a global view on the privacy threats identified for the different functional requirements and their relations. Similar or the same privacy threats may be identified for different functional requirements, and different privacy threats may depend on, or support each other. Hence, the analysis team's task is to provide a consistent view on the privacy threats to the privacy requirements of the system. If the step Privacy Risk Analysis was at least once performed and measures were integrated into the problem in the step Privacy Measure Integration, then these measures (represented by problem, aspect, and weaving diagrams) are added to the threat model to evaluate the risk reduction implied by them. In the last substep Evaluate privacy risks, the analysis team has to determine the likelihood with which the privacy threats may occur in the system and the consequences they may have on the privacy requirements. The combination of the likelihood of a threat to occur and the consequence that the threat has on a privacy requirement forms a privacy risk. These privacy risks are an input for the step Privacy Measure Integration introduced in Chapter 17 (cf. Chapter 8).

13.2. Identify Privacy Threats from Deviations

PRIOP aims at a systematic privacy and operability analysis of a software project. Figure 13.2 visualizes the central steps (arrows) of a PRIOP study and the created artifacts (boxes). First, the software project has to be *decomposed* into subproblems. PRIOP does not prescribe how the decomposition is achieved. In my thesis, I use problem diagrams that represent a decomposition of the system-to-be following Jackson's problem frame approach. For each of these subproblems, I propose to create a Table for further analysis. This table should contain a short summary of the subproblem that is considered and should mention the names of the people involved in the PRIOP study of the subproblem. Then each subproblem is *categorized* based on its functionality. I use the information elicited during the step Data Flow Analysis (see Chapter 11) to categorize the subproblems to the processing categories *collection*, *storage*, *flow*, and *deduction*. For each identified category of the subproblem a Block, i.e., a subtable consisting of 12 rows (one for each guide word), is added to the subproblem's table. Finally, the PRIOP guide words have to be *considered* for every combination of subproblem and category. The consideration of a guide word results in a Row in the block of the considered category in the table of the considered subproblem.

In Section 13.2.1, I briefly introduce Hazard and Operability Studies as background of this section. How the subproblems are categorized to processing categories is explained in Section 13.2.2. I present the guide words to identify deviations of the subproblems in Section 13.2.3. I introduce the template to assess the deviations of functional requirements based on their processing categories and the proposed guide words in Section 13.2.4. To support the analysis team to decide which privacy requirements a deviation may impact, I provide a mapping between the deviations implied by the combination of processing category and guide word, and the privacy requirements these possibly harm in Section 13.2.5.

13.2.1. Hazard and Operability Studies

The international standard IEC 61882 (IEC, 2001) defines what a Hazard and Operability (HAZOP) study is and a process to perform a HAZOP study. HAZOP aims at identifying potential hazards and operability problems. A hazard is defined as the potential source of *"physical injury or damage to the health of people or damage to property or the environment"* (IEC, 2001) and an operability problem is any *deviation* from the intended behavior of the system that leads to non-conformance with its (functional) requirements. During a HAZOP study small parts of a system are analyzed in isolation. To systematically identify the potential hazards or operability problems of these parts, HAZOP proposes the eleven guide words NO, MORE, LESS, AS WELL AS, PART OF, REVERSE, OTHER THAN, EARLY, LATE, BEFORE, and

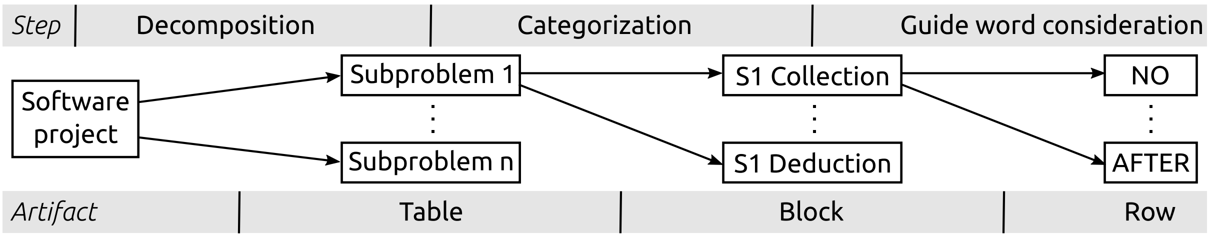


Figure 13.2.: *Steps and artifacts of a PRIOP study*

AFTER. These guide words are interpreted in the context of the behavioral characteristics of the part under consideration and lead to deviations of the intended behavior. The derived deviations for a part are documented together with the *possible causes* of the described situation, its *consequences* on the system's safety and operability, and *safeguards* that shall prevent the occurrence of this situation, or reduce the consequences the deviation may have in a template.

In this chapter, I adapt HAZOP to be used in the context of a privacy threat analysis to identify privacy threats implied by deviations instead of hazards.

13.2.2. PRIOP Processing Categories

During the analysis of the identified subproblems, I distinguish four categories of personal data processing that are performed by the machine. These categories are *collection*, *storage*, *flow*, and *deduction* of personal data. A subproblem is in the category *collection*, if it describes that data are collected by the machine from a given domain. Subproblems in the category *storage* are concerned with the storage of personal data at designed domains. If a subproblem causes a flow of personal data from the machine to a given domain, then it is in the category *flow*. Subproblems in the category *deduction* are concerned with the deduction or computation of personal data based on other data available to the machine.

A subproblem can be in none (i.e., it does not process personal data) or multiple of these categories, depending on the characteristics of the subproblem. This differentiation of processing categories helps to systematically assess the characteristics of a subproblem in order to identify privacy threats that it possibly causes. To refer to all of these categories simultaneously, I use the term *processing*.

Based on the information elicited during the step Data Flow Analysis, the ProPAn tool can automatically categorize the functional requirements. For the categorization of a functional requirement, I added additional operations to the class Statement of the ProPAn metamodel (see Figure 10.3 on page 159). The operations collectionOf, storageOf, flowOf, and deductionOf return the personal data that are collected, stored, flow to given domains, and are deduced due to the statement for a provided data subject, respectively. If one of these operations returns an empty set of personal data, then the statement is not categorized in the respective category. The domains from which the personal data are collected, at which the personal data are stored, to which given domains the personal data flow, and by which the personal data can be deduced are returned by the operations collectionFrom, storageAt, flowOf, and deductionBy for a provided data subject, respectively. These operations are formally specified in Listing 13.1. The operations for identifying the personal data collected, stored, and flowing to other domains due to the statement follow the same principles that I use to derive collection, storage, and flow edges in Section 10.5 and to generate collection, storage, and flow information requirements in Section 12.2.3. These principles are:

1. Personal data are collected due to the statement if they flow from a given domain to a designed domain due to the statement.

2. Personal data are stored due to the statement if they are available at a designed domain due to the statement.

3. Personal data flow due to the statement if they flow from a designed domain to a given domain due to the statement.

Whether personal data are derived due to a statement can be derived from the DerivedFrom relations (see Figure 11.6 on page 182) elicited during the Data Flow Analysis. This is, because the ProPAn tool automatically documents in the case that derived personal data flow from one domain to another from which statement this deduction was identified.

Listing 13.1: *OCL specification of additional operations for the class* Statements *that allow to automatically categorize a statement to the proposed processing categories*

```
context Statement
def: collectionOf(dataSubject : Person) : Set(Data) =
  dataSubject.relatedtos.personalData→select(pd|
    pd.pdavailableats→exists(pda1,pda2| pda1 <> pda2 and
      (not pda1.domain.domain.designed) and pda2.domain.domain.designed and
      pda1.purpose→includes(self) and pda2.origin→includes(self)))
def: collectionFrom(dataSubject) : Set(Domain) =
  self.collectionOf(dataSubject).pdavailableats→select(pda|
    (not pda.domain.domain.designed) and pda.purpose→includes(self)).domain

def: storageOf(dataSubject : Person) : Set(Data) =
  dataSubject.relatedtos.personalData.pdavailableats→select(pd|
    pd.domain.domain.designed and pd.origin→includes(self)).personalData
def: storageAt(dataSubject) : Set(Domain) =
  self.storageOf(dataSubject).pdavailableats→select(pda|
    pda.domain.domain.designed and pda.origin→includes(self)).domain

def: flowOf(dataSubject : Person) : Set(Data) =
  dataSubject.relatedtos.personalData→select(pd|
    pd.pdavailableats→exists(pda1,pda2| pda1 <> pda2 and
      pda1.domain.domain.designed and (not pda2.domain.domain.designed) and
      pda1.purpose→includes(self) and pda2.origin→includes(self)))
def: flowTo(dataSubject) : Set(Domain) =
  self.flowOf(dataSubject).pdavailableats→select(pda|
    (not pda.domain.domain.designed) and pda.origin→includes(self)).domain

def: deductionOf(dataSubject : Person) : Set(Data) =
  dataSubject.personalData→select(pd|
    pd.derivedfrom→exists(df|df.origin→includes(self)))
def: deductionBy(dataSubject) : Set(Domain) =
  self.deductionOf(dataSubject).derivedfrom→select(df|
    df.origin→includes(self)).derivedBy
def: deductionOriginal(dataSubject : Person) : Set(Data) =
  dataSubject.personalData→select(pd|
    pd.derivable→exists(df|df.origin→includes(self)))
```

Table 13.1.: *Processing categories for R3 with the respective processed personal data*

Category	Personal Data	from/at/to/by
Collection	patientBillingDetails, treatmentCosts, treatments	Doctor
Storage	patientBillingDetails, treatmentCosts, treatments	Invoice
Flow	diagnoses, insuranceNumber, treatments	Insurance Application
Deduction	treatmentCosts	Doctor

Application to EHS For the sake of simplicity, I focus on requirement R3 (see Chapter 4) for the identification of privacy threats. Table 13.1 shows the results of the evaluation of the previously introduced operations for requirement R3 and the patient as data subject. The patient billing details, treatment costs, and treatments are collected from doctors (see also Figure 11.16 on page 201). These data are stored at the designed domain invoice. The diagnoses, insurance number, and treatments flow to the insurance application. The treatment costs are derived by the doctor. Hence, R3 belongs to all processing categories.

13.2.3. PRIOP Guide Words

I consider all HAZOP guide words as useful to identify privacy threats, because these guide words describe in general the deviations that can occur in all kinds of processing a subproblem may be concerned with. I add one additional guide word, namely INCORRECT. This guide word shall cover the cases in which subproblems are performed incorrectly or with incorrect data as input.

Tables 13.2 and 13.3 show all PRIOP guide words and my deviation patterns for the four previously introduced processing categories. These deviation patterns are the starting point for the analysis team to assess the deviations of the subproblem they analyze. When these deviation patterns are used for a concrete subproblem, then the terms in angle brackets (⟨⟩) have to be instantiated for the subproblem. The term ⟨PD⟩ is instantiated with the personal data that are collected, stored, flow to given domains, or are deduced due to the subproblem. Hence, this term can be instantiated using the previously introduced operations collectionOf, storageOf, flowOf, and deductionOf. If a subproblem is in the category *flow*, then the term ⟨target⟩ has to be instantiated with the given domains to which the personal data flow. This term can be instantiated with the result of the previously introduced operation flowTo. The text template for the category *deduction* and the guide word REVERSE uses the placeholder ⟨original PD⟩ which refers to the personal data from which the deduced personal data are derived. This placeholder can also automatically be instantiated from the ProPAn model using the operation deductionOriginal (see Listing 13.1).

The other placeholders have to be instantiated manually by the analysis team and are hence printed in *italics*. The terms ⟨*other PD*⟩ and ⟨*additional PD*⟩ have to be instantiated with the personal data that are considered to be unintendedly collected, stored, flowing to given domains, or deduced. ⟨*other domain*⟩, ⟨*other target*⟩, and ⟨*additional target*⟩ have to be instantiated with the domains to which personal data unintendedly flow.

13.2.4. The PRIOP Template

The introduced guide words and deviation templates shall help the analysis team to identify deviations of the intended behavior of the processing a subproblem is concerned with. These deviations can lead to violations of privacy requirements and the subproblem's operability. In the case that an identified deviation leads to a violation of a privacy requirement, the deviation is a privacy threat. I propose a template that is based on the templates used in HAZOP studies (IEC, 2001).

An excerpt of the PRIOP template instance for R3 of the EHS example is shown in Tables 13.4 and 13.5. I omit the general information about the subproblem and the people involved in the PRIOP study. In Table 13.4, the *blocks* for the processing categories collection and storage are shown, while Table 13.5 contains the blocks for the processing categories flow and deduction. I selected one or two guide words for each processing category block to illustrate how the proposed template could be filled. For each processing category it is documented from, at, to, or by which domain which personal data are collected, stored, flow, or are deduced, respectively.

The first two columns show the considered guide word for the row and the deviations it can lead to for the processing category and the considered subproblem. The ProPAn tool can generate the deviation text automatically based on the text templates introduced in Tables 13.2 and 13.3. Hence, the ProPAn tool fills the cells of the first two columns, while keeping the others empty. As mentioned before, the placeholders ⟨PD⟩, ⟨target⟩, and ⟨original PD⟩ can be instantiated automatically, while the others have to be completed by the analysis team. The listed deviations possibly represent privacy threats or operability issues and have to be assessed by the analysis team.

In the third column, the analysis team has to document the causes that possibly lead to the

Table 13.2.: *Deviation patterns for the all combinations of proposed guide words and the processing categories collection and storage*

Deviation type	**Guide word**	**Deviation patterns for operation category**	
		Collection	**Storage**
Negative	NO	⟨PD⟩ are not collected	⟨PD⟩ are not stored
Quantitative modification	MORE	More ⟨PD⟩ are collected than intended, including collection of ⟨PD⟩ with additional methods, with higher linkability, in higher amount, or with higher availability.	More ⟨PD⟩ are stored than intended, including storage of ⟨PD⟩ with higher linkability, in higher amount, with higher availability, or with longer duration.
	LESS	Less ⟨PD⟩ are collected than intended, including collection of ⟨PD⟩ with less methods, with lower linkability, in lower amount, or with lower availability.	Less ⟨PD⟩ are stored than intended, including storage of ⟨PD⟩ with lower linkability, in lower amount, with lower availability, or with shorter duration.
Qualititative modification	AS WELL AS	In addition to ⟨PD⟩, ⟨*additional PD*⟩ are collected or in addition to the machine ⟨*other domains*⟩ collect the ⟨PD⟩.	In addition to ⟨PD⟩, ⟨*additional PD*⟩ are stored.
	PART OF	Only a part of ⟨PD⟩ are collected.	Only a part of ⟨PD⟩ are stored.
	INCORRECT	The collected ⟨PD⟩ are incorrect.	The stored ⟨PD⟩ are incorrect.
Substitution	REVERSE	⟨PD⟩ flow from machine to source of collection.	⟨PD⟩ are deleted.
	OTHER THAN	⟨*other PD*⟩ are collected instead of ⟨PD⟩.	⟨*other PD*⟩ are stored instead of ⟨PD⟩.
Time	EARLY	⟨PD⟩ are collected earlier than intended relative to clock time.	⟨PD⟩ are stored earlier than intended relative to clock time.
	LATE	⟨PD⟩ are collected later than intended relative to clock time.	⟨PD⟩ are stored later than intended relative to clock time.
Order or sequence	BEFORE	⟨PD⟩ are collected before another prior operation, e.g., collection before gaining consent.	⟨PD⟩ are stored before another prior subsequent operation, e.g., storing before gaining consent, or before anonymization.
	AFTER	⟨PD⟩ are collected after another subsequent operation, e.g., collection of up-to-date data after these were needed or the data subject withdrew consent.	⟨PD⟩ are stored after another subsequent operation, e.g., storage after another operation would have needed ⟨PD⟩, or data subject has withdrawn consent.

Table 13.3.: *Deviation patterns for the all combinations of proposed guide words and the processing categories flow and deduction*

Deviation type	**Guide word**	**Deviation patterns for operation category**	
		Flow	**Deduction**
Negative	NO	⟨PD⟩ do not flow to ⟨target⟩.	⟨PD⟩ are not deduced.
Quantitative modification	MORE	More ⟨PD⟩ flow to ⟨target⟩ than intended, including flow of ⟨PD⟩ with higher linkability, in higher amount, or with higher availability.	More ⟨PD⟩ are deduced than necessary, including deduction of ⟨PD⟩ with higher linkability, in higher amount, or with higher availability.
	LESS	Less ⟨PD⟩ flow to ⟨target⟩ than intended, including flow of ⟨PD⟩ with lower linkability, in lower amount, or with lower availability.	Less ⟨PD⟩ are deduced than necessary, including deduction of ⟨PD⟩ with lower linkability, in lower amount, or with lower availability.
Qualititative modification	AS WELL AS	In addition to ⟨PD⟩, ⟨*additional PD*⟩ flow to ⟨target⟩, or ⟨PD⟩ flow to an ⟨*additional target*⟩.	In addition to ⟨PD⟩, ⟨*additional PD*⟩ are deduced.
	PART OF	Only a part of ⟨PD⟩ flow to ⟨target⟩, or ⟨PD⟩ flow to fewer targets.	Only a part of ⟨PD⟩ are deduced.
	INCORRECT	The ⟨PD⟩ flowing to ⟨target⟩ are incorrect.	The deduced ⟨PD⟩ are incorrect.
Substitution	REVERSE	⟨PD⟩ or ⟨*other PD*⟩ flow from ⟨target⟩ to the machine.	⟨original PD⟩ are or can be deduced from ⟨PD⟩.
	OTHER THAN	⟨*other PD*⟩ flow to ⟨target⟩ instead of ⟨PD⟩ or ⟨PD⟩ flow to ⟨*other target*⟩.	⟨*other PD*⟩ are or can be deduced instead of ⟨PD⟩.
Time	EARLY	⟨PD⟩ flow earlier than intended to ⟨target⟩ relative to clock time.	⟨PD⟩ are deduced earlier than intended relative to clock time.
	LATE	⟨PD⟩ flow later than intended to ⟨target⟩ relative to clock time.	⟨PD⟩ are deduced later than intended relative to clock time.
Order or sequence	BEFORE	⟨PD⟩ flow before another prior operation to ⟨target⟩, e.g., sending before gaining consent, or before anonymization.	⟨PD⟩ are or can be deduced before another prior operation, e.g., deduction before gaining consent, or before anonymization.
	AFTER	⟨PD⟩ flow after another subsequent operation to ⟨target⟩, e.g., operation on ⟨target⟩ is performed before up-to-date ⟨PD⟩ were provided, or data subject has withdrawn consent.	⟨PD⟩ are or can be deduced after another subsequent operation, e.g., deduction after another operation would have needed ⟨PD⟩, or data subject has withdrawn consent.

Table 13.4.: *Excerpt of the instantiated template for functional requirement R3 (part 1)*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
Collection of treatmentCosts from Doctor				
NO	treatmentCosts are not collected	Doctor forgets to enter the costs of his or her treatments or does not save the changes made.	Doctor will not get paid and patients not billed.	-
MORE	Higher treatmentCosts are collected than intended.	Doctors incidentially enter costs for treatments not performed or too high treatment costs	Patients will get too high bills or are billed for treatments that they did not receive.	Integrity requirement: SIP (cf. Figure 12.4 on page 210)
Storage of treatments, patientBillingDetails, treatmentCosts at Invoice				
BEFORE	treatments, patientBillingDetails, and treatmentCosts are stored in the invoice before it is known which treatments are beared by the Patient’s insurance contract.	1. Doctor explicitly initiates the creation of an invoice without knowing whether the concerned treatments are beared by the Patient‘s insurance contract. 2. A software error causes the creation of the invoice before having the necessary information.	Patients will get too high or too low bills.	Integrity requirement: SIP

deviations. These possible causes are in most cases scenarios that explain how the deviation may occur, or situations or states of the system leading to it.

The last two columns are concerned with the consequences the deviations may have on the privacy requirements or the operability of the subproblem. The consequences are first documented as free text, then the harmed privacy requirements are explicitly listed. The consequences document unwanted situations or states reached because of the identified possible causes.

If the analysis team identified possible causes for a guide word and consequences that harm privacy requirements, then the deviation represents a privacy threat. Whether and how this threat has to be further assessed is determined by evaluating the likelihood with which the possible cause may happen and lead to the consequence, and the impact that the consequence has on the privacy requirement using a risk matrix that defines which combinations of likelihood and consequence of a threat are acceptable, tolerable, and unacceptable. This is done in the substep Evaluate Privacy Risks discussed in Section 13.4.

If the step Privacy Risk Analysis was already performed at least once, then the measures selected in the step Privacy Measure Integration (see Chapter 17) have also to be assessed by the analysis team for potential privacy threats. For these measures, I also propose to use the suggested guide words to assess which privacy threats may occur when the behavior of the selected privacy measure deviates from its normal behavior. In this way, the analysis team can assess whether the selected privacy measures sufficiently implement the privacy requirements, or introduce themselves privacy threats that need to be mitigated.

Table 13.5.: *Excerpt of the instantiated template for functional requirement R3 (part 2)*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
Flow of insuranceNumber, diagnoses, treatments to InsuranceApplication				
LESS	Less diagnoses and treatments flow to InsuranceApplication than the patients received, or it is not possible for the insurance application to link the diagnoses and treatments to the Patient‘s insuranceNumber.	1. Software error occurs before or during the transmission of diagnoses and treatments. 2. An error occurs during the transmission of diagnoses and treatments over the network	1. If too few diagnoses are transmitted to the insurance application or if diagnoses and insurance number are not linkable to the treatments, then the insurance application is not able to perform the accounting. This will result in higher bills. 2. If treatments that are not beared by the patient's insurance contract are not transmitted to the insurance application, then no invoice will be created for them.	Integrity requirement SIP
INCORRECT	The insuranceNumber, diagnoses, or treatments flowing to InsuranceApplication are incorrect.	1. Software error occurs and leads to a flow of incorrect insurance number, diagnoses, and treatments. 2. The insurance number, diagnoses and treatments are alread incorrectly stored in the EHR	Patients will get too high or low bills, or are billed for treatments that they did not receive or treatments that they received are not billed.	Integrity requirement SIP
Deduction of treatmentCosts by Doctor				
REVERSE	healthStatus and patientDetails can be deduced from treatmentCosts.	Treatment costs may allow to deduce the treatments performed, diagnoses, or insurance contracts if observed over a longer time especially if to some extent the date of deduction is known.	Financial employees that are able to observe the treatment costs over a longer time might be able to deduce the treatments, diagnoses, or insuranceContracts of specific Patients.	1. Data confidentiality requirement SDP3FE (cf. Section 12.3.2.1) 2.Exceptional information requirement TEP0 (cf. Section 12.2.3.4)

The identification of possible causes, consequences, and harmed privacy requirements is a creative process that strongly depends on the expertise of the analysis team. This task could be further assisted by providing libraries of known privacy threats. I supervised a bachelor (Barth, 2016) and a master thesis (Yu, 2017) that created a foundation for a model-based database to document known privacy threats. Such a database may be used by the analysis team as an input for the privacy risk analysis, because it supports the analysis team to identify possible causes, consequences, and harmed privacy requirements. Additionally, I supervised a master thesis (Jiang, 2018) that has collected different privacy threat catalogs and has shown how these can be used to support PRIOP. These theses have also shown that PRIOP is compatible with other risk analysis approaches.

Application to EHS I only discuss the last row of the template instance for R3 in Table 13.5 in detail. The row is concerned with the deduction of the treatment costs for treatments not covered by the patients' insurance contracts which is performed by doctors based on the patient details (including insurance information), the patient's health status (containing treatments and diagnoses), and the result of the accounting provided by the insurance application. The deviation that is derived from the guide word REVERSE is that the patient's personal data patient details and health status can be deduced from the treatment costs. This deviation is possible if the treatment costs are observed over a longer time, e.g., because specific diagnosed illnesses could imply a series of treatments that lead to specific treatment costs allowing to conclude from the treatment costs the diagnosed illness and received treatments. As a consequence the financial employees who are able to observe the treatment costs over a longer time might be able to partly deduce patient details and the health status of specific patients. I identified that this consequence harms the data confidentiality requirement SDP3FE and the exceptional information requirement TEP0. SDP3FE requires that all personal data of patients except their treatment costs and patient billing details shall be kept confidential from financial employees and hence, is violated if financial employees can deduce additional personal data from the treatment costs. TEP0 is concerned with informing patients about data breaches concerning their personal data and hence, may be violated if the controller is not able to detect that financial employees are able to deduce additional personal data from the treatment costs.

13.2.5. Relation of Guide Words to Privacy Requirements

To provide the analysis team with further guidance on which privacy requirements of ProPAn's privacy taxonomy (see Chapter 7) may be harmed by a guide word for a processing category, I provide a mapping between these. Based on the deviation patterns (see Tables 13.2 and 13.3), I identified the privacy requirements that are expected to be harmed by a deviation. Table 13.6 shows the relations that I identified. An "X" in the table means that a deviation implied by the guide word for the processing category, could harm the respective privacy requirement. If a cell is empty or a privacy requirement is not mentioned, then I do not expect a violation of this privacy requirement for deviations implied by the respective guide word and processing category. In Table 13.6, I use the abbreviations for the privacy requirements that are listed in Table 13.7, and I additionally introduce three groups (Gn) of privacy requirements that share the combinations of guide words and processing categories for which they are relevant.

The mapping of combinations of guide words and processing categories to privacy requirements shall help the analysis team to identify the privacy requirements that are harmed by an identified deviation, but it could also serve as a starting point to elicit situations or states of the system (consequences) that violate the privacy requirements in the context of a deviation.

I identified that for all combinations of guide words and processing categories the privacy requirements integrity (SI), availability (SA), exceptional information (TE), data subject inter-

Table 13.6.: *Privacy requirements that might be harmed by guide words' deviations*

Guide	Collection			Storage		Flow			Deduction	
Words	**G1, TC**	**G3**	**TF**	**G1, TS**	**G3**	**G1, TF**	**G3**	**TC**	**G1, G2**	**G3**
NO, LESS, PART OF, INCOR-RECT	X			X		X			X	
MORE, AS WELL AS, EARLY, LATE, BEFORE, AFTER	X	X		X	X	X	X		X	X
REVERSE	X	X	X	X		X	X	X	X	X
OTHER THAN	X	X	X	X	X	X	X		X	X

G1 = SI, SA, TE, ID, IA, G2 = TC, TF, TS, G3 = SD, UU, UA, UD, UP

Table 13.7.: *Abbreviations used for privacy requirements*

Protection Goal	Privacy Requirement	Abbreviation
Security	Data confidentiality requirement	SD
	Integrity requirement	SI
	Availability requirement	SA
Unlinkability	Undetectability requirement	UU
	Anonymity requirement	UA
	Data unlinkability requirement	UD
	Pseudonymity requirement	UP
Transparency	Collection information requirement	TC
	Storage information requirement	TS
	Flow information requirement	TF
	Exceptional information requirement	TE
Intervenability	Data subject intervention requirement	ID
	Authority intervention requirement	IA

vention (ID), and authority intervention (IA), which all belong to group G1, might be harmed. This is, because every change in the behavior of a subproblem could damage the integrity and availability of the processed data, and every change of the way that the personal data are processed by the machine could lead to exceptional cases about which the data subject has to be informed and that could violate intervention options the data subject or authorities have. Additionally, all modifications of how personal data are collected, stored, or flow can lead to a violation of the transparency requirements collection, storage, and flow information, respectively. A change in the deduction of personal data might affect collection, storage, and flow information requirements (G2).

Group G3 consists of the privacy requirements data confidentiality (SD), undetectability (UU), anonymity (UA), data unlinkability (UD), and pseudonymity (UP). These requirements might be relevant for the guide words MORE, AS WELL AS, OTHER THAN, EARLY, LATE, BEFORE, and AFTER in all processing categories, because the guide words imply either that more, additional or other data are processed by the machine, or in a different order, earlier, or later as expected, which could lead to a violation of these requirements. Note that for the guide words MORE, EARLY, LATE, BEFORE, and AFTER, the privacy requirements about the personal data that are processed are affected. In contrast, for the guide words AS WELL AS and OTHER THAN, the privacy requirements about the additional or other personal data that are processed in addition to or instead of the personal data that originally should be processed could be harmed. The guide words NO, LESS, PART OF, and INCORRECT are not implying a violation of the privacy requirements in group G3, because they only concern that fewer or incorrect personal data are processed, which does not harm the privacy requirements contained in G3. The guide word REVERSE is interpreted differently depending on the processing category (cf. Tables 13.2 and 13.3). Hence, for the categories collection, flow, and deduction it might harm the requirements in G3, but for the category *storage* it does not.

The transparency requirement flow information (TF) might be harmed by deviations for the guide words REVERSE and OTHER THAN in the processing category *collection*, because they possibly imply a flow from the machine to another domain that is not intended. Similarly, collection information requirements (TC) might be harmed by deviations characterized by the guide word REVERSE in the processing category *flow*. This is, because these scenarios would consider that instead of sending personal data to other domains, the machine would receive (collect) these or even other personal data from the target of the personal data flow.

13.3. Assess Privacy Threats Globally

Having identified the privacy threats for all subproblems, the analysis team has to assess how these privacy threats are related to each other and to obtain a consistent global view on the privacy threats of the system in the substep Assess privacy threats globally. This is, because different deviations may be caused by similar scenarios or lead to similar unwanted situations or states of the system. The homogenized *global threat model* supports the analysis team to perform a consistent risk evaluation, i.e., likelihoods and consequences of same or similar scenarios, situations, and states are consistently assigned. Additionally, the global model makes it easier to identify and integrate privacy measures that mitigate privacy risks in the step Privacy Measure Integration (see Chapter 17).

For the global view, I propose to create *threat diagrams* similar to those proposed by Lund et al. (2010). I explain these threat diagrams and how the analysis team shall create them in Section 13.3.1. If the step Privacy Measure Integration was already performed once by the analysis team, then the selected privacy measures shall be added to the threat diagrams[1] to evaluate the risk reduction implied by these. I describe this task in Section 13.3.2.

13.3.1. Create Threat Diagram

To create a global model for the locally identified privacy threats, I adopted threat diagrams from Lund et al. (2010). Figure 13.3 shows the part of ProPAn's metamodel for the creation of threat diagrams. I have taken the elements *Cause*, Causes, UnwantedIncident, ThreatScenario, and Harms from Lund et al.. The abstract class *Cause* is the super class of UnwantedIncident and ThreatScenario and has a likelihood as attribute that describes how likely it is that the cause can occur. The class Causes represents that one cause leads to another cause with a specific likelihood. A threat scenario may be performed by a collection of domains and represents itself a sequence of actions that the domains perform. An unwanted incident is a situation or state of the system

[1] Lund et al. (2010) call a threat diagram complemented by measures a treatment diagram.

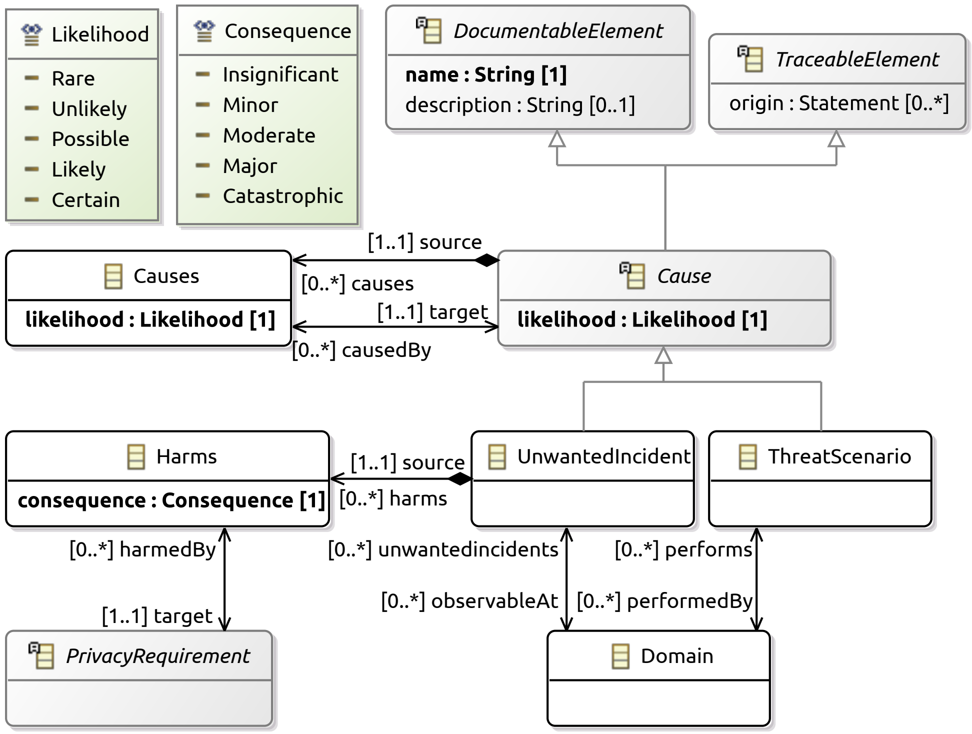


Figure 13.3.: *Part of the ProPAn metamodel dedicated to the privacy risk analysis*

that may be observable at a collection of domains, and may harm privacy requirements. The class Harms represents the latter relation between unwanted incidents and privacy requirements. The attribute consequence of the class Harms allows to specify to which degree the unwanted incident leads to non-satisfaction of the privacy requirement. Lund et al. more generally assess how unwanted incidents may harm *assets*. In my thesis, I only consider privacy requirements as assets, as these specify the privacy protection needs the system has to address. For the definition of likelihoods and consequences, I propose two enumerations that each list 5 literals. The analysis team has to specify the exact meaning of these literals using qualitative or quantitative metrics. If a quantitative metric is chosen, then, e.g., intervals are assigned to the literals. While the likelihood definition may be used for the whole system, the consequence definition depends on the privacy requirement that is harmed.

To create the global threat model, the analysis team has to translate each row of the filled out PRIOP templates to a threat diagram. A possible cause of a deviation is translated to a ThreatScenario if it describes a series of actions, and to an UnwantedIncident if it describes a situation or state of the system. The consequence of a deviation in most cases describes an UnwantedIncident. It is also possible that the consequence additionally includes a ThreatScenario. For the created threat scenarios, the analysis team shall link the domains that perform the scenario to it, and for unwanted incidents, the domains at which the unwanted situation or state is observable. Before adding a new threat scenario or unwanted incident, the analysis team shall check whether a respective threat scenario or unwanted incident is already present in the model and can be reused. In this way, the analysis team obtains a consistent model of all threats identified during the previous substep. Anyway, the functional requirement of the currently considered subproblem is added to the origin of the created or reused threat scenario or unwanted incident to document from which subproblem the respective element was identified.

As the threat scenarios and unwanted incidents created for the possible causes of the deviation cause the threat scenarios and unwanted incidents created for the consequences of the deviation, these are respectively linked using the class Causes. If a consequence of a deviation

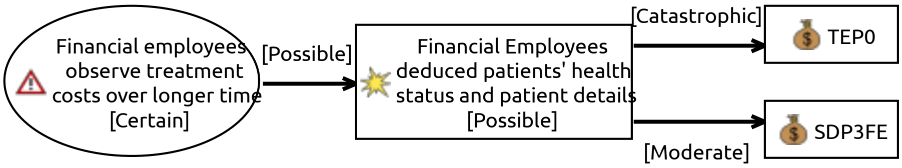

Financial employees observe treatment costs over longer time [Certain]
[Possible]
Financial Employees deduced patients' health status and patient details [Possible]
[Catastrophic]
TEP0
[Moderate]
SDP3FE

Figure 13.4.: *Threat diagram for the deviation implied by the guide word REVERSE and the processing category deduction*

contains a threat scenario in addition to an unwanted incidents, then the analysis team shall also create a Causes instance with the threat scenario as source and the unwanted incident as consequence. Finally, the analysis team has to create an instance of the class Harms for each privacy requirement that is harmed by an unwanted incident derived from the consequences of the deviation.

I use the same notation and icons as Lund et al. (2010) to present threat diagrams. Threat scenarios are presented as ellipses with a warning symbol, unwanted incidents are presented as rectangles with a star symbol, privacy requirements are presented as rectangles with a moneybag symbol (used for assets by Lund et al.), and solid arrows to present the causes and harms relations. The attributes likelihood and consequence are annotated at the elements they belong to in square brackets. Note that these values can be ignored in this substep. They are set in the substep Evaluate privacy risks (see Section 13.4).

Application to EHS Figure 13.4 shows the threat diagram for the deviation implied by the guide word REVERSE and the processing category deduction (see last line in Table 13.5). The threat scenario is that financial employees observe the treatment costs over longer time, which may cause the unwanted incident that financial employees have deduced patients' health status and patient details. This unwanted incident harms the data confidentiality requirement SDP3FE and the exceptional information requirement TEP0. Note that the likelihoods and consequences shown in the threat diagram are set in the substep Evaluate privacy risks (see Section 13.4) and not in this substep.

The other deviations shown in Tables 13.4 and 13.5 all lead to harm of the integrity requirement SIP. The threat diagram shown in Figure 13.5 shows the homogenized view on all threat scenarios and unwanted incidents leading to harm to SIP.

During the global assessment of the privacy threats also longer *causes chains* may be obtained. A causes chain is a sequence of threat scenarios and unwanted incidents which starts at a threat scenario and ends at an unwanted incident that harms a privacy requirement. A threat scenario or unwanted incident in the sequence has to *cause* its following threat scenario or unwanted incident. For example, by assessing the deviations of functional requirements R1 and R6, I identified that deviations of them have the consequence that inconsistent data are stored in the EHR. This consequence is listed as unwanted incident in Figure 13.5 and is consequently amended with threat scenarios that cause it when the deviations of R1 and R6 are translated to threat diagrams.

13.3.2. Add Privacy Measures

When the analysis team performs the step Privacy Risk Analysis again after having integrated privacy measures to operationalize privacy requirements or to reduce privacy risks in the step Privacy Measure Integration (see Chapter 17), the analysis team has to add these measures to the global threat model to document how the privacy risks are mitigated by them. The measures can lead to a reduction of the likelihood that a specific threat scenario is performed, of the likelihood that an unwanted incident occurs, or of the consequence that an unwanted incident

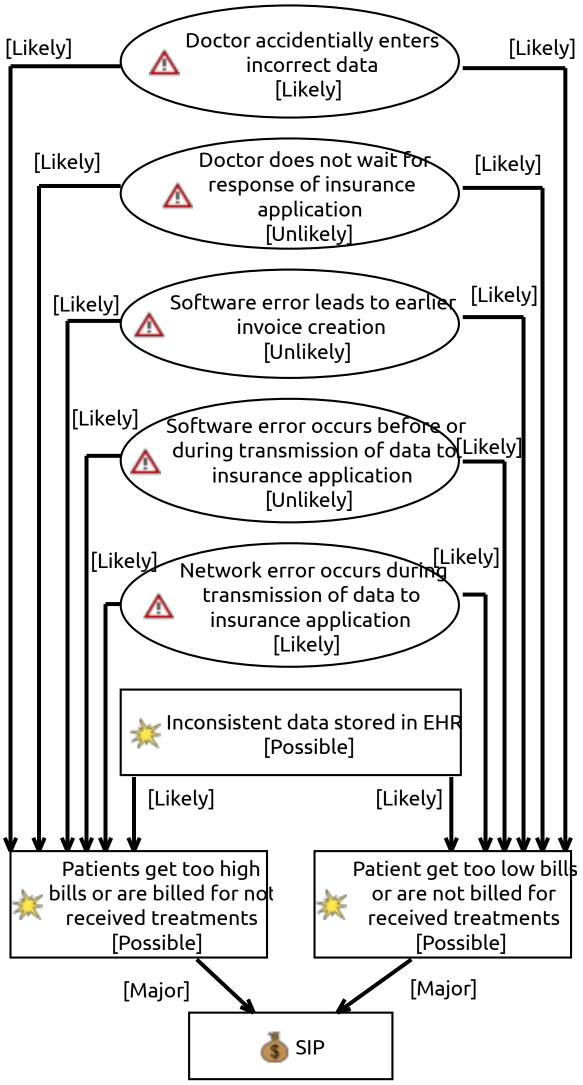

[Likely]
Doctor accidentially enters incorrect data [Likely]
[Likely]
[Likely]
Doctor does not wait for response of insurance application [Unlikely]
[Likely]
[Likely]
Software error leads to earlier invoice creation [Unlikely]
[Likely]
[Likely]
Software error occurs before or during transmission of data to insurance application [Unlikely]
[Likely]
[Likely]
Network error occurs during transmission of data to insurance application [Likely]
[Likely]
Inconsistent data stored in EHR [Possible]
[Likely]
[Likely]
Patients get too high bills or are billed for not received treatments [Possible]
Patient get too low bills or are not billed for received treatments [Possible]
[Major]
[Major]
SIP

Figure 13.5.: *Threat diagram for the deviations of R3 harming integrity requirement* SIP

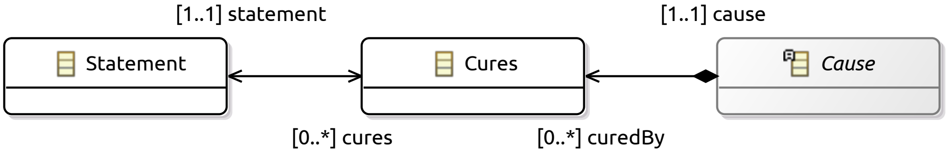


Figure 13.6.: *Representation of treatments in the ProPAn metamodel*

has on a privacy requirement it harms. These reductions are annotated in the following substep Evaluate privacy risks.

In the step Privacy Measure Integration, I propose to represent non-technical measures as facts or assumptions about the system, and technical measures as functional requirements and *aspects*, which are cross-cutting functional requirements (for details see Chapter 14). Hence, all measures are represented by statements. Figure 13.6 shows the part of the ProPAn metamodel that introduces the class Cures. This class allows to document that a measure (represented by a statement) cures a cause (threat scenario or unwanted incident), which directly or indirectly harms a privacy requirement.

For each statement that represents a privacy measure integrated into the system, the analysis

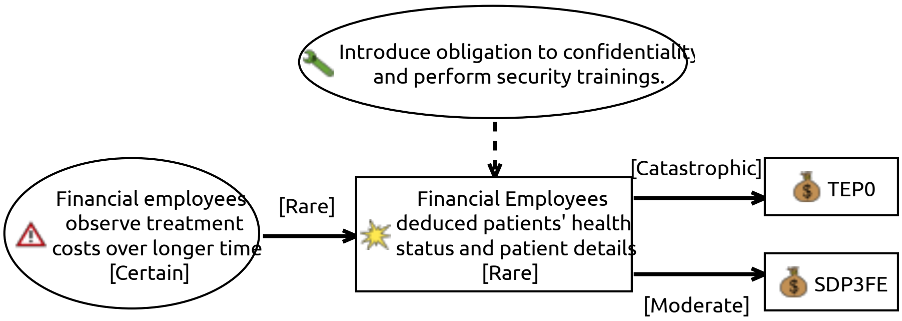

Introduce obligation to confidentiality and perform security trainings.
Financial employees observe treatment costs over longer time [Certain]
[Rare]
Financial Employees deduced patients' health status and patient details [Rare]
[Catastrophic]
TEP0
[Moderate]
SDP3FE

Figure 13.7.: *Threat diagram including a privacy measure for the deviation implied by the guide word REVERSE and the processing category deduction*

team has to create instances of the class Cures that document for which threat scenarios or unwanted incidents the statement reduces the likelihood to occur, or for which unwanted incident it reduces the consequence on a privacy requirement.

The privacy measures are presented as ellipses in a threat diagram with a tool symbol and a dashed arrow pointing to the threat scenario or unwanted incident it cures.

Application to EHS Considering the threat diagram shown in Figure 13.4, it is hardly possible or impractical to introduce technical measures to prevent that financial employees are able to observe treatment costs over a long period and deduce further information from this observation. Hence, I chose in step Privacy Measure Integration that an obligation to confidentiality shall apply to financial employees and that these shall complete security trainings. This non-technical measure is modeled as an assumption on the financial employees in the ProPAn model and cures the unwanted incidents that financial employees deduced additional information from the observed treatment costs by reducing its likelihood to occur. Figure 13.7 shows the corresponding threat diagram including the non-technical privacy measure.

13.4. Evaluate Privacy Risks

In the substep Evaluate privacy risks, the analysis team has first to estimate the likelihoods and consequences of the previously identified elements documented in the privacy threat model (see Section 13.4.1). Then the privacy risks are evaluated to decide whether the risks are acceptable or not (see Section 13.4.2).

13.4.1. Risk Estimation

In the substep Evaluate privacy risks, the analysis team has to assess the likelihoods with which the threat scenarios and unwanted incidents may occur and cause each other. Additionally, the analysis team has to determine the consequences of the unwanted incidents on the privacy requirements. For this, the analysis team should agree on a common likelihood scale for the risk analysis, be it qualitative or quantitative. A common scale will make it easier to homogeneously evaluate the risks implied by the identified privacy threats. Similar to the likelihood scale, the analysis team has also to agree on consequence scales. These consequence scales may vary for the different types of privacy requirements, because of the diverse dimensions of the software quality privacy.

The resulting threat diagrams are similar to (a collection of) attack (Schneier, 1999) and fault trees (IEC, 2006). Hence, likelihood calculations can be performed on these to calculate the likelihood that an unwanted incident that harms a privacy requirement occurs based on

Table 13.8.: *Likelihood scale for the EHS*

Likelihood	Definition
Rare	Less than once per ten years
Unlikely	Less than once per two years
Possible	Less than twice per year
Likely	Two to five times per year
Certain	Five times or more per year

Table 13.9.: *Consequence scales for the EHS*

Consequence	Security	Transparency
Insignificant	1-4 records are affected	Information is provided in a sufficient manner, but not recognized
Minor	5-20 records are affected	Information is provided, but in an insufficient manner
Moderate	21-100 records are affected	Information is available only with manual effort
Major	101-1000 records are affected	Information is only available on request
Catastrophic	1000+ records are affected	Information is not accessable and not provided on request

the cause chains that lead to them starting at the *leaf* nodes. Such likelihood calculations, as described in (IEC, 2006), may be used by the analysis team, but they are out of the scope of my thesis.

If the analysis team selected measures in the step Privacy Measure Integration and performs the step Privacy Risk Analysis again, then the analysis team has to adapt the likelihoods and consequences of the threat scenarios and unwanted incidents that are cured by the privacy measure and the likelihood calculations have to be reperformed.

Application to EHS I use the likelihood scale in Table 13.8 to define the semantics of the five likelihood literals, and the consequence scales in Table 13.9 to define the semantics of the five consequence literals for the EHS. In Table 13.9, I provide one scale for the protection goal security and one for the goal transparency. Note that the analysis team may decide to specify for each privacy requirement class a consequence scale, or even for each single privacy requirement specific to the concerned personal data and data subject.

Figures 13.4 and 13.5 already show the likelihoods and consequences that I estimate based on the defined likelihood and consequence scales. Figure 13.7 shows the privacy measure reducing the likelihood that financial employees deduce further information from the treatment costs they are able to observe over a longer time by introducing the obligation to confidentiality and security trainings for financial employees.

13.4.2. Risk Evaluation

Having assigned likelihoods and consequences to the threat model, the privacy risks to the system can be deduced. Following Lund et al. (2010), a privacy risk is given by the likelihood of an unwanted incident and its consequence to harm a privacy requirement. To determine the severity of a privacy risk, a *risk matrix* is used (Lund et al., 2010). A risk matrix specifies for each combination of likelihood and consequence its severity, i.e., whether the privacy risk is acceptable, tolerable, or unacceptable. An acceptable risk does not need to be treated, a

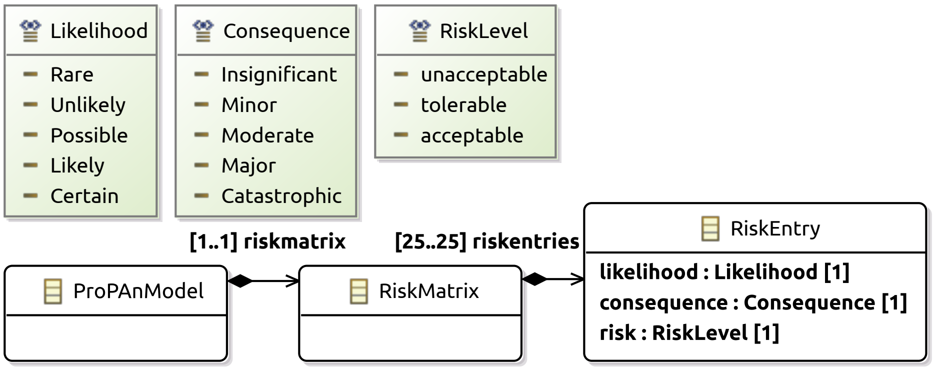


Figure 13.8.: *Metamodel for the risk matrix of a ProPAn model*

tolerable risk may be accepted if measures to prevent the risk are too expensive or impractical to integrate into the system, and unacceptable risks have to be mitigated using appropriate measures. For the risk analysis, one common risk matrix should be specified by the analysis team. Note that the analysis team should align the risk matrix with the defined likelihood and consequence scales for a consistent definition of acceptable, tolerable, and unacceptable risks. Note also that the analysis team has to specify the highest acceptable risk level for each privacy requirement in the step Privacy Measure Integration (see Chapter 17). That is, the analysis team can specify that for some privacy requirements tolerable risks can also be accepted and for others not.

Figure 13.8 shows how the risk matrix can be documented in the ProPAn model. The risk matrix consists of 25 risk entries (one for each combination of likelihood and consequence). A risk entry specifies whether its combination of likelihood and consequence represents an unacceptable, tolerable, or acceptable risk.

Using the documented risk matrix and the privacy threat model, the ProPAn tool can visualize the identified privacy risks in the risk matrix, so that the analysis team knows for which risks measures have to be selected in the step Privacy Measure Integration (see Chapter 17).

Application to EHS The ProPAn tool derives from the threat model given in Figure 13.4 the privacy risks that the shown unwanted incident harms the exceptional information requirement TEP0 (this risk is called FED_TEP0), and that the unwanted incident harms the data confidentiality requirement SDP3FE (this risk is called FED_SDP3FE). The mitigated versions of these risks (given by Figure 13.7) are called FED2_TEP0 and FED2_SDP3FE. From Figure 13.5, the ProPAn tool derives two risks. Namely, that the unwanted incidents about too high bills and too low bills harm the integrity requirement SIP. These risks are called HighBills_SIP and LowBills_SIP.

All these risks are visualized in the risk matrix specified for the EHS show in Table 13.10. The cells with white background present the acceptable risks, the cells with light-gray background the tolerable risks, and the cells with dark-gray background and white font the unacceptable risks. The risk matrix shows that all risks identified from Figures 13.4 and 13.5 are unacceptable and hence, have to be treated. The privacy risks FED2_TEP0 and FED2_SDP3FE represent the treated risks of FED_TEP0 and FED_SDP3FE. The risk matrix shows that the measure introduced in Figure 13.7 sufficiently mitigates the unacceptable risk FED_SDP3FE to the acceptable risk FED2_SDP3FE, while the risk FED2_TEP0 is still unacceptable and needs further treatment.

Table 13.10.: *Risk matrix for the EHS*

	Insignificant	Minor	Moderate	Major	Catastrophic
Rare			FED2_SDP3FE		FED2_TEP0
Unlikely					
Possible			FED_SDP3FE	HighBills_SIP, LowBills_SIP	FED_TEP0
Likely					
Certain					

13.5. Comparison to the State of the Art

In this section, I briefly discuss the state of the art privacy requirements engineering methods that I introduced in Chapter 3 and that support the identification of privacy threats.

The state of the art methods follow different approaches to identify privacy risks. The methods proposed by De and Le Métayer (2016); Knirsch et al. (2015); Deng et al. (2011); Crespo et al. (2015); Vicini et al. (2016); Oliver (2016); Yee (2017) derive privacy threats from (annotated) DFDs. Liu et al. (2003); Islam et al. (2010); Gürses et al. (2011) describe attackers against which the system shall be protected and derive threats based on the attackers' motivation and capabilities. Several methods assist the threat identification task by providing questionnaires that aim at the elicitation of privacy threats (Yee, 2017; Jensen et al., 2005; Hong et al., 2004; Mead et al., 2011). Oetzel and Spiekermann (2014); van Blarkom et al. (2003); Spiekermann and Cranor (2009) propose to take different view points on the system for an identification of privacy threats, while Oliver (2016); Knirsch et al. (2015) use ontologies to identify risks.

Breaux et al. (2015); Ahmadian and Jürjens (2016) provide tool supported methods to identify privacy threats by checking a system model for inconsistencies to privacy requirements. In the substep Validate privacy requirements of the step Privacy Requirements Identification (see Chapter 12), my validation conditions that check the consistency between the adjusted privacy requirements and the system model provide a similar functionality and can hence also be considered as a kind of privacy threat identification. Note that these privacy threats are already treated in the step Privacy Requirements Identification, by adjusting the requirements or adapting the system model.

All methods that support the evaluation of privacy risks (De and Le Métayer, 2016; Liu et al., 2003; Hong et al., 2004) use attack trees or concepts similar to them.

With PRIOP, I present a novel approach to identify of privacy threats by considering deviations of a system's subproblems using guide words. The deviations that are candidates for privacy threats are tailored to the system's functionalities to a high degree. This approach may be supported using questionnaires, attacker models, taking different viewpoints, and threat catalogs as used by the state of the art methods. Additionally, I provide support for the systematic creation of a global threat model (encoding several attack trees) based on the identified privacy threats, which is not explicitly supported by the other state of the art methods. I provide no support beyond the state of the art methods for the evaluation of privacy risks, however, I use the same or similar notations used by them. Hence, the analysis team can choose different methods for the evaluation of privacy risks.

13.6. Conclusions

The step Privacy Risk Analysis of the ProPAn method supports the analysis team to identify privacy risks from the functional requirements on the system to be by considering deviations of the functional requirement. The deviations are systematically derived by categorizing the functional requirements to processing categories and by assessing a list of guide words. The identified deviations are privacy threats if they harm privacy requirements. Based on the identified threats the analysis team creates a global threat model for the whole system. This model is used to evaluate the privacy risks implied by the identified privacy threats. Additionally, the analysis team can document privacy measures selected to mitigate identified risks and re-evaluate then the privacy risks.

My proposed privacy threat identification method PRIOP is itself a novel contribution to the state of the art and may also be adopted by or combined with other privacy analysis methods.

The privacy threat model is documented in the ProPAn model and contains traceability links allowing to link the threat scenarios and unwanted incidents to the statements they have been identified from, to the domains that perform them or at which they are observable, and to the privacy requirements that are harmed by the unwanted incidents. These traceability links support the analysis team to create a consistent model and a transparent documentation of the identified privacy risks. This is a novel contribution to the state of the art.

In future research, PRIOP may be further enhanced and evaluated. The procedure described in (IEC, 2001) to perform a HAZOP study stresses that for an analysis, the analysis team has to carefully select the guide words that are considered for the system under consideration. Similarly, it can be the case that only a subset of the proposed guide words is relevant for a PRIOP study of a specific software project or that even additional guide words are identified as important. Hence, I do not claim that my selection of guide words represents a complete set of guide words relevant for the identification of privacy threats of a software project. Similarly, the processing categories could be extended. For example, Gürses (2010) mentions that information can be collected, used, processed, distributed, or deleted. Collection and distribution (flow to other domains) are covered by my proposed categories. Usage contains from my point-of-view deduction and storage, but other kinds of usage might be identified for a concrete system as additional processing categories. Processing is considered by me as a high-level term describing that something is done with the personal data, be that collection, storage, etc. Deletion is an additional category that is worth to analyze in future work, because it is only partly covered by PRIOP. The categorization of subproblems to specific operation categories is a novelty of PRIOP and is not mentioned in the HAZOP standard. I think that making these processing categories explicit can help the analysis team to identify scenarios that harm privacy requirements. Nevertheless, it can also be valuable to consider the guide words for a given subproblem without considering the processing categories, because the processing categories could cause that the scope of the considered deviations is unnecessarily limited.

Anyway, no method for the identification of any kind of threats can guarantee to elicit a complete set of relevant threats (Young and Leveson, 2014). Nevertheless, I think that my proposed systematic analysis helps the analysis team to identify, evaluate, and document the privacy threats relevant for the system and may be complemented with techniques used by other state of the art privacy risk analysis methods (see Section 13.5).

The next step of the ProPAn method is the Privacy Measure Integration (see Chapter 17). During the Privacy Measure Integration, the analysis team has to select technical and non-technical measures to reduce the unacceptable privacy risks and to implement the privacy requirements. To implement the privacy requirements, the analysis team has to specify *success criteria* that need to be satisfied in order to satisfy the respective privacy requirement.

Most technical privacy measures have to be integrated into different functional requirements, i.e., they are *cross-cutting*. Hence, I propose to follow aspect-oriented requirements engineering, which aims at describing cross-cutting requirements (called *aspects*) mostly independently from the functional requirements they shall be integrated into it. I introduce aspect-oriented requirements engineering in Chapter 14, and patterns for aspects, called *aspect frames* in Chapter 15.

In Chapter 16, I then introduce how privacy enhancing technologies (PETs) can modularly be represented as aspects and documented together with their properties in a pattern format. These documented patterns are used as input for the step Privacy Measure Integration.

Part IV.

Integration of Privacy Solutions into the Problem

Chapter 14

Aspect-oriented Requirements Engineering with Problem Frames

In this chapter, I introduce how concepts of aspect-oriented requirements engineering can be integrated into the problem frames approach. The central idea of aspect-orientation is that functionalities exist that are or need to be integrated into multiple other functionalities. These cross-cutting functionalities are called *aspects* and shall be described mostly independently of the functionalities they shall be integrated into. This integration is called *weaving*.

This chapter is based on (Faßbender et al., 2014) and its extended version (Faßbender et al., 2015). These papers describe a method, called AORE4PF, for aspect-oriented requirements engineering supporting the consideration of quality requirements based on the problem frames approach. Stephan Faßbender and Maritta Heisel are my co-authors in both papers. Stephan Faßbender contributed the method steps to classify requirements into cross-cutting and not cross-cutting functionalities, and to identify the software qualities the cross-cutting requirements target. Additionally, he performed the evaluation of the AORE4PF method. Maritta Heisel provided substantial feedback on the papers that helped to improve them. In this chapter, I focus on how aspects can be modeled and integrated (weaved) into the requirements they cross-cut. The other steps of the AORE4PF method are out of the scope of my thesis.

I provide an introduction to aspect-oriented requirements engineering in Section 14.1. In Section 14.2, I describe how UML sequence diagrams can be used to provide a behavioral view on statement diagrams (including problem and domain knowledge diagrams; cf. Section 2.3). This behavioral view specifies how the domains in the system interact with each other to satisfy the system's requirements. The modeling of aspects is introduced in Section 14.3 and their integration into the requirements they cross-cut in Section 14.4. I provide an overview of other aspect-oriented requirements engineering methods in Section 14.5. Section 14.6 concludes this chapter.

14.1. Introduction

Keeping an eye on good and sufficient requirements engineering is a long-known success factor for software projects and the resulting software products (Hofmann and Lehner, 2001). Nowadays, for almost every software system, various stakeholders with diverse interests exist. These interests give rise to different sets of requirements. These diverse requirements not only increase the complexity of the system-to-be, but also contain different cross-cutting concerns, such as qualities, which are desired by the stakeholders. In such a situation, the requirements engineer is really challenged to master the complexity and to deliver a coherent and complete description of the system-to-be.

One possible option to handle the complexity of a system-to-be is the concept of *separation of concerns* (Parnas, 1972). In its most general form, the separation of concerns principle refers to the ability to focus on, and analyze or change only those parts of a system which are relevant for one specific problem. The main benefits of this principle are a reduced complexity, improved comprehensibility, and improved reusability (Parnas, 1972).

Both, *aspect-oriented requirements engineering (AORE)* (Rashid, 2008) and the problem frames approach implement this principle, but in different ways. The approach of AORE, which originates from aspect-oriented programming, is to separate each cross-cutting requirement into an *aspect*. Instead of integrating and solving the cross-cutting requirement for all requirements it cross-cuts, the aspect is solved in isolation. Hence, aspect-orientation leads to a clear separation of concerns. To combine an aspect with a requirement, an aspect defines a pointcut (set of join points), which describes how the aspect and the subproblem of a requirement can be combined. The actual integration of the aspects into the requirements is called *weaving*. The *problem frames approach* (Jackson, 2000) generally also follows the separation of concerns principle (see also Section 2.2). It decomposes the overall problem of building the system-to-be into small subproblems that fit to a problem frame. Each subproblem is solved by a machine, which has to be specified using the given domain knowledge. All machines have to be composed to form the overall machine.

Faßbender et al. (2014, 2015) have shown that aspect-orientation gives guidance for the process of decomposing the overall problem and for the composition of the machines, especially, when software qualities have to be considered. This is, because mechanisms addressing software qualities need often to be integrated into multiple other functionalities.

Both ways of separating concerns seem to be complementary, and hence, it is promising to combine them. In the following, I describe how to create behavioral views for statement diagrams, and how aspects can be modeled and weaved into the functional requirements based on the problem frames approach.

14.2. Modeling Behavior

So far, I only considered problem diagrams for the description of the system's functional requirements. Problem diagrams only provide a structural view on a subproblem and lack information about how the domains interact with each other to satisfy the functional requirements. For the integration of privacy measures, a behavioral view is necessary. This is, because the success of the privacy measures strongly depends on how and when they are integrated into the behavior of the system.

I propose to specify how the domains interact with each other using UML sequence diagrams (Bock et al., 2015). For each statement diagram, a sequence diagram may be created that visualizes the interaction between the domains contained in it using the phenomena annotated at the interfaces between the domains. The occurrence of a causal phenomenon or an event is translated to an asynchronous message from the controller to the observer of the causal phenomenon or event. The observation of a symbolic phenomenon is either modeled as described before, or as a synchronous message from the observer to the controller using the prefix *get_* in the name of the message, and a reply message from the controller to the observer. The analysis team may use all features of sequence diagrams to model the interaction between the domains, e.g., interaction uses referencing other sequence diagrams, combined fragments, and state invariants (Bock et al., 2015).

Figure 14.1 shows the metamodel that allows to connect UML sequence diagrams to problem frame models. Each sequence diagram (represented by the type UML::Interaction) is mapped to a Problemframes::StatementDiagram by an InteractionMapping. That is, the sequence diagram shows the interaction between the domains described by the statements of the respective statement diagram. The UML::Lifeline and UML:Message objects of such a sequence diagram represent the Problemframes::Domain and Problemframes::Phenomenon objects of the problem

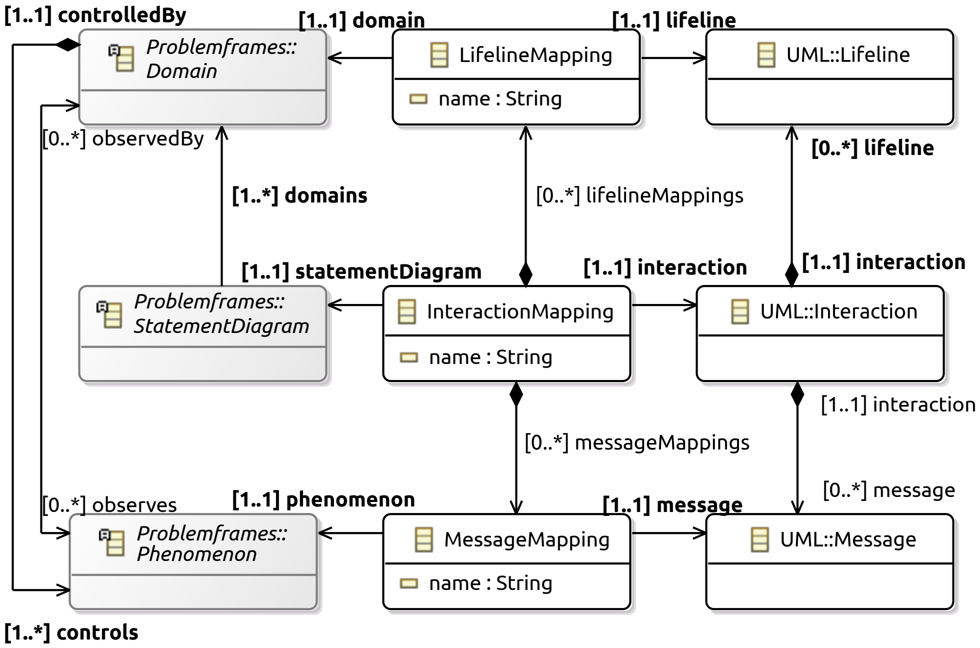


Figure 14.1.: *Metamodel for the mapping of UML sequence diagrams to problem frame models*

Listing 14.1: *OCL invariants describing consistency conditions for the class* InteractionMapping *and its elements*

```
context InteractionMapping
inv: self.lifelineMappings→forAll(lm|
   self.interaction.lifeline→includes(lm.lifeline) and
   self.statementDiagram.domains.includes(lm.domain))
inv: self.messageMappings→forAll(mm|
   self.interaction.message→includes(mm.message) and
   self.statementDiagram.domains.controls.includes(mm.phenomenon))
```

frame model, respectively. These mappings are presented using the classes LifelineMapping and MessageMapping. Listing 14.1 provides two invariants that specify that the lifeline and message mappings of an interaction mapping shall only map elements of the statement diagram and interaction linked to the interaction mapping. Further validation conditions may be set up, e.g., constraining that phenomena shall only be annotated at messages between lifelines for which lifeline mappings exist that map these to domains that share the annotated phenomena, but are out of the scope of my thesis.

Application to EHS Requirement R1 is concerned with the management of EHRs performed by doctors (see Figure 4.2 on page 67). To address the requirement, the machine EHS has to react on the commands createEHR and modifyEHR from the Doctor, and issue the commands newEHR and changeEHR observed by the EHR. This is specified in the sequence diagram shown in Figure 14.2.

The sequence diagram for the problem diagram for functional requirement R3 (see Figure 4.4 on page 68) is shown in Figure 14.3. It shows that the Doctor initiates the accounting. In reaction, the EHS queries the health records from the EHR to retrieve unaccounted treatments. These are then sent together with the patient's insurance number and diagnoses (also retrieved from the health records) to the Insurance Application using the causal phenomenon accounting. Based on the provided data, the Insurance Application returns the accountingResult to the EHS.

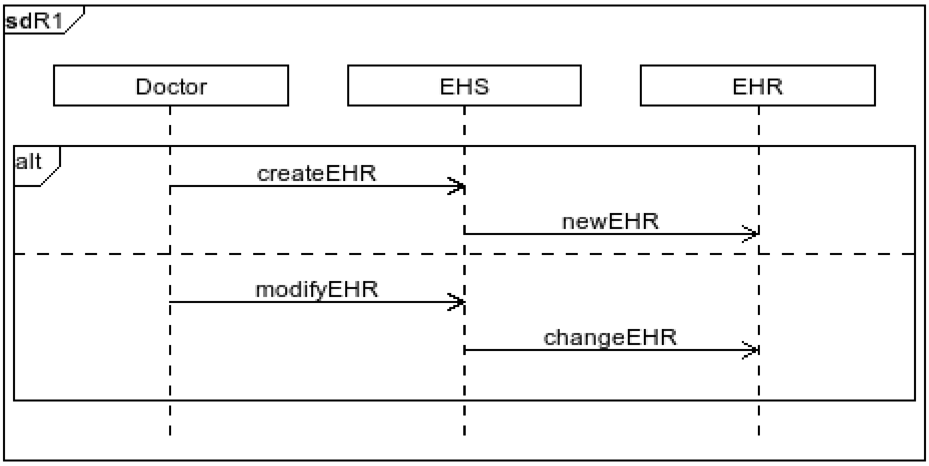


Figure 14.2.: *Sequence diagram describing the behavioral view on the problem diagram for R1*

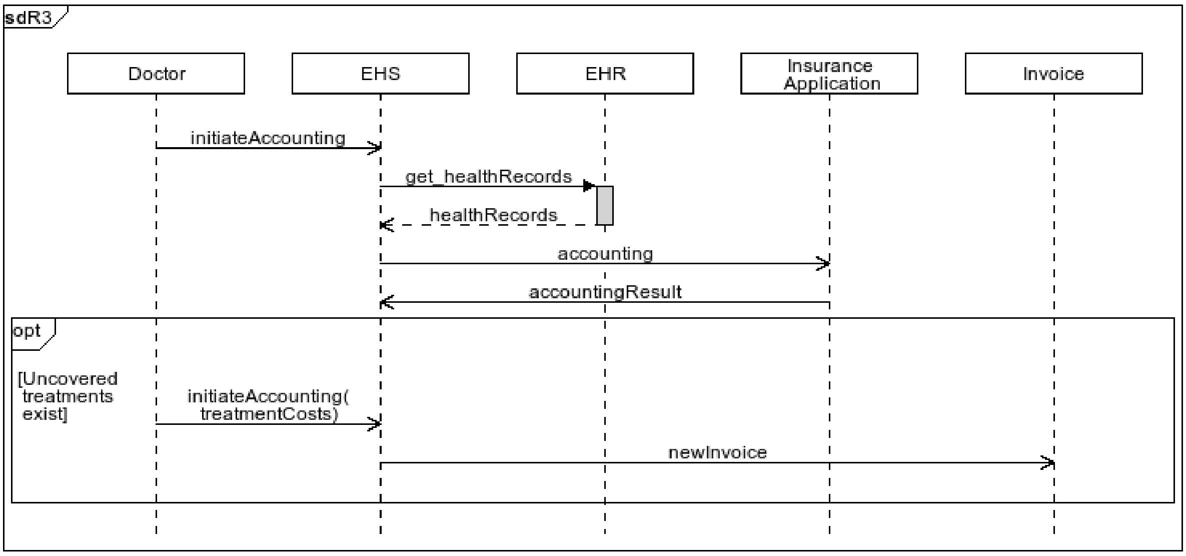


Figure 14.3.: *Sequence diagram describing the behavioral view on the problem diagram for R3*

If this result states that specific treatments are not covered by the patient's insurance contract, then the Doctor has to specify the treatment costs and the EHS creates a new invoice managed by the domain Invoice.

14.3. Modeling Aspects

Aspect-oriented requirements engineering (Rashid, 2008) introduces the following new concepts to requirements engineering.

Aspect An aspect is a functional requirement that cross-cuts other functional requirements. That is, an aspect describes functionalities that need to be integrated into other functionalities. A common example for an aspects is the logging of performed actions. As the actions to be logged may be spread over different functionalities, the logging aspect cross-cuts multiple functionalities. Instead of implementing the cross-cutting functionality for each functionality separately, the idea of aspect-orientation is to provide one implementa-

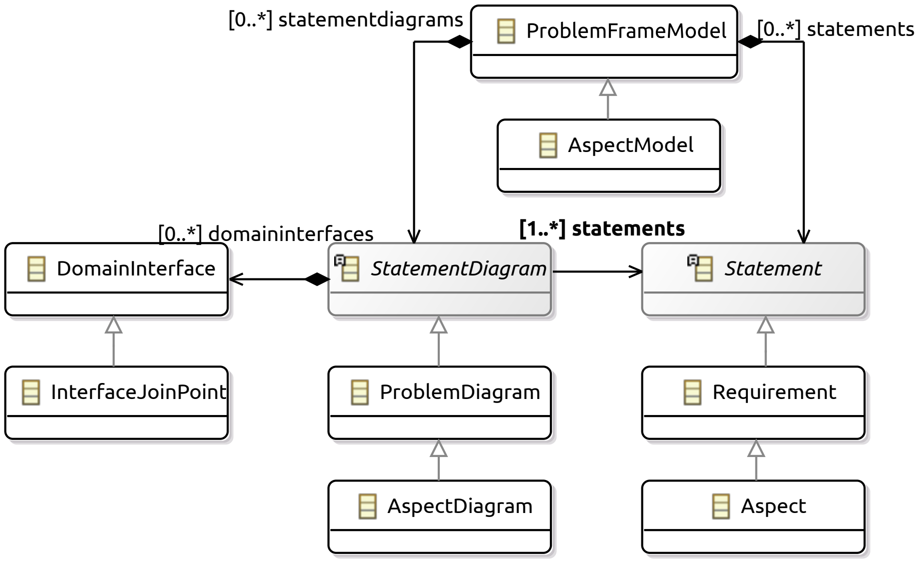


Figure 14.4.: *Extension of the UDEPF metamodel with aspect-oriented elements*

tion of the cross-cutting functionality and to integrate it then into the other functionalities. This integration is called weaving.

Join point A join point is a placeholder representing an element of the part of the system realizing the functionality into which the aspect shall be integrated into. That is, the join points define into which problems the aspect might be integrated into. To integrate an aspect into a problem, the join points have to be instantiated with the domains of the problem.

Weaving The integration of aspects into problems is called weaving. To weave an aspect into a problem, the join points of the aspect have to be instantiated and the behavior to implement the aspect has to be integrated into the behavior to solve the subproblem (see Section 14.4).

To embed aspect-orientation into the problem frames approach, I extended the UDEPF metamodel (see Section 2.3) with aspect-oriented concepts (see Figure 14.4). To describe aspects separately from the other functional requirements, I introduce the class AspectModel as a subclass of ProblemFrameModel to specify a collection of related aspects in an independent model. An Aspect is a special functional Requirement. The part of the system that is responsible for addressing the aspect is visualized in an AspectDiagram, which is a special ProblemDiagram. The part of the system that is responsible for addressing the aspect contains join points that are later instantiated with elements of the subproblem into which the aspect shall be integrated into. Domains, phenomena, and the interfaces between domains can be join points in an aspect diagram. Hence, I introduce the type InterfaceJoinPoint as a subclass of DomainInterface to represent that an interface between two domains is a join point.

Figure 14.5 shows the additional classes that I introduce to represent domain join points. For each domain type described by Jackson (2000), I created a class representing a respective join point that can only be instantiated by domains of the respective type. Additionally, I provide a general DomainJoinPoint that may be instantiated with any domain type. This is especially useful in cases, where an aspect can be integrated into problems with different domain types. The class AspectMachine represents the part of the software that is responsible to implement the aspect.

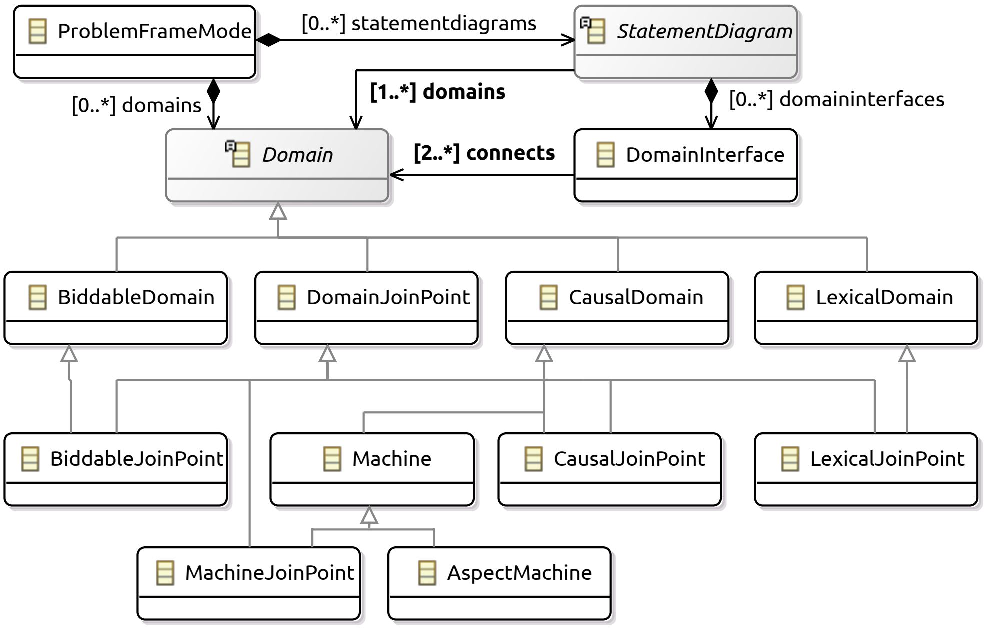


Figure 14.5.: *Extension of the UDEPF metamodel with domain join points*

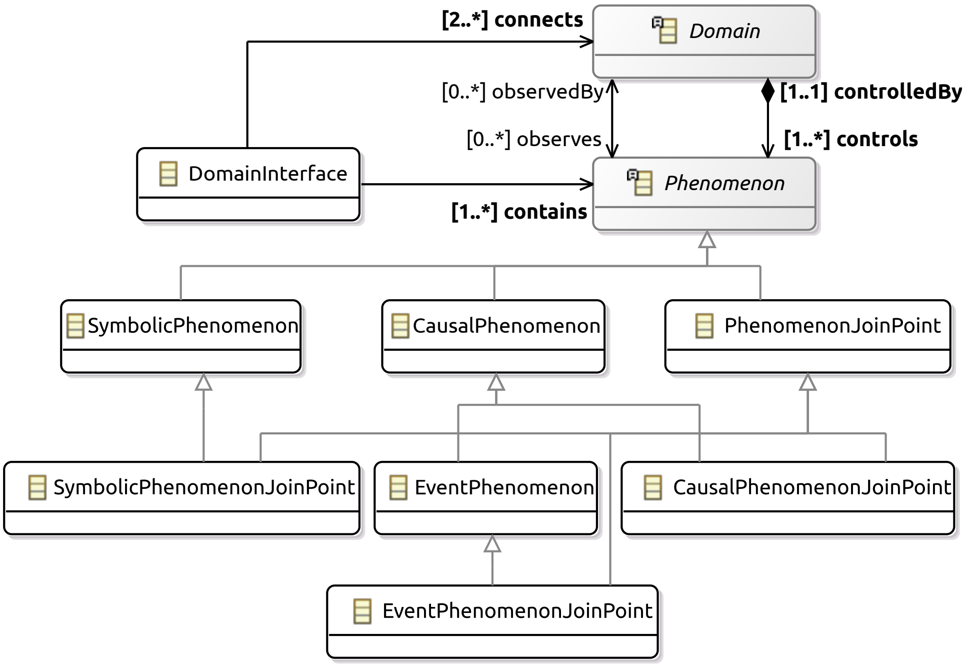


Figure 14.6.: *Extension of the UDEPF metamodel with phenomenon join points*

Similar to the domain join points, I introduce the phenomenon join points shown in Figure 14.6. For each phenomenon type, I created a respective join point that can only be instantiated with the respective phenomenon type, and a general PhenomenonJoinPoint that may be instantiated with any phenomenon type.

To create an aspect diagram, the requirements engineer has to identify the join points which are necessary to combine the aspect with the problems it cross-cuts, and to understand the problem of building the aspect machine. In most cases, an aspect diagram contains a machine, besides the aspect machine, as join point. This machine will be instantiated during the weaving with the machine of the problem that the aspect is weaved into. The interface between this join

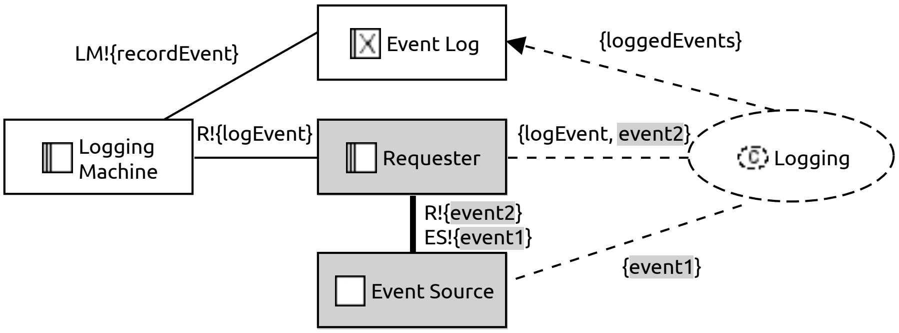


Figure 14.7.: *Aspect diagram for the aspect Logging*

point and the aspect machine describes how a problem machine can utilize an aspect and which context information is needed by the aspect machine. The requirements engineer has to derive the join points important for the problem described by the aspect from its description and the requirements it may cross-cut. Besides the specialized types for the machine (AspectMachine) and the requirement (Aspect), and the definition of join points for the later weaving, the process of creating an aspect diagram is similar to the process of creating problem diagrams.

As for problem diagrams, I also propose to create sequence diagrams for each aspect that describe the behavior of the system to address the aspect (see Section 14.2). The sequence diagrams contain two kinds of information. First, the messages mapped to phenomena that are join points describe the pointcut scenario. That is, these messages describe when during the behavior necessary to accomplish the cross-cut requirement the behavior of the aspect shall be integrated into the behavior for the cross-cut requirement. Second, all other messages describe the internal behavior necessary to accomplish the aspect.

Join points are presented using gray background in aspect diagrams and also in the corresponding sequence diagrams. The symbol of aspects contains the letter "C" for cross-cutting requirement, because the letter "A" is already used for assumptions in the UDEPF tool.

As mentioned before, logging is a commonly considered aspect. Figure 14.7 shows the aspect diagram for this aspect. It contains the aspect machine Logging Machine, which is able to record events in the Event Log. Furthermore, the aspect diagram contains two domains as join points. The machine join point Requester will be instantiated by a problem machine and the domain join point Event Source by the origin or target of the event to be logged. The machine Requester observes the phenomenon join point event1 of Event Source and is able to issue the phenomenon join point event2. These phenomena represent the events that shall be logged and need to be instantiated during the weaving. Note that it is possible to integrate the aspect also in problems where only one type of these events shall be logged. If an event that has to be logged is observed, then the machine join point Requester instructs the aspect machine Logging Machine to record that event.

I distinguish four cases of events to be logged. An event could be issued by the Event Source (first case) and optionally be answered by the Requester (second case), or the machine Requester issues the event (third case), which is optionally answered by the Event Source (fourth case). These four cases are encoded in the sequence diagram for the logging aspect shown in Figure 14.8. As a result of the observation of event1 and/or event2, the machine join point Requester asks the aspect machine Logging Machine to record the observed event. The Logging Machine then records the event in the Event Log.

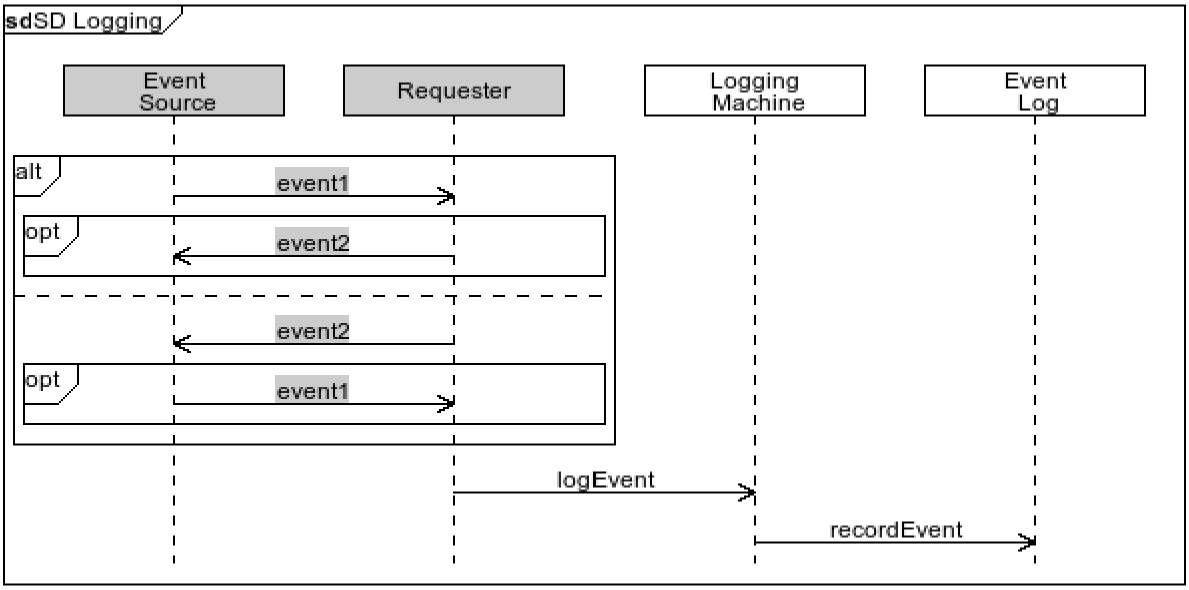


Figure 14.8.: *Sequence diagram describing the behavioral view on the logging aspect*

14.4. Weaving Aspects into Requirements

To weave an aspect into a subproblem, the join points of the aspect have to be instantiated with respective elements of the subproblem. For this, I propose the metamodel shown in Figure 14.9. A WeavingModel contains a set of WeavingDiagram instances. Each WeavingDiagram is a specialization of a ProblemDiagram. A weaving diagram references exactly one ProblemDiagram for which the weaving diagram describes the integration of the referenced AspectDiagrams into it. The integration is specified by a collection of *JoinPointInstantiation* instances. As domains, phenomena, and interfaces between domains can be join points, I propose three join point instantiation (JPI) classes, namely, DomainJPI, PhenomenonJPI, and InterfaceJPI. Every JPI specifies for a join point of an aspect diagram with which respective element of the problem diagram it shall be instantiated.

Having these join point instantiations, a graphical representation of the weaving diagram can be derived that shows all domains of the referenced problem and aspect diagrams, while the join points are instantiated with the elements specified by the respective join point instantiations. Similar to problem diagrams, I also propose to create a behavioral view on weaving diagrams using UML sequence diagrams (cf. Section 14.2). The behavioral view on a weaving diagram describes for each referenced aspect diagram how and when the aspect's behavior is integrated into the subproblem's behavior. The aspect's behavior is integrated into the subproblem's behavior if the pointcut scenario, given by the messages representing phenomenon join points can be matched with corresponding messages in the subproblem's behavior that represent phenomena that instantiate the respective phenomenon join points. For example, the pointcut scenarios described by the logging aspect (see Figure 14.8) are that either event1 optionally followed by event2 occurs, or event2 optionally followed by event1. When this aspect is integrated into a subproblem, then its behavior is integrated after the occurrence of a phenomenon instantiating event1 optionally followed by a phenomenon instantiating event2, or after the occurrence of a phenomenon instantiating event2 optionally followed by a phenomenon instantiating event1.

Application to EHS Table 14.1 shows the join point instantiations of the weaving diagram shown in Figure 14.10. This weaving diagram specifies how the logging aspect shall be integrated into the subproblem to address R1 of the EHS (see Figure 4.2 on page 67). The logging aspect

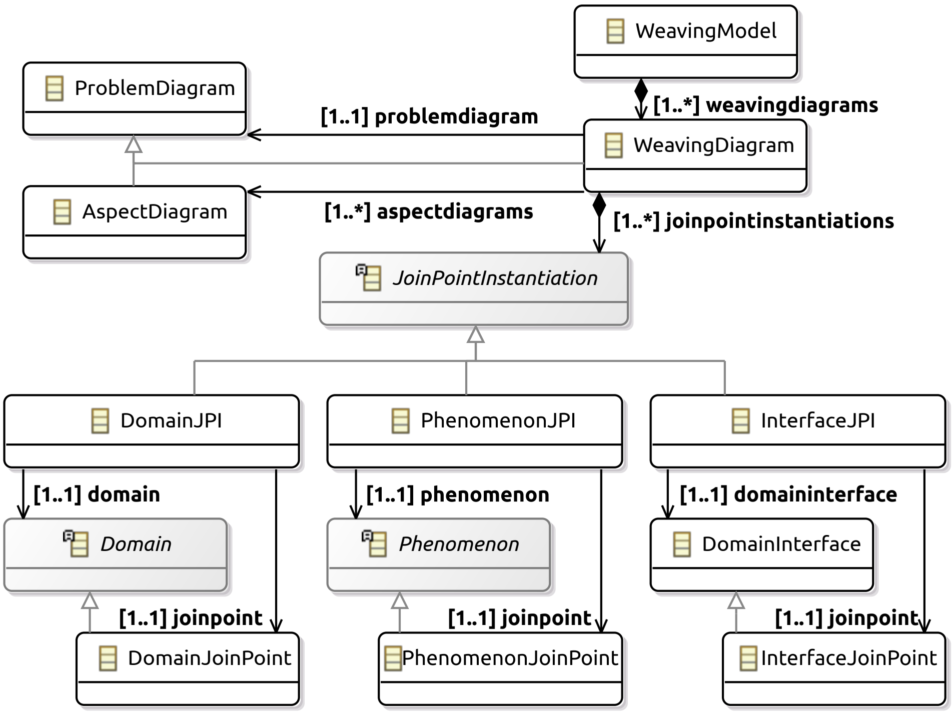


Figure 14.9.: *Extension of the UDEPF metamodel weaving models*

Table 14.1.: *Join point instantiations for the integration of the logging aspect into R1 of the EHS*

Join Point	Instantiation for createEHR	Instantiation for modifyEHR
Event Source	Doctor	Doctor
Requester	EHS	EHS
event1	createEHR	modifyEHR
event2	-	-
R!{event2} ES!{event1}	D!{createEHR}	D!{modifyEHR}

shall be integrated into the subproblem of R1 to log the modifications to the health records made by the doctors. Hence, the doctors are the Event Source, and their caused phenomena createEHR and modifyEHR are the events that shall be logged. These phenomena instantiate the phenomenon join point event1 as they are controlled by the event source. The interface join point between the Requester and the Event Source is instantiated with the respective domain interface between the Doctor and the EHS. The phenomenon join point event2 is not instantiated. The machine join point Requester is instantiated with the machine of the subproblem EHS (cf. Table 14.1).

The behavioral view on the weaving diagram is shown in Figure 14.11. This sequence diagram is obtained by integrating the sequence diagram for the logging aspect (see Figure 14.7) into the sequence diagram for R1 (see Figure 14.2). This is done twice, because the pointcut scenario of the logging aspect (occurrence of event1) occurs twice in the sequence diagram for R1 (occurrence of createEHR and modifyEHR; cf. Table 14.1). Note that the requirements engineer may decide to modify the obtained weaving sequence diagram. For example, he or she could decide that the logging shall take place after the messages newEHR and changeEHR, or in parallel to them.

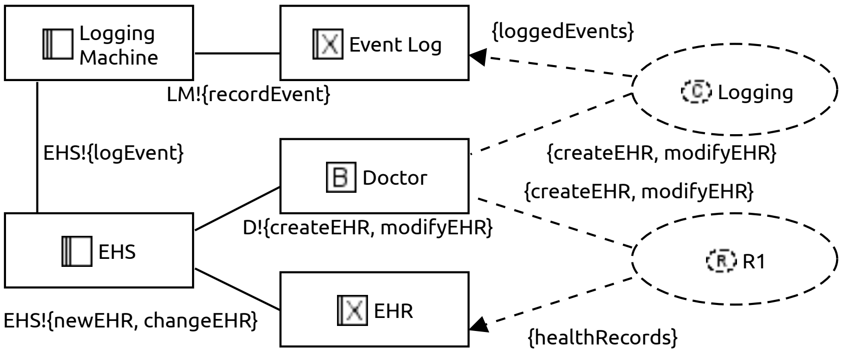


Figure 14.10.: *Weaving diagram showing how the logging aspect is integrated into R1 of the EHS*

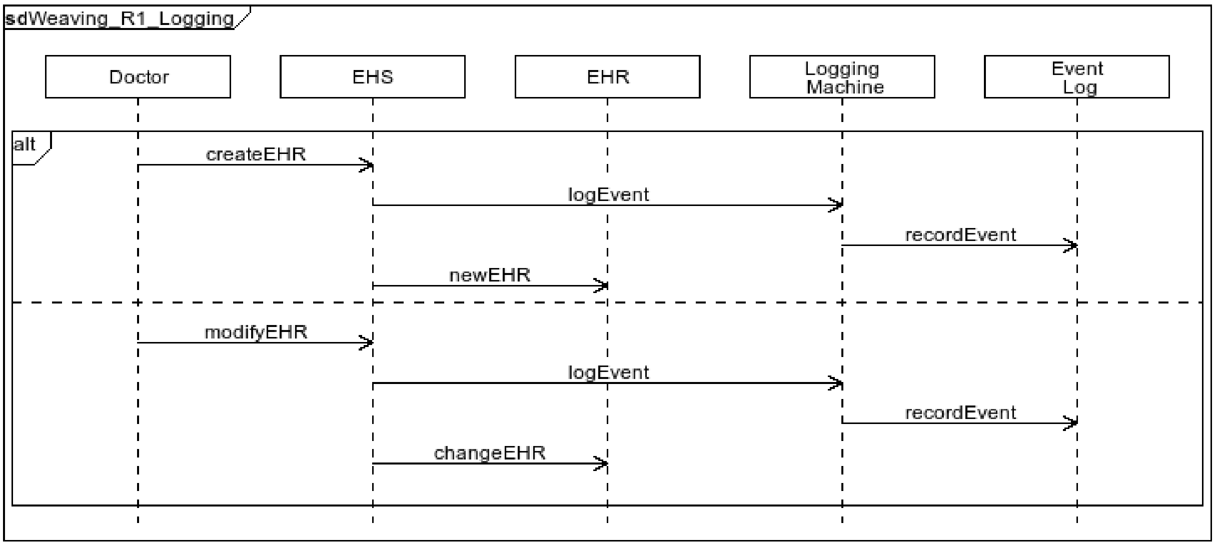


Figure 14.11.: *Sequence diagram describing the behavioral view on the integration of the logging aspect into R1*

14.5. Related Work

There are many works considering aspect-oriented requirements engineering (Rashid, 2008; Yu et al., 2004; Jacobson and Ng, 2004; Whittle and Araújo, 2004; Sutton and Rouvellou, 2002; Moreira et al., 2005; Grundy, 1999). Most of these methods deal with goal-oriented approaches and use-case models. Goal and use-case models are of a higher level of abstraction than problem frames. Additionally, goal and use-case models are stakeholder-centric, while problem frames are system-centric. Therefore, refining functional requirements taking into account more detail of the system-to-be and analyzing the system-to-be described by the functional requirements is reported to be difficult for such methods (Alrajeh et al., 2009). Furthermore, Mohammadi et al. (2013) and Beckers et al. (2013b) report a successful integration of goal- and problem-oriented methods. Hence, the goal-oriented methods might be combined with AORE4PF to combine the stakeholder- and system-centric views of both approaches.

Only few methods consider the integration of early aspects in the problem frames approach. Lencastre et al. (2006, 2008) investigated how early aspects can be integrated into problem frames. Their method to model aspects in the problem frames approach differs from mine. For an aspect, the authors first select a problem frame as *PF Pointcut Scenario*. This pointcut scenario defines into which problems the aspect can be integrated. The pointcut scenario is then extended to the *PF Aspectual Scenario*, which is similar to my aspect diagrams, with the difference that the pointcut always has to be a problem frame. This reduces flexibility, because an aspect (e.g., logging of all system events) may have to be integrated into different problem diagrams fitting to different problem frames.

14.6. Conclusion

In this chapter, I presented parts of the AORE4PF method, which integrates aspect-oriented concepts into the problem frames approach. The presented parts are the modeling of a behavioral view for statement diagrams, the modeling of cross-cutting functional requirements as aspects, and the weaving of aspects into subproblems. I extended the UDEPF metamodel with classes and relations that allow requirements engineers to map sequence diagrams as behavioral views to statement diagrams, to model cross-cutting functional requirements in aspect diagrams, and to weave aspects into subproblems using weaving diagrams. I further explained how to model aspects, and to weave them into the problems they cross-cut.

Note that the resulting requirements model not necessarily leads to an aspect-oriented implementation of the software. The identified aspects can also help to define the structure of a component-based implementation. That is, an aspect may be implemented by a single component that is used by the components realizing subproblems into which the aspect shall be integrated into.

As future work, the tool support for AORE4PF can be further improved. For example, the instantiation of pointcut scenarios during the weaving can be automated to a high degree. Additionally, the derivation of a software architecture based on an aspect-oriented requirements model may be researched. This derivation may be based on the work of Choppy et al. (2011) and Alebrahim et al. (2012a) that already researched the deduction of software architectures from problem frames.

In the next chapter, I introduce classes of aspects that share a common concern, called aspect frames. These aspect frames provide support to model aspect diagrams, similar to Jackson's problem frames.

Chapter 15

Aspect Frames

In this chapter, I introduce *aspect frames* as classes of cross-cutting concerns similar to Jackson's problem frames (see Section 2.2). I identified these frames during the modeling of different aspects using the AORE4PF method (see Chapter 14). I describe aspect frames and their properties as patterns in a structured format.

This chapter is based on (Meis and Heisel, 2017a) of which I am the main author. My co-author Maritta Heisel provided substantial feedback that helped to improve the paper.

I provide an introduction to this chapter in Section 15.1. In Section 15.2, I describe the pattern format that I use to present aspect frames. In Section 15.3, I introduce the four aspect frames that I observed. I provide guidance for the creation and usage of aspect frames in Section 15.4. I briefly discuss other research concerning aspect-orientation and patterns in Section 15.5, and Section 15.6 concludes this chapter.

15.1. Introduction

I introduce aspect frames as a means to support the description of cross-cutting concerns in aspect-oriented requirements engineering (AORE) in this chapter. An aspect frame describes a class of aspects that share a common concern and behavior. For example, encryption, anonymization, filtering, compression, and correction mechanisms have in common that these often need to be applied on data before these are transmitted to a specific receiver. From these examples, I derived the *Transform before transmission aspect* frame (introduced in Section 15.3.3) that represents the class of all aspects that perform a transformation on data before these can be transmitted to a specific receiver. I observed the aspect frames during the application of the AORE4PF method (see Chapter 14) and my research on the presentation of privacy enhancing technologies (PETs) as aspects (see Chapter 16).

I propose a pattern format to uniformly represent aspect frames as patterns. The idea of aspect frames is similar to Jackson's problem frames (Jackson, 2000). Namely, the definition of a class of software development problems that share a common structure and behavior. This is the reason why I use my extension of the problem frames approach for aspect-orientation (see Chapter 14) to describe aspect frames. Note that I have observed the aspect frames only in my own research and not yet in other work. Hence, I do not claim that aspect frames are patterns in the understanding of the pattern community (cf. Harrison (2003); Wellhausen and Fießer (2011)). They are rather pattern candidates.

15.2. Pattern Format for Aspect Frames

In this section, I describe the pattern format that I use to describe aspect frames. I base my pattern format on the suggestions of Wellhausen and Fießer (2011). For the pattern sections **Context** and **Solution**, I introduce subsections that are tailored to present aspect frames. In the following, I show and discuss the sections of the pattern format. For the sake of simplicity, I refer to instances of aspect frames as aspects.

Name The name of the aspect frame

Context The context of an aspect frame describes the problems into which it shall be integrated, called *base problems*. I propose to present on the one hand a structural view on the base problem and the relevant behavior of it for the aspect, i.e., the aspect's pointcut scenario.

Structure The structure of a base problem is described by a (possibly incomplete) problem frame. This problem frame has to contain the core domains of a base problem into which instances of the aspect frame might be integrated into. I permit that the domain types in the context description are left open. That means that the context is not limited to base problems with domains of a specific type. Additionally, it is allowed that a base problem contains more domains than described by the provided kind of problem frame. This is, the provided problem frame does not need to be complete.

Behavior The relevant behavior of the base problem is described in a sequence diagram. This sequence diagram describes in most cases the behavior that needs or leads to the integration of an aspect.

Problem A short phrase that describes the base problem of the requirement that is addressed by aspects fitting to the aspect frame.

Forces A list of possibly conflicting properties that make it hard to address the aforementioned base problem.

Solution The solution contains the actual aspect frame in the sense of Jackson's problem frames. Similar to the context, I divide the description of the solution into subsections that explain its structure, behavior, and how it is integrated into the base problem.

Structure I describe the basic structure of the aspect frame as a high-level aspect diagram that contains as join points the parts of the base problem that are relevant for the functionality of the aspect and the domains that the aspect newly introduces to address the base problem. I allow that domain types are left out in the aspect frames. This leads in most cases to the definition of variants of the aspect frame (see **Variants**).

Behavior The behavior part shows when and how the machine of the base problem will integrate the aspect and the aspect's behavior to achieve the needed solution of the base problem using a sequence diagram.

Integration This part describes how the aspect's behavior shall be integrated into the base problem, by combining the sequence diagrams of the base problem and the aspect frame.

Variants This section is optional. In some cases, domains of the aspect frame can be instantiated with different types while the underlying concern of the aspect frame remains the same. In these cases, I permit to describe variants that assign specific domain types to the general domains of the aspect frame. These variants can then be referenced in the consequences to be more precise on the benefits and liabilities of the different variants.

Formally, the variants are candidates for separate aspect frames, but I decided to allow the descriptions of these in one pattern if they share the same context, problem, and solution.

Consequences The positive and negative impact of the integration of the aspect frame into the base problem shall be discussed in the consequences section. The consequences shall be discussed based on the previously defined forces for all described variants of the aspect frame.

Benefits The positive consequences of instances of the aspect frame to the **Forces**.

Liabilities The negative consequences of instances of the aspect frame to the **Forces**.

Examples Known aspects that are instances of this aspect frame. If variants are defined, then examples for all variants shall be given.

15.3. Aspect Frames

In this section, I present the four aspect frames *Decision aspect* (see Section 15.3.1), *Transform received data aspect* (see Section 15.3.2), *Transform before transmission aspect* (see Section 15.3.3), and *Independent behavior aspect* (see Section 15.3.4) using the pattern format introduced in the previous section.

15.3.1. Decision Aspect Frame

Name Decision aspect

Context

Structure You have base problems whose behavior shall depend on specific properties of the received request. The base problem's essential structure is shown in Figure 15.1. It consists of a Base Machine that is concerned with handling requests of a domain Requester concerning a Resource that the Base Machine manages.

Behavior Figure 15.2 shows the relevant behavioral view on the base problem. It consists of a Before behavior, the request of the Requester to the Base Machine, and an After behavior.

Problem You want to integrate a mechanism that decides based on the Requester's request whether the after behavior shall occur or not.

Forces

- *Complexity of the implementation:* The complexity to implement the mechanism should be kept to the minimum needed.
- *Integration into the base problem:* It shall be possible to integrate the decision mechanism in a way that the base problem is only affected in the desired manner.
- *Response time of decision:* The selected mechanism shall respond in a timely manner depending on the needs of the base problem.

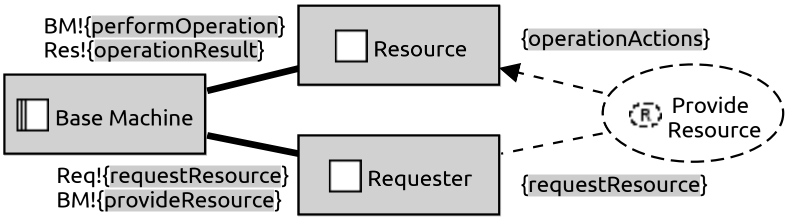


Figure 15.1.: *The base problem of Decision aspects*

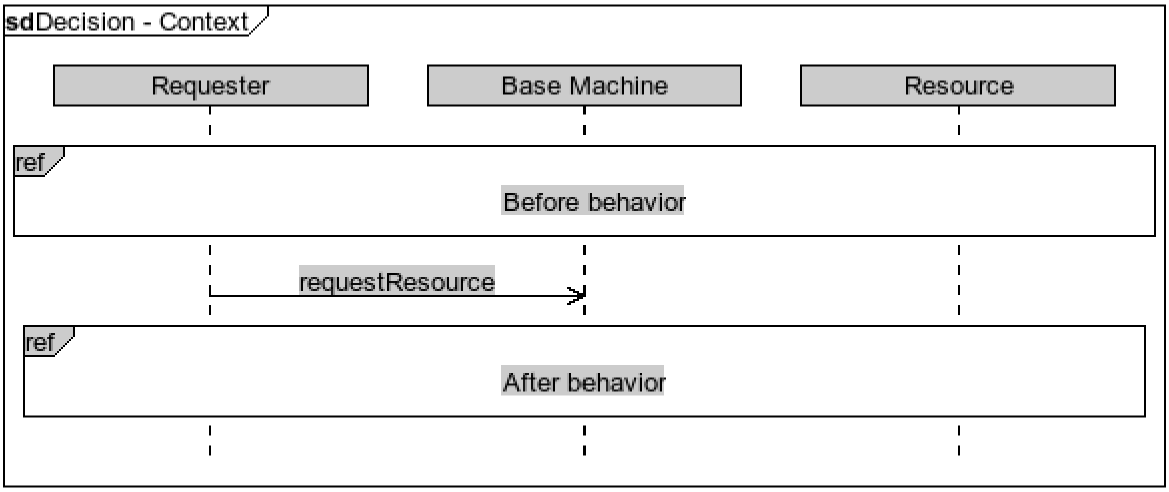


Figure 15.2.: *Relevant behavior of the base problem of Decision aspects*

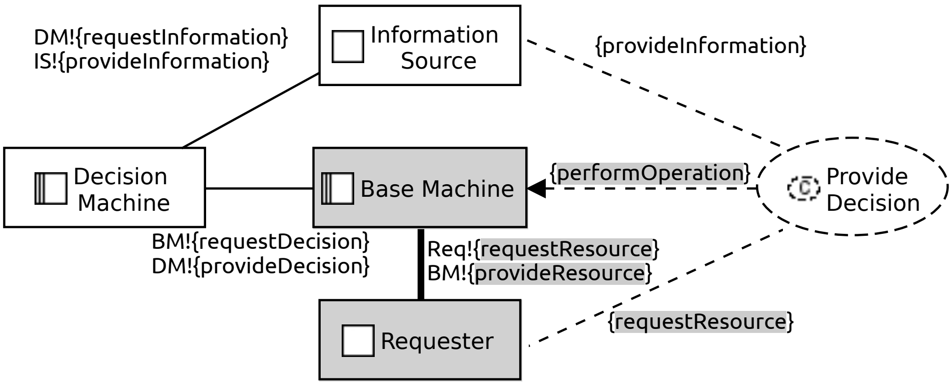


Figure 15.3.: *The structure of Decision aspects*

- *Reliability of decision:* The selected mechanism shall provide a reliable decision.
- *Security of decision process:* It shall be as hard as possible to by-pass the decision mechanism by malicious requesters.

Solution

Structure A mechanism to decide whether a base machine shall initiate the after behavior or not has the basic structure shown in Figure 15.3. This figure shows the Requester and the Base Machine as the relevant parts of the base problem (join points) for the Decision aspect. The Decision Machine is the domain that provides a decision to the Base Machine based on additional information provided by an Information Source. The Base Machine then uses this decision to determine whether it shall proceed with its behavior or not. The cross-cutting requirement Provide Decision refers to the request of the Requester and the used Information Source as basis to derive the decision, and it constrains the Base Machine to consider the provided decision. Note that the provided decision does not need to be Boolean. However, it shall be in a form that is easily accessable by the Base Machine without much further (computational) effort.

Behavior The behavioral part of the solution is shown in Figure 15.4. The interaction is started by the join point Requester that requests a resource from the join point Base Machine. The Base Machine then requests a decision for this request from the

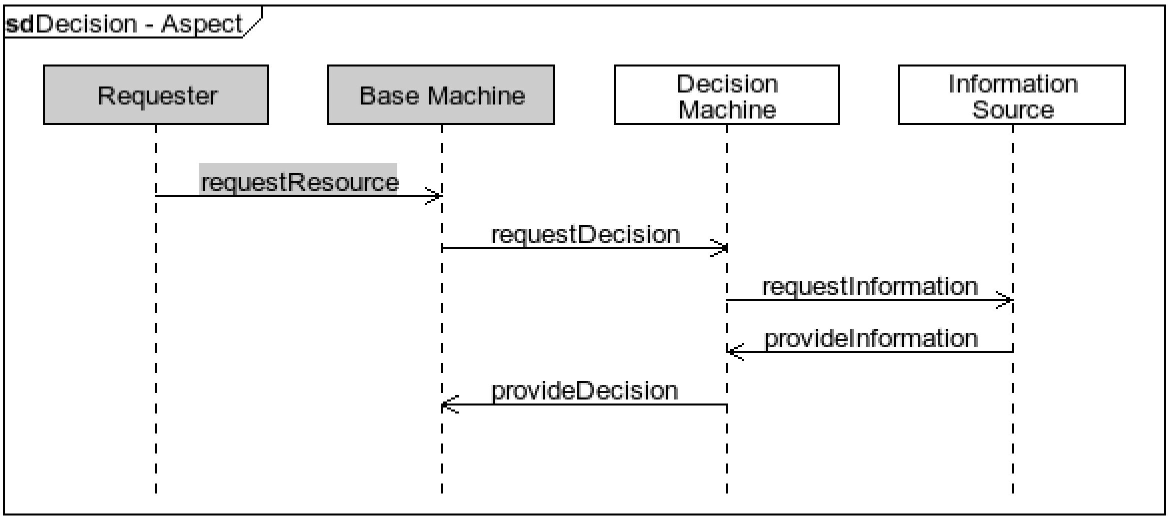


Figure 15.4.: *Behavior of Decision aspects*

aspect's machine Decision Machine. The Decision Machine provides a decision based on additional information that it retrieves from an Information Source beforehand.

Integration How a Decision aspect is integrated into a base problem is shown in Figure 15.5. First, the Before behavior of the base problem (cf. Figure 15.2) happens. In reaction to the Requester's request, the aspect's behavior is integrated (cf. Figure 15.4). Based on the provided decision the base machine has now to decide whether the base problem's After behavior is executed or not. Optionally, the base problem could also be extended to perform an Error behavior. Note that the message provideDecision of the Decision Machine could provide helpful information for the error behavior, e.g., the information why the decision was not positive.

Variants I distinguish three variants of Decision aspects depending on the type of the information source.

Data-based decision	Service-based decision	Human-based decision
If the information source is lexical, this means that the decision machine has to compute the decision based on the data contained in this lexical domain.	If the information source is causal, this means that an external service is used that provides necessary information to derive the decision.	If the information source is biddable, this means that the decision machine needs input of a human operator to provide a decision.

Consequences

Benefits

- *Complexity of the implementation:*

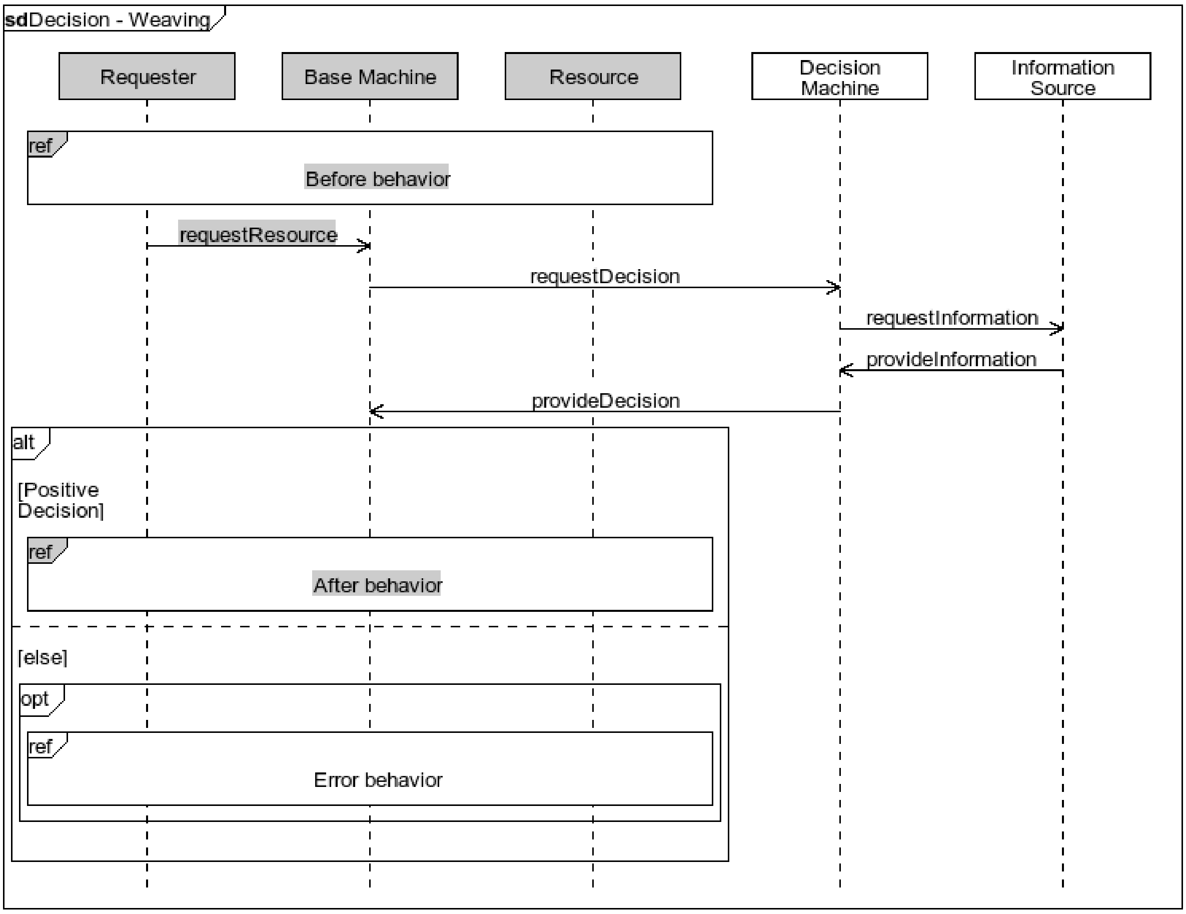


Figure 15.5.: *Integration of Decision aspects*

Data-based decision	Service-based decision	Human-based decision
-	As the decision making is outsourced, the implementation is only concerned with the correct usage of the service's API.	In general, we do not expect complex algorithms to be implemented if the decision making is based on humans.

- *Integration into the base problem:*

  All: -

- *Response time of decision:*

Data-based decision	Service-based decision	Human-based decision
The response time of data-based decisions benefits from the local availability and automated evaluation of the decision.	The computation of the decision is delegated to an external service. This can be an advantage if the computational capabilities of the machine are limited and it is not reasonable for it to manage the decision itself.	-

- *Reliability of decision:*

Data-based decision	Service-based decision	Human-based decision
The decision is under the control of the machine. Hence, its reliability is controllable by it.	A certified external service that is specialized on providing the needed decision can be considered as reliable.	Trained humans are in several situations the only option for reliable decisions. Especially in cases where the decision is based on requests whose semantics are hardly machine processable.

- *Security of decision process:*

Data-based decision	Service-based decision	Human-based decision
The decision is under the control of the machine. Hence, its security is controllable by it.	A certified external service that is specialized on providing the needed decision can provide the needed security.	Well trained personnel may be harder to be bypassed than automatic processes.

Liabilities

- *Complexity of the implementation:*

Data-based decision	Service-based decision	Human-based decision
We have to manage the lexical information source that, e.g, can be a database. The management of the lexical information source can introduce additional base problems that are concerned with adding, changing, or deleting information from the source. Depending on the mechanism, complex algorithms have to be implemented to derive the decision.	It has to be ensured that a reliable connection to the service exists and that its API is suitable for the needed purposes and correctly used.	A user interface has to be created for the humans that act as information source. This interface has to provide the humans all information that they need to take the decision and shall be usable.

- *Integration into the base problem:*

  All: The phenomenon requestResource of the base problem has in most cases to be enhanced in a way that it also transmits the information that is needed to decide whether the base machine shall perform the After behavior or not. The base machine has then to consider the provided decision. Optionally, an additional error behavior has to be implemented in the case of a negative decision.

- *Response time of decision:*

Data-based decision	Service-based decision	Human-based decision
The decision machine needs to have sufficient computational resources to compute the decision based on the information source in a timely manner.	To provide a timely decision, the external service has to have an appropriate response time and has to be available.	The response time depends on the availability of the human resources, their efficiency, and the number of requests that they have to handle. In general, we have to expect significant higher response times than for completely automatic decisions.

- *Reliability of decision:*

Data-based decision	Service-based decision	Human-based decision
The reliability of a data-based decision depends on the correctness of the information source. Hence, the machine has to ensure that it is accurate and up-to-date.	The external service has to be trusted to provide reliable information and decisions.	The humans involved in the decision process have to be trained to be able to provide correct decisions. Depending on the base problem it can be reasonable to involve more than one human in the decision process to improve the reliability of the made decisions.

- *Security of decision process:*

Data-based decision	Service-based decision	Human-based decision
The information source has to be sufficiently secured against unintended modification.	We have to trust the security claims of the used external service and have to ensure that the communication with the service is secure.	The humans involved in the decision process need to be trained in security to prevent, e.g., social engineering attacks and insecure behavior.

Examples

Data-based decision	Service-based decision	Human-based decision
Access control mechanisms, e.g., role-based access control (RBAC) and attribute-based access control (ABAC), are examples for data-based decisions.	Authentication or authorization based on an external service is an example for a service-based decisions. Examples for these mechanisms are “Login with Facebook” or “Pay with Amazon”.	Examples for human-based decisions are moderated internet forums or a report functionality of posts on social media sites. In moderated internet forums a moderator serves as information source and decides whether a post is published or not. For the report functionality of posts on social media sites, one or several humans have to decide whether the report is grounded and which action shall be performed.

15.3.2. Transform Received Data Aspect Frame

Name Transform received data aspect

Context

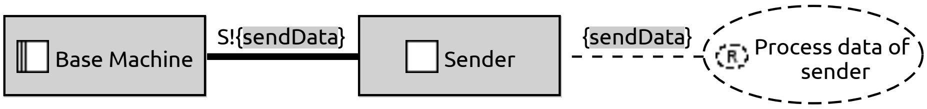


Figure 15.6.: *The base problem of Transform received data aspects*

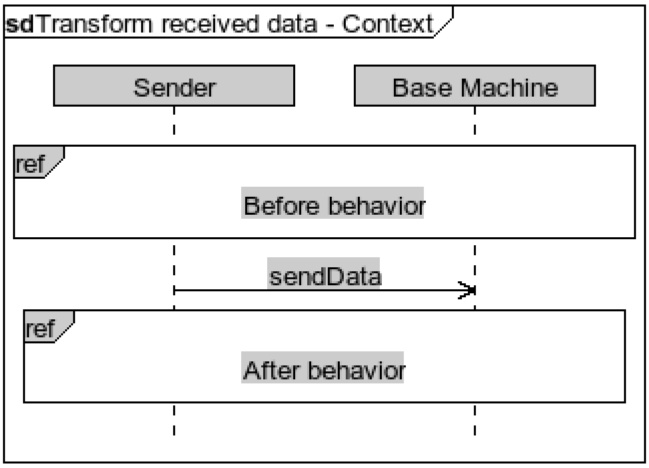


Figure 15.7.: *Relevant behavior of the base problem of Transform received data aspects*

Structure You have base problems in which the Base Machine needs to transform data received or retrieved from a domain Sender to proceed to achieve its requirement (see Figure 15.6).

Behavior Figure 15.7 shows the relevant behavioral view on the base problem. It consists of a Before behavior, the send event of the Sender to the Base Machine, and an After behavior.

Problem You want to integrate a mechanism that transforms the data received from the Sender in order to proceed with the processing of that data (After behavior).

Forces

- *Complexity of the implementation:* The complexity to implement the mechanism should be kept to the minimum needed.
- *Integration into the base problem:* It shall be possible to integrate the transformation mechanism in a way that the base problem is only affected in the desired manner.
- *Response time of transformation:* The selected mechanism shall respond in a timely manner depending on the needs of the base problem.
- *Reliability of transformation:* The selected mechanism shall perform the transformation reliably.

Solution

Structure A mechanism to transform data received from a sender to allow a further processing of these data has the basic structure shown in Figure 15.8. Figure 15.8 shows the Sender and the Base Machine as the relevant parts of the base problem (join points) for the Transform received data aspect. The Transformation Machine is the domain that performs the transformation of the received data using a Transformation Resource (optional) and provides the transformed data to the Base Machine. The

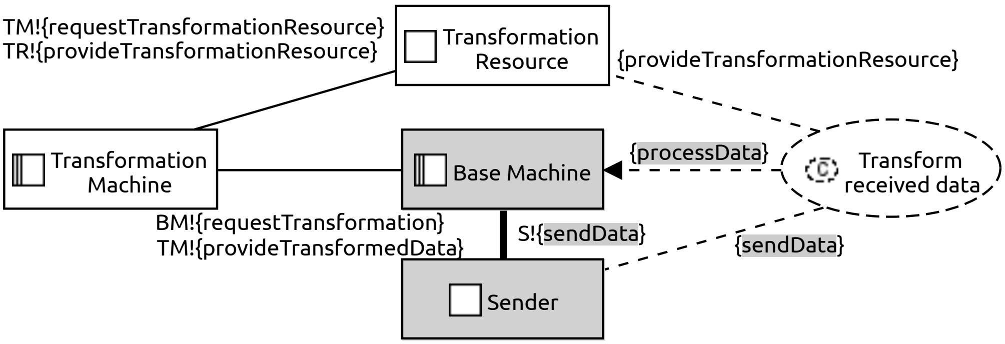

TM!{requestTransformationResource}
TR!{provideTransformationResource}
Transformation Resource
{provideTransformationResource}
Transformation Machine
Base Machine
{processData}
Transform received data
BM!{requestTransformation}
TM!{provideTransformedData}
S!{sendData}
{sendData}
Sender

Figure 15.8.: *The structure of Transform received data aspects*

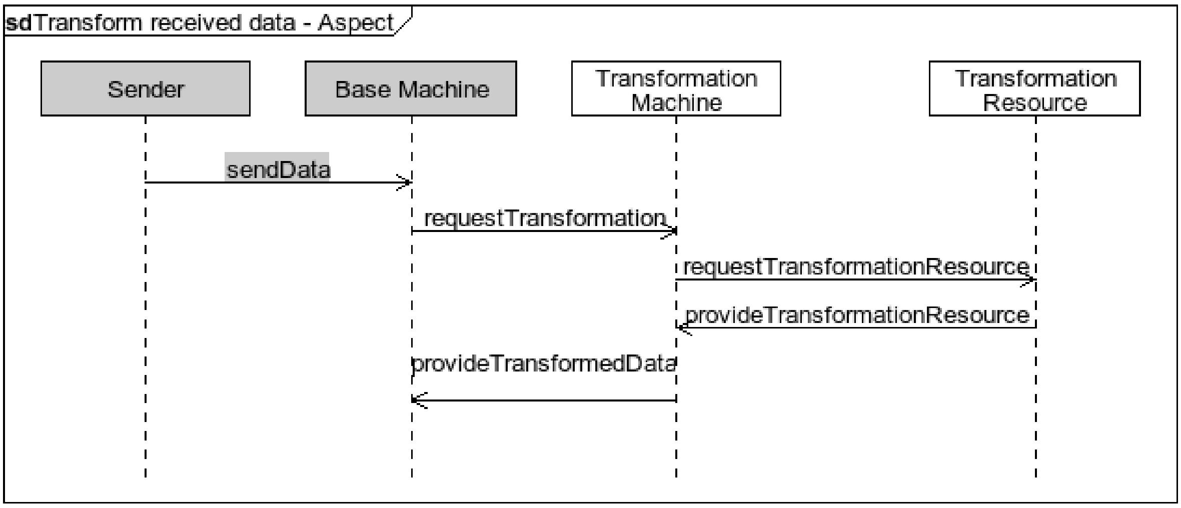


Figure 15.9.: *Behavior of Transform received data aspects*

Base Machine can then further process the transformed data. The cross-cutting requirement Transform received data refers to the sending event of the Sender and the Transformation Resource that is used to perform the transformation, and it constrains the Base Machine to process the transformed data.

Behavior The behavioral part of the solution is shown in Figure 15.9. The interaction is started by the join point Sender that sends data to the join point Base Machine. The Base Machine then requests the transformation of the received data from the aspect machine Transformation Machine. The Transformation Machine then provides the transformed data that it computed using the Transformation Resource.

Integration How a Transform received data aspect is integrated into a base problem is shown in Figure 15.10. First, the Before behavior of the base problem (cf. Figure 15.7) happens. After the Sender sends data, the aspects behavior is integrated (cf. Figure 15.9). Using the transformed data, the base machine executes the After behavior.

Variants I distinguish three variants of transformation aspects depending on the type of the transformation resource.

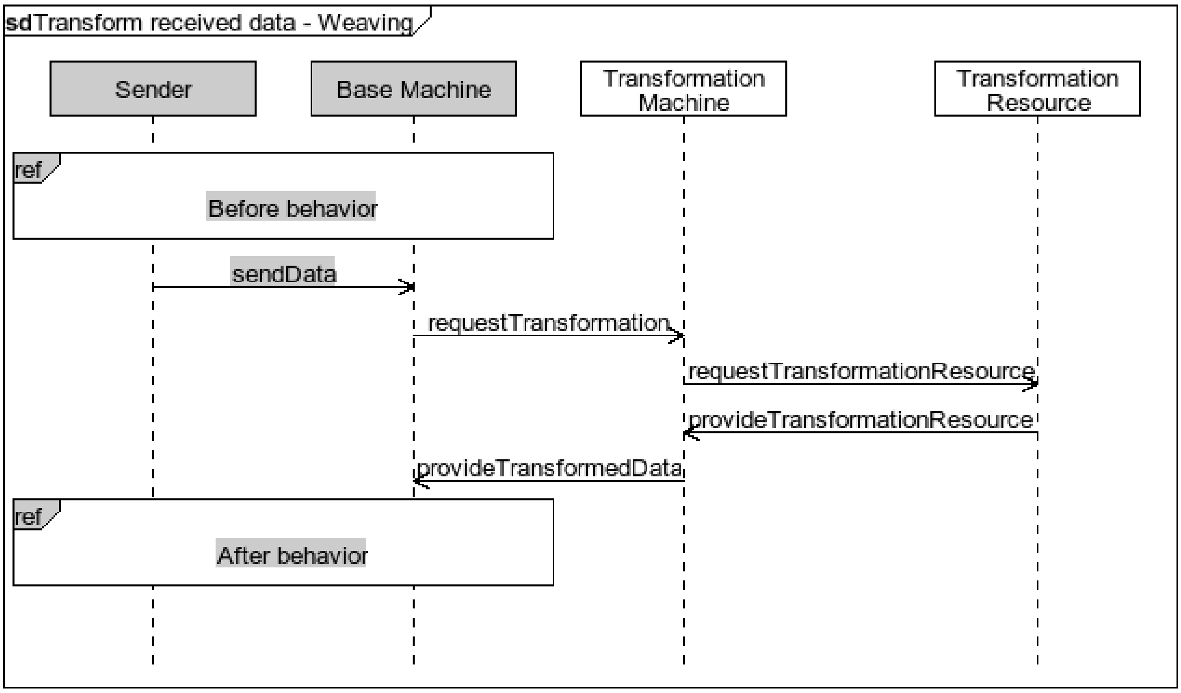


Figure 15.10.: *Integration of Transform received data aspects*

Data-based transformation	Service-based transformation	Human-based transformation
If the transformation resource is lexical, this means that the transformation machine uses this information as input for the transformation.	If the transformation resource is causal, this means that an external service exists that is used to perform the transformation.	If the transformation resource is biddable, this means that the transformation machine needs inputs from human operators to transform the received data.

Consequences

Benefits

- *Complexity of the implementation:*

Data-based transformation	Service-based transformation	Human-based transformation
-	As the transformation is outsourced, the implementation is only concerned with the correct usage of the service's API.	In general, we do not expect complex algorithms to be implemented if the transformation is performed by humans.

- *Integration into the base problem:*

  All: -

- *Response time of transformation:*

Data-based transformation	Service-based transformation	Human-based transformation
The response time of a data-based transformation benefits from the local availability and automatic transformation.	The transformation is delegated to an external service. This can be an advantage if the computational capabilities of the machine are limited and it is not reasonable for it to perform the transformation itself.	-

- *Reliability of transformation:*

Data-based transformation	Service-based transformation	Human-based transformation
The transformation is under the control of the machine. Hence, its reliability is controllable by it.	A certified external service that is specialized on providing the needed transformation can be considered as reliable.	Trained humans are in several situations the only option for reliable transformations. Especially in cases where the transformation is based on requests whose semantics are hardly machine processable.

Liabilities

- *Complexity of the implementation:*

Data-based transformation	Service-based transformation	Human-based transformation
We have to manage the lexical transformation resource that, e.g, can be a database. The management of the lexical transformation resource can introduce additional base problems that are concerned with adding, changing, or deleting information from the resource. Depending on the mechanism, complex algorithms have to be implemented to perform the transformation.	It has to be ensured that a reliable connection to the service exists and that its API is correctly used.	A user interface has to be created for the humans that act as transformation resource. This interface has to provide the humans all information that they need to perform the transformation and shall be usable.

- *Integration into the base problem:*

  All: The base machine has to use the provided transformed data instead of the received data for the further processing.

- *Response time of transformation:*

Data-based transformation	**Service-based transformation**	**Human-based transformation**
The transformation machine needs to have sufficient computational resources to compute the transformation based on the transformation resource in a timely manner.	To provide the transformed data timely, the external service has to have a appropriate response time and has to be available.	The response time depends on the availability of the human resources, their efficiency, and the number of requests that they have to handle. In general, we have to expect significant higher response times than for completely automatic transformations.

- *Reliability of transformation:*

Data-based transformation	**Service-based transformation**	**Human-based transformation**
The reliability of a data-based transformation depends on the correctness of the transformation resource. Hence, the machine has to ensure that it is accurate and up-to-date.	The external service has to be trusted to provide reliable transformations.	The humans that perform the transformation have to be trained to be able to provide correct transformations. Depending on the base problem it can be reasonable to involve more than one human in the transformation process to improve the reliability of the made transformations.

Examples

Data-based transformation	**Service-based transformation**	**Human-based transformation**
Decryption of data received by a sender is an example for a data-based transformation. In this case the transformation resource is a key storage.	An optical character recognition (OCR) or face recognition software could be used in a service-based transformation as information resource to derive information that the base machine needs for a further processing.	Examples for human-based transformations are spell-checking or formatting of provided texts.

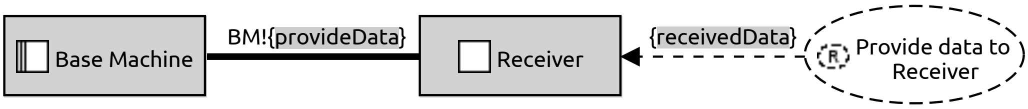


Figure 15.11.: *The base problem of Transform before transmission aspects*

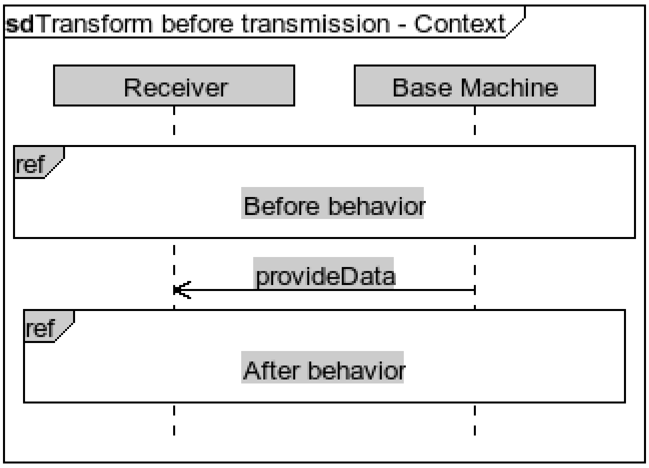


Figure 15.12.: *Relevant behavior of the base problem of Transform before transmission aspects*

15.3.3. Transform Before Transmission Aspect Frame

Name Transform before transmission aspect

Context

Structure You have base problems in which the Base Machine needs to transform data (previously retrieved by the base machine) that shall be sent to or stored by a Receiver (see Figure 15.11).

Behavior Figure 15.12 shows the relevant behavioral view on the base problem. It consists of a before behavior, the event that the Base Machine provides data to the Receiver, and an after behavior.

Problem You want to integrate a mechanism that transforms the data before these are provided to a Receiver.

Forces

- *Complexity of the implementation:* The complexity to implement the mechanism should be kept to the minimum needed.
- *Integration into the base problem:* It shall be possible to integrate the transformation mechanism in a way that the base problem is only affected in the desired manner.
- *Response time of transformation:* The selected mechanism shall respond in a timely manner depending on the needs of the base problem.
- *Reliability of transformation:* The selected mechanism shall perform the transformation reliably.

Solution

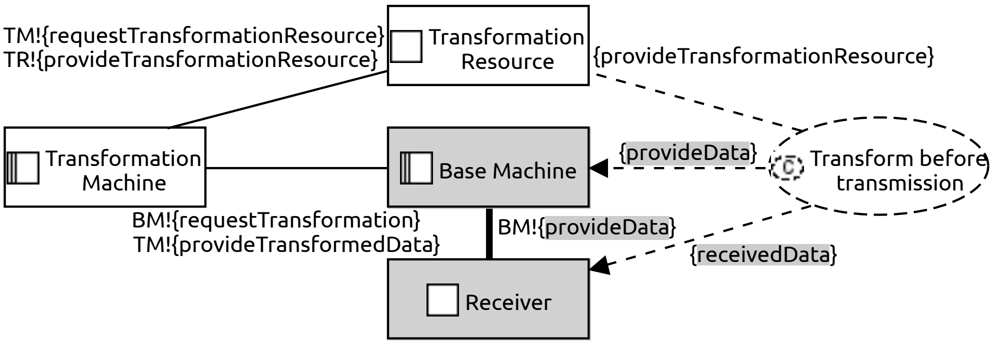

TM!{requestTransformationResource}
TR!{provideTransformationResource}
Transformation Resource
{provideTransformationResource}
Transformation Machine
Base Machine
{provideData}
C
Transform before transmission
BM!{requestTransformation}
TM!{provideTransformedData}
BM!{provideData}
{receivedData}
Receiver

Figure 15.13.: *The structure of Transform before transmission aspects*

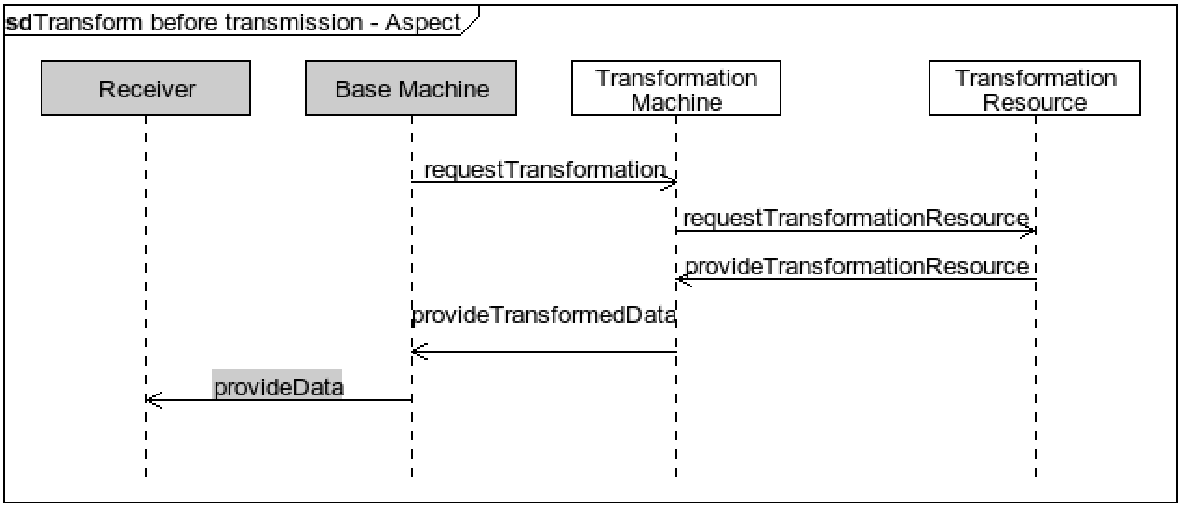


Figure 15.14.: *Behavior of Transform before transmission aspects*

Structure A mechanism to transform data to provide these transformed data to a receiver has the basic structure shown in Figure 15.13. Figure 15.13 shows the Receiver and the Base Machine as the relevant parts of the base problem (join points) for the transform before transmission aspect. The Transformation Machine is the domain that performs the transformation using a Transformation Resource (optional) and provides the transformed data to the Base Machine. The Base Machine can then provide the transformed data to the Receiver. The cross-cutting requirement Transform before transmission refers to the Transformation Resource that is used to perform the transformation, and it constrains the Base Machine to provide these data to the Receiver and that the Receiver will receive the transformed data.

Behavior The behavioral part of the solution is shown in Figure 15.14. The interaction is started by the join point Base Machine that requests the transformation of the received data from the aspect's machine Transformation Machine. The Transformation Machine then provides the transformed data that it computed using the Transformation Resource. Finally, the Base Machine provides the transformed data to the Receiver.

Integration How a Transform before transmission aspect is integrated into a base problem is shown in Figure 15.15. First, the Before behavior of the base problem (cf. Figure 15.12) happens. Before the Base Machine provides data to the Receiver, the aspects behavior is integrated (cf. Figure 15.14). After providing the transformed

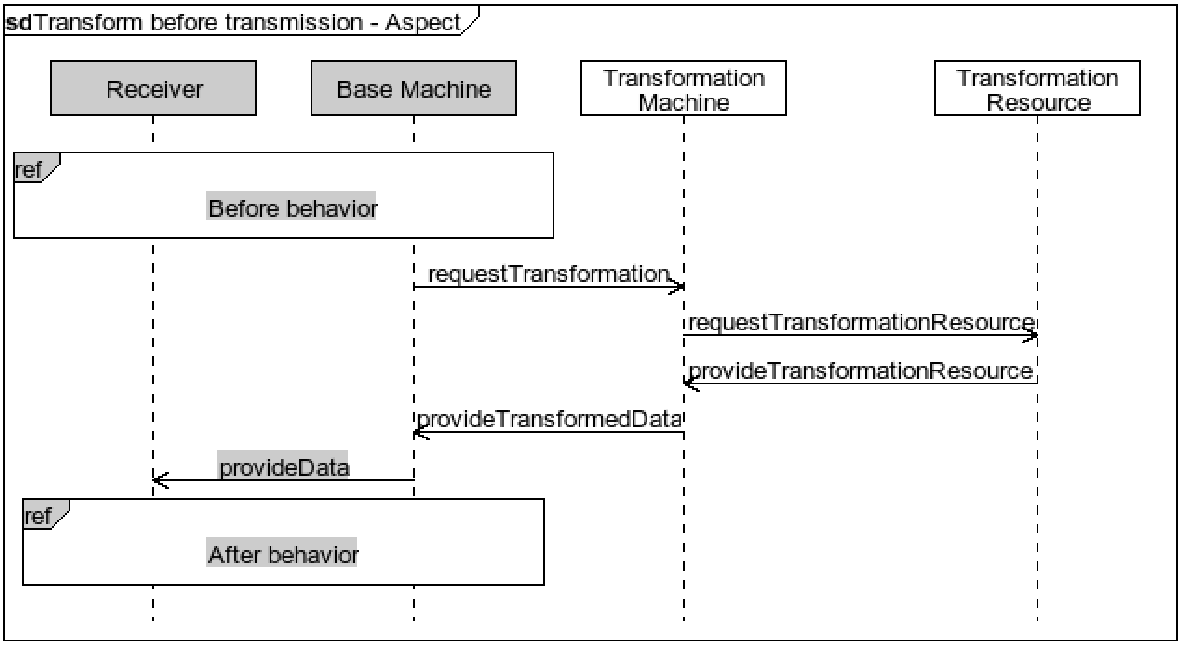


Figure 15.15.: *Integration of Transform before transmission aspects*

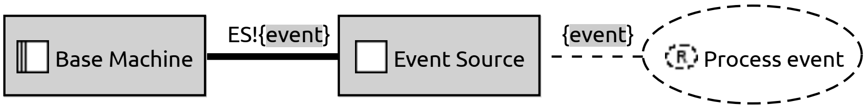


Figure 15.16.: *The base problem of Independent behavior aspects*

data to the Receiver, the After behavior is executed.

Variants I distinguish the same three variants of transformation aspects as in Section 15.3.2 depending on the type of the transformation resource.

Consequences Most consequences of the Transform before transmission aspect are the same as for the Transform Received Data Aspect or apply analogously. Hence, I refer to Section 15.3.2 for the consequences of this aspect frame.

Examples

For examples, I also refer to Section 15.3.2.

15.3.4. Independent Behavior Aspect Frame

Name Independent behavior aspect

Context

Structure You have base problems into which an additional behavior shall be integrated in reaction to specific events caused by an Event Source that the Base Machine observes (see Figure 15.16). The additional behavior shall not influence the behavior of the Base Machine, i.e., the Base Machine's behavior shall be independent of the additional behavior.

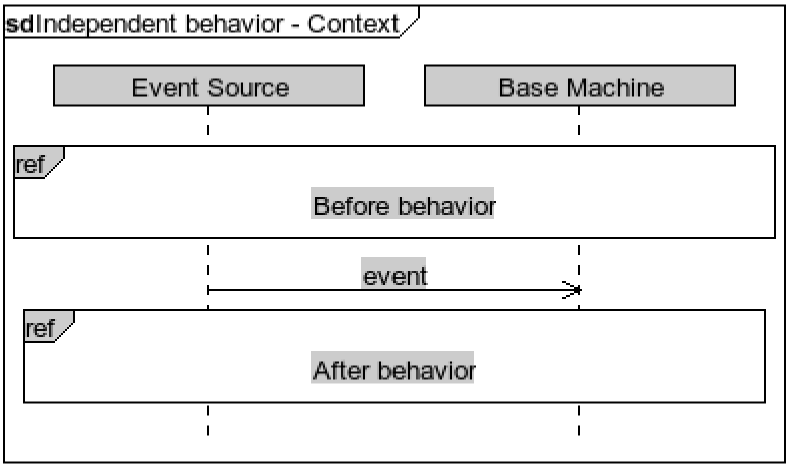


Figure 15.17.: *Relevant behavior of the base problem of Independent behavior aspects*

Behavior Figure 15.17 shows the relevant behavioral view on the base problem. It consists of a before behavior, the event that the Base Machine receives from the Event Source, and an after behavior.

Problem You want to integrate a mechanism that performs a task in reaction to an event caused by an Event Source and the performance of the additional task shall not further influence the behavior of the Base Machine.

Forces

- *Complexity of the implementation:* The complexity to implement the mechanism should be kept to the minimum needed.
- *Integration into the base problem:* It shall be possible to integrate the independent behavior in a way that the base problem is not affected by it.
- *Reliability of independent behavior:* The selected mechanism shall perform the independent behavior reliably.

Solution

Structure A mechanism to perform an independent behavior in reaction to an event caused by an Event Source is shown in Figure 15.18. Figure 15.18 shows the Event Source and the Base Machine as the relevant parts of the base problem (join points) for the independent behavior aspect. The Independent Machine is the domain that performs the independent behavior by influencing the Influenced Domain. The cross-cutting requirement Independent behavior refers to the Event Source that issues an event and the Base Machine that triggers the Independent Machine. The behavior of the Influenced Domain is constrained by the cross-cutting requirement.

Behavior The behavioral part of the solution is shown in Figure 15.19. The interaction is started by the join point Event Source that issues an event that the join point Base Machine observes. The Base Machine then triggers the Independent Machine to cause the additional behavior that the Influenced Domain shall perform.

Integration How an Independent behavior aspect is integrated into a base problem is shown in Figure 15.20. First, the Before behavior of the base problem (cf. Figure 15.17) happens. In reaction to the Event Source's event, the aspect's behavior is

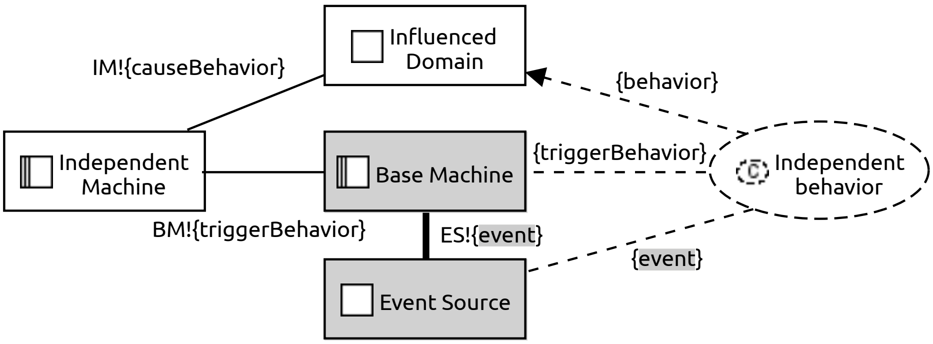


Figure 15.18.: *The structure of Independent behavior aspects*

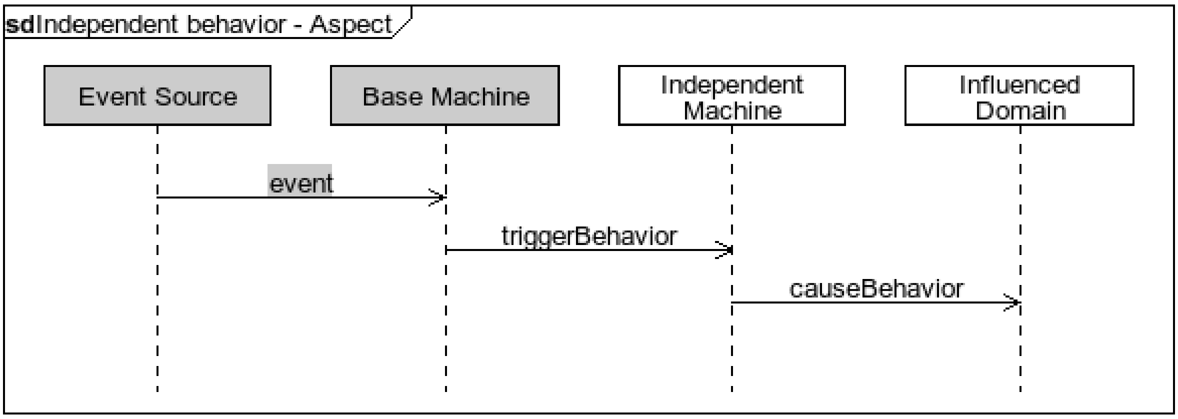


Figure 15.19.: *Behavior of Independent behavior aspects*

integrated (cf. Figure 15.19). In parallel to the aspect's behavior, the base problem's After behavior is executed.

Variants I distinguish three variants of Independent behavior aspects depending on the type of the influenced domain.

Influenced data	Influenced service	Influenced human
If the influenced domain is lexical, this means that the data managed by this domain shall be modified.	If the influenced domain is causal, this means that a technical device, service or the like shall be controlled.	If the influenced domain is biddable, this means that information shall be provided to humans such that they behave in a desired way.

Consequences

Benefits

- *Complexity of the implementation:*

  All: -

- *Integration into the base problem:*

  All: Outsourcing the independent behavior to the independent machine allows to parallelize the after behavior of the base machine and the independent behavior.

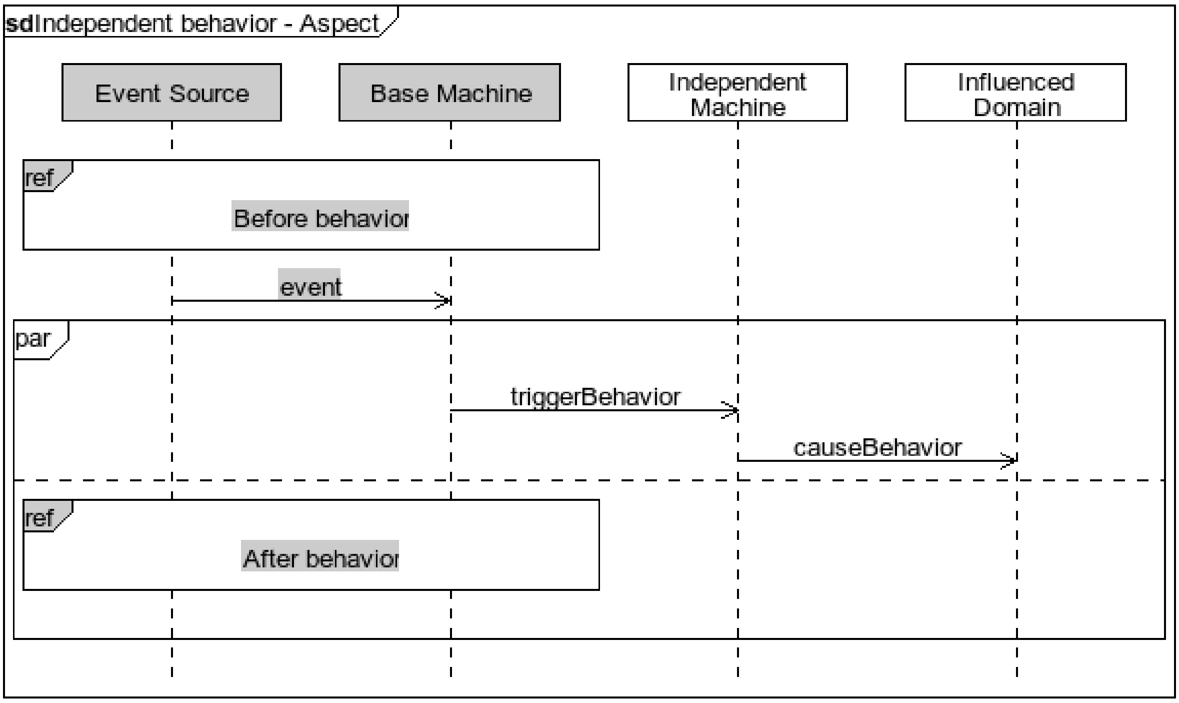


Figure 15.20.: *Integration of Independent behavior aspects*

- *Reliability of independent behavior:*

  All: -

Liabilities

- *Complexity of the implementation:*

Influenced data	Influenced service	Influenced human
The complexity varies based on the complexity of the desired independent behavior. An independent behavior aspect about influencing data could be about adding a single entry to a database, or additionally involve more complex operations like consistency checks.	It has to be ensured that a reliable connection to the service exists and that its API is correctly used.	An appropriate interface to the humans has to be chosen. This interface has to be designed in a way that the aspect machine is sufficiently be able to influence the humans in the desired way. For example, if the independent behavior is to display information to humans, then the interface could be a GUI on a computer. This GUI has to be designed in a way that the human notices the displayed information.

- *Integration into the base problem:*

  All: The base machine has to provide sufficient information to allow the independent machine to perform the independent behavior. Additionally, it has to be ensured that the behavior of the independent machine and the influenced domain do not influence the base machine's behavior, e.g., due to side-effects caused by the usage of shared resources.

- *Reliability of independent behavior:*

Influenced data	Influenced service	Influenced human
If data are influenced, the reliability depends on the correctness of the independent machine's implementation and the consistency of the influenced data.	The external service has to be reachable and to comply to its specification.	To influence humans in the desired way, actuators (refining the interface between the independent machine and the influenced humans) have to be chosen that are noticed by the humans and cause them to behave in the desired way. In general, it is necessary to explain to the humans how, when, and why they are informed and what their reaction to this information shall be.

Examples

Influenced data	Influenced service	Influenced human
Logging of specific events is an example for an independent behavior that influences data. In this case, the influenced domain is a log file or database, to which the new event with additional information, e.g. a timestamp, is added.	All examples of independent behavior aspects that influence humans are at some point refined to aspects that influence a service. For example, if we consider an alarm system in a hospital that informs nurses about critical vital signs of patients (event), then the interface to the nurses will be refined with, e.g., a display and/or speaker in the nurse's station. The independent machine is then only concerned with influencing this display and/or speaker in the desired way.	An example for an independent behavior that influences humans is an alarm system in a hospital that informs nurses about critical vital signs of patients (event).

15.4. Discussion

In this section, I provide further guidance on how aspect frames shall be created and how they can be used. I discuss the lessons that I learned during the creation of the four aspect frames presented in the previous section.

15.4.1. Name

As usual, the name of a pattern should be carefully selected. An aspect frame's name shall provide a clear intuition about the kinds of problems the aspect frame's instances shall address. For example, the shown Decision aspect is concerned with providing decisions that are needed in a base problem to decide on the further processing.

15.4.2. Context

The specification of the aspect frame's context is not easy. The base problem description has, on one hand, to be general enough to fit to all base problems instances the aspect frame shall be integrated into. On the other hand, the context should be as specific as possible to ensure that aspects of the frame are only integrated into fitting base problems and to support requirements engineers to decide whether the aspect they want to model fits to the aspect frame.

For example, I decided that the base problem of Decision aspects consists of a base machine, a resource, and a requester (cf. Figure 15.1). The latter two have no fixed domain type assigned. That is, they possibly be instantiated with biddable, causal, or lexical domains. A more general description of the Decision aspect could have left out the resources that the requester wants to access. This would not even have much impact on the other diagrams of the Decision aspect, because the aspect's behavior is not concerned with interacting with the resource (cf. Figure 15.4). However, the decision is based also on the kind of resource that the requester wants to access. Hence, it is important that the base problems contain the resource that shall be accessed.

A too specific description of the Decision aspect could, e.g., limit the join point requester to be a biddable domains and the resource to be a lexical domain. This would limit the application of Decision aspects to base problems in which humans want to access data which are controlled by the machine. Such a restriction would be too strong, because the types of the requester and resource are not relevant to provide the decision for most aspects that fit to the Decision aspect.

The behavioral view on the base problem shall explicitly specify the situation in which the frame's aspects shall be integrated into it. This can be, e.g., a single message, a specific sequence of messages, or situations described by a state predicate. In aspect-oriented programming, this situation description is called the aspect's *pointcut* (cf. Chapter 14). The behavior before and after the pointcut may be introduced as references (cf. Figure 15.2). The before and after behavior can then be used in the weaving to specify a modification of the base problem's behavior in reaction to the aspect's behavior. For Decision aspects, the pointcut consists only of the requester's request to get access to the resource (cf. Figure 15.2), because it shall be asked for a decision about the access to a resource if and only if the resource is requested.

15.4.3. Problem

The problem shall be a short statement that condenses the intention that all instances of the aspect frame share. This problem statement shall help requirements engineers to decide whether their aspect addresses the described problem and is potentially an instance of the frame.

15.4.4. Forces

The forces of an aspect frame summarize issues that make it difficult to solve the problem existing in the described context. The forces help to further understand the properties of instances of an aspect frame. During the development of four aspect frames, I identified two forces that are relevant for all aspect frames. These are the *complexity of the implementation* and the *integration into the base problem* that are obviously relevant for all kinds of aspects, because all aspects have to be implemented and integrated into a base problem. Additionally, it may be worth it to discuss the software qualities that instances of an aspect frame should have. For example, I identified issues concerning the *response time*, *reliability*, and *security* as relevant for Decision aspects.

15.4.5. Solution

Similar to the description of the base problems, it is important to describe the aspect frame generally enough, but not too generally. For example, if the requester was left out in Figure 15.3, then the relevant interface for the decision machine to the base machine that is needed to request the decision and to provide the result of the decision would still be contained, but the original request on which the base machine reacts would no longer be part of the aspect. In this case, the original trigger (pointcut) of the interaction into which the aspect needs to be integrated would be missing (cf. Section 15.4.2). The pointcut is needed to clearly specify when the aspect's behavior shall be weaved into the base problem's behavior. If no pointcut is contained in an aspect frame, it is more difficult to decide how the aspect has to be integrated into the affected base problems. Consequently, the Decision aspect's behavior description (see Figure 15.4) contains the pointcut described in Figure 15.2, namely the message requestResource. This message is the first message in the aspect's behavior description, i.e., the aspect's behavior shall be performed after this message.

It would be reasonable to add the requested resource to Figure 15.3, because the decision is likely to depend on both the requester and the resource. However, the information which resource is requested is implicitly encoded in the phenomenon requestResource and the resource itself is not an actor in the aspects behavior (cf. Figure 15.4). Hence, I decided to leave out the resource in Figure 15.3 to keep the diagram simple.

The integration description shall explain how the aspect's behavior shall be added to the base problem's behavior. Ideally, the aspect's behavioral view contains the base problem's pointcut and the aspect can easily be integrated into the base problem. For example, the Decision aspect's behavioral description contains the base problem's pointcut as first message. Hence, the aspect's behavior just needs to be added after this message (see Figure 15.5).

Additionally, the sequence diagram for the integration of aspects may describe deviations of the base problem's behavior in reaction to the aspect. For Decision aspects, I specified that the base problem's after behavior shall only be executed if the provided decision is positive. If the decision is negative, then optionally an error behavior may be performed by the base machine. This error behavior strongly depends on the concrete context of the base problem and does itself not belong to the Decision aspect.

I introduce the concept of *variants* to aspect frames to be able to discuss different flavors of an aspect. A variant assigns specific types to domains without specified type. Variants should be used if the aspect's structure, behavior, and integration do not depend on the type of an involved domain, but the benefits and liabilities may vary for different types of it. For example, the information source in Decision aspects has no specific type and the structure, behavior, and integration of Decision aspects do not depend on its type. However, the properties of the Decision aspects, i.e., the benefits and liabilities, are different depending on the information source's concrete type. Hence, I distinguish the Decision aspect variants data-based decision, service-based decision, and human-based decision. If a domain without specified type shall not be instantiated with a specific type, this can also be specified by declaring that this variant is not allowed or possible. Note that it is not mandatory to describe variants for all domains without specified type. For example, the requester and the resource have both no specified type in the decision aspect, and they are not mentioned in the variants. This means that there are no restrictions on the type of these domains and that the benefits and liabilities are not expected to vary for different types of these.

15.4.6. Consequences

The positive and negative consequences should discuss all stated forces for all defined variants. If a consequence is identified that has not yet a corresponding force, this indicates that a corresponding force is missing and that it should be added.

The consequences of an aspect frame shall help requirements engineers to specify the relevant properties of the concrete mechanism they want to express as an aspect. These properties can also be used to compare aspects that are instances of the same aspect frame with each other.

15.4.7. Examples

The examples shall help requirements engineers to understand the kind of problems that belong to the aspect frame and consequently, to decide whether the mechanism they want to express as an aspect fits to the class described by the aspect frame. If variants are specified, then at least one example should be provided for each variant.

15.5. Related Work

I provide an overview of aspect-oriented requirements engineering methods and their relation to AORE4PF in Chapter 14.5. As stated in that section, Lencastre et al. (2008) also propose to use problem frames in the context of aspect-oriented requirements engineering. However, Lencastre et al. do not propose frames for reoccurring cross-cutting concerns. Alebrahim et al. (2012b) show how role-based access control (RBAC) can be integrated into specific problem frames in an aspect-oriented way. The RBAC aspect proposed by Alebrahim et al. can be classified as a Decision aspect. To the best of my knowledge, there is not yet any research on patterns to express classes of aspects in the field of AORE.

In the following, I discuss literature that considers both aspect-orientation and patterns. Clarke and Walker (2001) present an approach to model design patterns as composition patterns. They propose UML templates that capture the structure and behavior of the design patterns independently of the design elements they may cross-cut. Additionally, Clarke and Walker show how their proposed composition patterns can be translated to the aspect-oriented programming language AspectJ. While Clarke and Walker express aspects in the design phase, I consider aspects in the requirements phase. Furthermore, I provide with the aspect frames pattern candidates that help to describe aspects that fit into the class of aspects described by the frame.

Garcia et al. (2006) performed a quantitative study that investigates the advantage of using aspects to implement 23 Gang-of-Four patterns (Gamma et al., 1995), which have shown to be cross-cutting in several cases. The authors found out that an aspect-oriented implementation of design patterns leads to a better separation of concerns, but introduces also issues like more complex operations and more lines of code in some cases.

15.6. Conclusion

In this chapter, I introduced aspect frames that represent classes of aspects (cross-cutting functional requirements) that share a common concern and behavior. The aspect frames are described in a pattern format that summarizes the shared problem and the common solution provided by aspects of the frame's class. Aspect frames are a means to support the description of aspects in AORE, because requirements engineers can use the aspect frames as blueprints for aspects and fit their cross-cutting concerns to an aspect frame. The aspect frame provides requirements engineers with further information about forces of the problem, and the benefits and liabilities that aspects fitting to the frame might have.

I described a pattern format that can be used to describe aspect frames in a structured way. Using this pattern format, I introduced the aspect frames Decision aspect, Transform received data aspect, Transform before transmission aspect, and Independent behavior aspect as examples of aspect frames. Finally, I summarized my lessons learned to provide guidance on how to create and use aspect frames.

In future work, it could be studied whether there are additional classes of cross-cutting concerns that could be represented as aspect frames and evaluate how much aspect frames help requirements engineers to specify cross-cutting concerns.

In the next chapter, I show how privacy enhancing technologies (PETs) can be represented as aspects, also using a pattern format. In this way, I provide the foundation for a structured catalog of PETs containing the information about the PETs that the analysis team needs to select and integrate privacy measures in the step Privacy Measure Integration of the ProPAn method (see Chapter 17). The aspects created for the PETs are all instances of aspect frames.

Chapter 16

Privacy Enhancing Technology Patterns

In this chapter, I introduce *PET patterns* as a means to document privacy enhancing technologies (PETs) in a reusable way for requirements engineers who have to operationalize privacy requirements. For a comprehensible and reusable documentation, I propose a pattern format that can be used to capture the most relevant information about a PET for a requirements engineer. As many PETs involve functionalities that need to be integrated into different other functionalities, I propose to document these functionalities using the aspect-oriented notation of the AORE4PF method introduced in Chapter 14, and by instantiating the aspect frames introduced in Chapter 15. These PET patterns may be used by the analysis team in the step Privacy Measure Integration of the ProPAn method (see Chapter 17) as support for selecting PETs that operationalize privacy requirements or mitigate privacy risks, and to integrate these into the system-to-be.

This chapter is based on (Meis and Heisel, 2017d) of which I am the main author. My co-author Maritta Heisel provided substantial feedback that helped to improve the paper.

An introduction to this chapter is given in Section 16.1. I introduce the pattern format to represent PETs in Section 16.2. Two example instantiations of the format follow in Section 16.3. I discuss the contribution of this chapter in Section 16.4. Related work is presented in Section 16.5, and Section 16.6 concludes the chapter.

16.1. Introduction

Regulations, such as the EU General Data Protection Regulation (European Commission, 2016), and industrial standards, such as ISO 29100 (ISO/IEC, 2011), emphasize that privacy shall already be considered from the very beginning in software development. To realize the privacy requirements of the software-to-be, privacy enhancing technologies (PETs) may be used at different stages. First, the selection of a PET can emerge from the given requirements. For example, it could be an initial or an identified privacy requirement that an anonymous authentication scheme shall be used to ensure the authenticity and correctness of personal data provided by data subjects, e.g., end-users, without revealing too much information to the software-to-be and consequently its controller. Second, during a privacy risk analysis it can become apparent that the integration of PETs is necessary to mitigate unacceptable privacy risks. For example, it could be identified that specific data need first to be anonymized before they are transmitted, or that a mechanism needs to be integrated to inform end-users about the controller's privacy policy.

In both situations, requirements engineers face the following questions.

1. How to find out whether and which PETs exist with the needed properties?

2. How to select from a set of PETs addressing a privacy requirement the most appropriate for the system-to-be?

3. How to align the selected PET with the other requirements? That is, the selected PET needs to be integrated into one or more other functional requirements to satisfy the desired privacy requirement, or to mitigate a privacy risk.

In this chapter, I propose a pattern-based representation of PETs that aims at assisting requirements engineers to answer questions 1. and 2. by providing a common structure to describe PETs. This structure shall help requirements engineers to assess whether a PET can be integrated into their software system (question 1.), and to compare the benefits and liabilities of different PETs to select the best-fitting PET (question 2.).

To support question 3., I propose to consider PETs as early aspects. PETs (or parts of them) describe *cross-cutting functionality* that is integrated into the *base functionality* of the software-to-be to ensure certain privacy properties, e.g., anonymity and transparency, or to mitigate specific privacy threats, e.g., eavesdropping and unawareness. The cross-cutting functionalities are also called *aspects* in aspect-oriented requirements engineering. Aspects are described independently from the base functionality they shall be integrated into. Additionally, it needs to be described how an aspect is integrated into the base functionality. This integration is also called *weaving*. For more details on aspect-oriented requirements engineering see Chapter 14.

16.2. Pattern Format for PET Patterns

Table 16.1 shows the pattern format that I propose to represent PETs. The audience of the PET patterns are requirements engineers who want to identify, select, and integrate PETs that address certain privacy requirement they have to consider. The patterns themselves can be created by anyone who is familiar with the respective PET and the aspect-oriented problem frames notation (see Chapter 14). The pattern format is based on the suggestions of Harrison (2003); Wellhausen and Fießer (2011).

The pattern sections **Motivation**, **Context**, **Problem**, **Privacy Forces**, and **General Forces** are concerned with the kind of system the PET can be integrated into. These pattern sections can be used by requirements engineers to assess whether they can use the PET or not. I propose to describe the **Context** using a high-level context diagram that contains the machines, domains, and interfaces that such a system typically has as join points. The base problems in the pattern section **Problem** shall be presented as high-level problem diagrams using the machines, domains, and interfaces of the context diagram provided in the **Context**. Additionally, a behavioral view on the base problems shall be specified for the later description of the *Weaving*.

I suggest to describe the *forces* that make it difficult to address the problem that the PET provides a solution for. First, the **Privacy Forces** shall be documented, i.e., the privacy requirements that the PET shall address and may impact. As a reference, I use the privacy requirements of ProPAn's privacy requirements taxonomy that I introduced in Chapter 7. However, the list can also be extended with further privacy requirements, just as ProPAn's privacy requirements taxonomy may be extended if this is necessary. Second, **General Forces** shall be discussed. I identified six generic general forces that correspond to different software qualities (cf. Table 16.1). The list of **General Forces** may also be extended when additional relevant software qualities are identified that are relevant for a PET.

The PET itself and its consequences are considered in the pattern sections **Solution**, **Design Issues**, **Privacy Benefits**, **General Benefits**, **Privacy Liabilities**, **General Liabilities**, and **Examples**. The **Solution** is structured into five subsections. First, it contains a *General Overview* of the domains involved in the PET (including the join points of the **Context**) and

Table 16.1.: *The pattern format for PET patterns*

1 Name	All known names of the PET
2 Motivation	Example scenarios that show the necessity of the PET
3 Context	Description of the systems the PET can be integrated into.
4 Problem	Description of the system's base problems with privacy requirements the PET shall address.
5 Privacy Forces	Privacy requirements in the **Problem** the PET addresses
5.1 Data Confidentiality	Personal data (PD) shall be kept secret
5.2 Integrity	PD shall be correct and up-to-date
5.3 Availability	PD shall be accessable
5.4 Anonymity	PD shall not be linkable to the data subject (DS)
5.5 Data Unlinkability	PD shall not be linkable to each other
5.6 Undetectability	Existence or occurrence of PD shall not be recognizable
5.7 Pseudonymity	Only pseudonyms shall be linkable to the PD
5.8 Collection Information	DS shall be informed about data collection
5.9 Storage Information	DS shall be informed about storage procedures
5.10 Flow Information	DS shall be informed about flows of PD to others
5.11 Exceptional information	DS and authorities shall be informed about breaches
5.12 Data subject intervention	DS shall be able to intervene in the processing
5.13 Authority Intervention	Authorities shall be able to intervene in the processing
6 General Forces	Other issues making it difficult to address the **Problem**
6.1 End-user friendliness	The PET's influence on the user experience
6.2 Performance	The PET's impact on the system's performance
6.3 Costs	Costs and effort to be spent emerging from the PET
6.4 Impact on functionality	Potential effects of the PET on the system
6.5 Abuse of PET	Unintended usage of the PET
6.6 Revocation	Possibilities to abolish privacy properties of the PET
7 Solution	Description of the PET and how it can be integrated into base problems fitting to the **Context** and **Problem**
7.1 General Overview	Overview of the domains involved in the PET
7.2 Assumptions	Assumptions on which the PET relies
7.3 Aspects	Description of the PET‘s cross-cutting functionality
7.4 Weaving	Explains how *Aspects* are integrated into the **Problem**
7.5 Base Problems	Not cross-cutting functionality introduced by the PET
8 Design Issues	Discussion of specific design and implementation details
9 Privacy Benefits	The PET's positive consequences on the **Privacy Forces**
10 General Benefits	The PET's positive consequences on the **General Forces**
11 Privacy Liabilities	The PET's negative consequences on the **Privacy Forces**
12 General Liabilities	The PET's negative consequences on the **General Forces**
13 Examples	Applications of the PET (e.g., on the **Motivation**)
14 Related Patterns	A list of patterns describing related PETs

the interfaces between them in the form of a context diagram. It is also important to document the *Assumptions* on which the functionality of the PET and its proposed privacy-enhancing properties rely. The PET's cross-cutting functionality shall be described as *Aspects* providing both a structural and a behavioral view. The *Weaving* explains how the aspects can be integrated into the base problems that are described in the pattern section **Problem**. The weaving shall combine the behavioral views of the base problems and the PET's aspects. Finally, a PET can introduce additional *Base Problems*, i.e., functionality that is not cross-cutting. These base problems are mostly concerned with the configuration, initialization, and maintenance of the PET. Examples are the configuration of an encryption or anonymization algorithm, and the management of cryptographic keys.

To describe the consequences of a PET, I distinguish, as usual, between positive consequences, called *benefits*, and negative consequences, called *liabilities*. I further differentiate between **Privacy Benefits/Liabilities** and **General Benefits/Liabilities**, as I also do for the forces. The documented consequences shall help requirements engineers to compare different PETs that fit to their system with each other and to finally select a PET that best fits to the system's privacy requirements, other non-functional requirements, and identified privacy risks that need to be mitigated.

16.3. PET Patterns

In this section, I present the two PET patterns *Privacy-ABCs* (see Section 16.3.1), and *Data Anonymization* (see Section 16.3.2) using the pattern format introduced in the previous section.

16.3.1. Privacy-ABCs

In this section, I present the PET Privacy-ABCs (Attribute-Based Credentials). As source for this PET pattern, I took the description from the ABC4Trust project (Camenisch et al., 2011). The rest of this section shows the PET Pattern for Privacy-ABCs.

1 Name Privacy-ABCs, Attribute-Based Credentials

2 Motivation A cigarette vending machine shall only provide cigarettes to adults. Hence, the machine has to check whether a customer is adult before cigarettes are provided to him or her. The cigarette vending machine shall not be able to gain more information about the customer or to learn that a certain customer already purchased cigarettes from it.

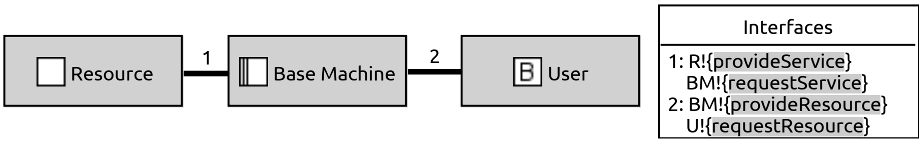


Figure 16.1.: *Context of Privacy-ABCs*

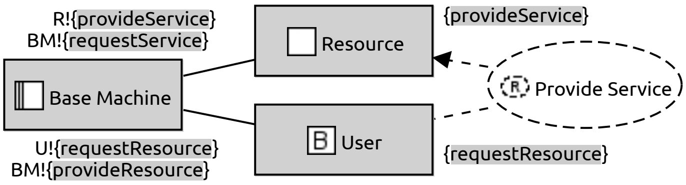


Figure 16.2.: *Basic structure of base problems addressed by Privacy-ABCs*

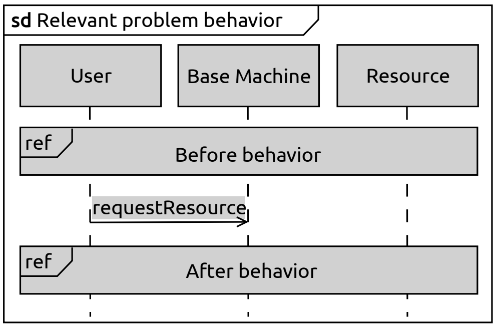


Figure 16.3.: *Behavior of base problems addressed by Privacy-ABCs*

3 Context A software shall be developed that processes personal data of its users in order to provide a service using an additional resource. It shall be ensured that certain personal data provided by the user are correct and authentic. That is, users shall not be able to input incorrect data about them. Figure 16.1 shows a context diagram that consists of the core elements of systems the PET shall be integrated into. The context diagram shows the User who can request the service of the Resource from the Base Machine that manages this Resource.

4 Problem A mechanism is needed to prove that a user's personal data have a certain property or to provide parts of the personal data while as little personal data as necessary are revealed to the software. Figure 16.2 shows the kind of base problems Privacy-ABCs might be integrated into. The problems have in common that a User requests a Resource's service from a Base Machine. The Base Machine processes the request and shall only provide under specific circumstances the requested service of the Resource to the User. Figure 16.3 shows the relevant behavior of the base problems. The behavior consists of an arbitrary Before behavior, the request of the User, and an arbitrary After behavior. The request is the relevant behavior because the Base Machine shall only execute the After behavior if the information provided with the request is authentic, correct, and satisfies certain properties, e.g., it contains a proof that the user's age is above 18.

5 Privacy Forces

5.1 Data Confidentiality: Only partial personal data or the proof that the personal data satisfy a certain property are needed. The actual personal data shall not be disclosed entirely or at all, respectively.

5.2 Integrity: The provided information shall be authentic and correct.

5.4 Anonymity: The service provider shall not be able to link the data collected during an interaction with the user to him or her.

5.5 Data unlinkability: The service provider shall not be able to link the data collected during an interaction with the user to the data collected during other interactions of him or her.

5.7 Pseudonymity: Transaction pseudonyms are needed. That is, for each interaction with a user, a new pseudonym is created that is neither linkable to the user nor to other actions of him or her (for details see Pfitzmann and Hansen (2010)).

5.8 Collection information: Users shall be informed about the personal data to be collected.

6 General Forces

6.1 End-user friendliness: The mechanism to check the authenticity and correctness of the user's request and the provided data shall not introduce much inappropriate effort that needs to be spent by users in comparison to the sensitivity of the personal data that are needed to provide the requested service.

6.2 Performance: The mechanism to check the authenticity and correctness of the user's request and his or her data shall not unnecessarily reduce the response time of the software-to-be or slow down the overall system.

6.3 Costs: The costs, also in the sense of effort, to implement, integrate, deploy, and maintain the PET shall be appropriate in comparison to its benefits.

6.4 Impact on functionality: The integration of a solution into the base problems shall not negatively influence other system functionality.

6.5 Abuse of PET: It shall not be possible to get access to the service by providing incorrect data.

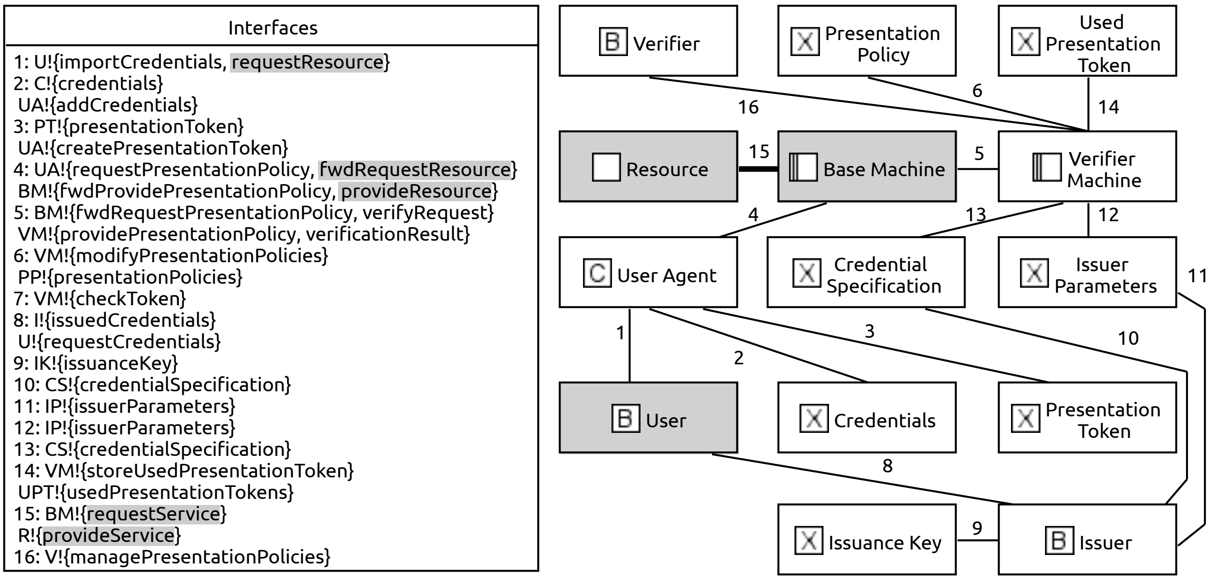
6.6 Revocation: In certain situations, e.g., abuse of the service, it may be wished to be able to reidentify the individual user who performed certain actions that led to that certain situation.

7 Solution

7.1 General Overview: Figure 16.4 shows the context diagram for a basic Privacy-ABCs system derived from Camenisch et al. (2011). The gray domains originate from the base problem Privacy-ABCs shall be integrated into, and the white domains are introduced by Privacy-ABCs. The machine that needs to be built is the Verifier Machine. This machine is operated by the Verifier (service provider) who is able to manage the Presentation Policy. The Presentation Policy describes which information a User has to disclose in order to get access to the Resource. To create a Presentation Policy, the Credential Specification and Issuer Parameters provided by an Issuer are used. The Issuer's task is to provide Credentials to Users and to ensure that these Credentials contain only valid information about the respective User. Which information can be stored in a Credential is defined in the Credential Specification. The Issuer Parameters specify how Presentation Tokens generated from User's Credentials can be verified to satisfy or to not satisfy certain properties. To generate Credentials, the Issuer uses his or her Issuance Key. The Verifier Machine represents the software part of Privacy-ABCs that needs to be integrated into the software-to-be. It receives requests from the Base Machine to provide the Presentation Policy and to check whether a User is allowed to access the Resource by verifying a provided Presentation Token using the Presentation Policy and the Credential Specification. The Verifier Machine may store these Used Presentation Tokens. Instead of receiving the requests directly from the User, the Base Machine receives the User's request from his or her User Agent. The User Agent manages the User's Credentials and generates on demand Presentation Tokens based on a Presentation Policy and the Credentials to request access to a Resource. A User can request Credentials from an Issuer and import these to his or her User Agent.

7.2 Assumptions: Some assumptions have to be considered for the issuing of credentials. The Issuer shall create only authentic and correct Credentials for Users using the Credential Specification and Issuance Key. The Issuer needs to be trusted by the User and the Verifier. Furthermore, it has to be assumed that the User will add the Credentials provided by the Issuer to his or her User Agent and is not able to modify them. For the generation of Presentation Tokens, it has to be assumed that a User's User Agent is able to properly generate Presentation Tokens for the Resource the User requests based on the User's Credentials and the Verifier's Presentation Policy. Furthermore, it has to be assumed that User Agent and Base Machine use an anonymous communication channel, e.g., using Tor[1]. Otherwise, it could be possible for the Base Machine to use meta-data, e.g., the User's IP address, to link Presentation Tokens to each other.

[1] https://www.torproject.org/ Accessed 21 Mar 2017


Interfaces

1: U!{importCredentials, requestResource}
2: C!{credentials}
UA!{addCredentials}
3: PT!{presentationToken}
UA!{createPresentationToken}
4: UA!{requestPresentationPolicy, fwdRequestResource}
BM!{fwdProvidePresentationPolicy, provideResource}
5: BM!{fwdRequestPresentationPolicy, verifyRequest}
VM!{providePresentationPolicy, verificationResult}
6: VM!{modifyPresentationPolicies}
PP!{presentationPolicies}
7: VM!{checkToken}
8: I!{issuedCredentials}
U!{requestCredentials}
9: IK!{issuanceKey}
10: CS!{credentialSpecification}
11: IP!{issuerParameters}
12: IP!{issuerParameters}
13: CS!{credentialSpecification}
14: VM!{storeUsedPresentationToken}
UPT!{usedPresentationTokens}
15: BM!{requestService}
R!{provideService}
16: V!{managePresentationPolicies}

Figure 16.4.: *Context diagram of Privacy-ABCs*

7.3 Aspects: Privacy-ABCs contain three aspects that need to be integrated into base problems which are concerned with requests of a User to access a Resource that shall be protected.

1. The Presentation Policy that specifies the information a Presentation Token shall contain to get access to the requested Resource needs to be provided to the User Agent. The aspect diagram for this cross-cutting concern is shown in Figure 16.5. The behavioral view to address the aspect is shown in Figure 16.6. The sequence diagram shows that if the User requests a resource via his or her User Agent, the User Agent first requests the Presentation Policy from the Base Machine. The Base Machine forwards the request to the Verifier Machine that retrieves the Presentation Policy and provides it to the Base Machine. Finally, the Base Machine provides the Presentation Policy to the User Agent, which then consequently received the presentation policy.

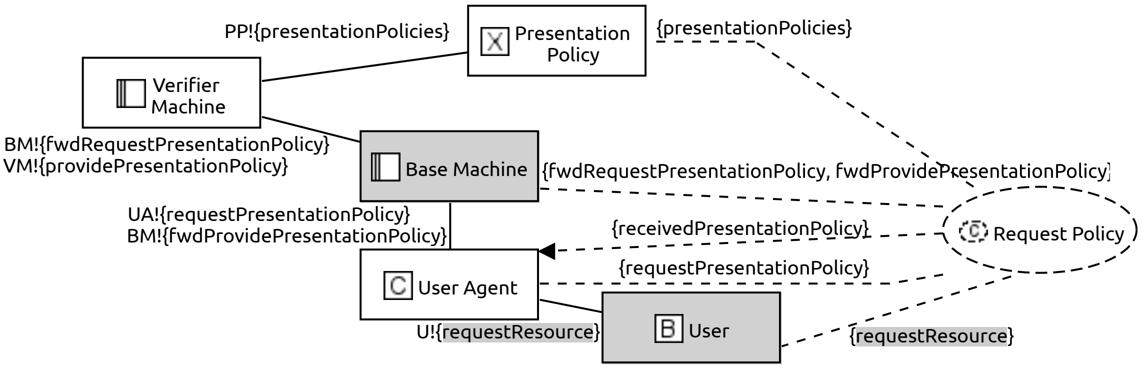

PP!{presentationPolicies}
X Presentation Policy
{presentationPolicies}
Verifier Machine
BM!{fwdRequestPresentationPolicy}
VM!{providePresentationPolicy}
Base Machine
{fwdRequestPresentationPolicy, fwdProvidePresentationPolicy}
UA!{requestPresentationPolicy}
BM!{fwdProvidePresentationPolicy}
{receivedPresentationPolicy}
C Request Policy
{requestPresentationPolicy}
C User Agent
U!{requestResource}
B User
{requestResource}

Figure 16.5.: *Aspect diagram for providing the presentation policy*

2. The Presentation Token provided by the User Agent needs to be verified to check whether the User is allowed to access the requested Resource using the Presentation Policy, Credential Specification, and Issuer Parameters. The result of the verification is sent to the Base Machine that then denies or provides access to the Resource for the User. For the sake of simplicity, I omit the corresponding aspect diagram. The necessary interaction between the domains to achieve the aspect is shown in Figure 16.8. The interaction can be started

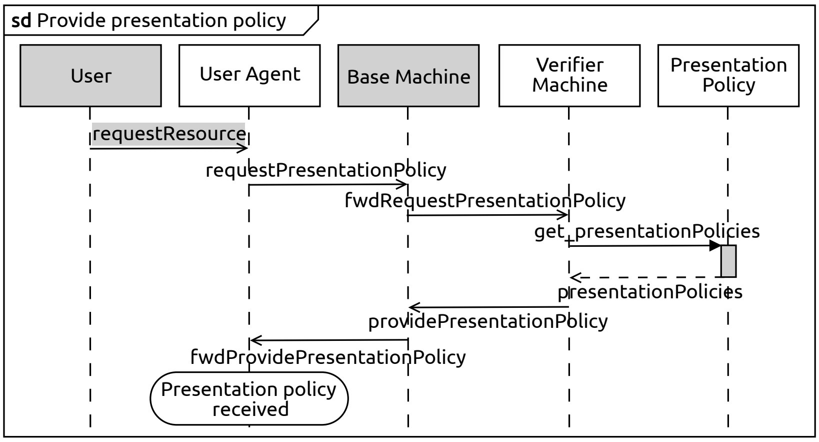

sd Provide presentation policy
User
User Agent
Base Machine
Verifier Machine
Presentation Policy
requestResource
requestPresentationPolicy
fwdRequestPresentationPolicy
get_presentationPolicies
presentationPolicies
providePresentationPolicy
fwdProvidePresentationPolicy
Presentation policy received

Figure 16.6.: *Sequence diagram for providing the presentation policy*

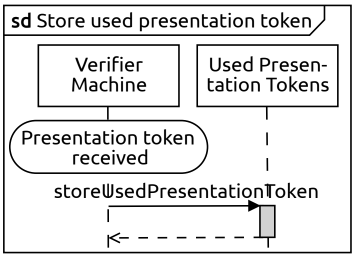

sd Store used presentation token
Verifier Machine
Used Presentation Tokens
Presentation token received
storeUsedPresentationToken

Figure 16.7.: *Sequence diagram for the storage of used presentation tokens*

if the User Agent has received the Presentation Policy. The User Agent then generates a respective presentation token (see Assumptions) for the user's request of a resource and requests the resource from the Base Machine using the generated Presentation Token. The Base Machine then asks the Verifier Machine to verify the received request. To do this, the Verifier Machine needs to retrieve the Presentation Policy, Credential Specification, and Issuer Parameters. The result of the verification is finally returned to the Base Machine.

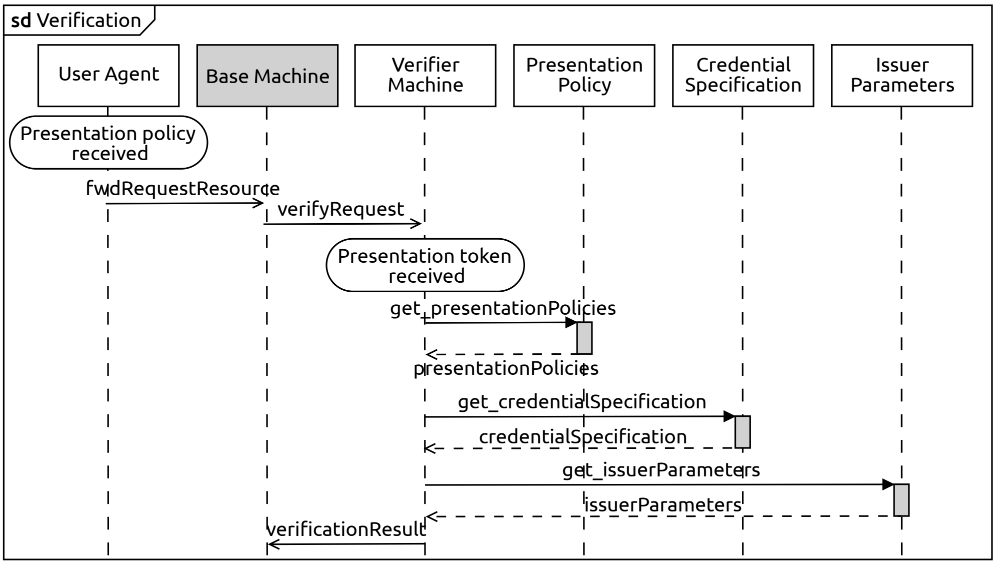


Figure 16.8.: *Sequence diagram for the verification of presentation tokens*

3. The Presentation Tokens used by Users to request access to a Resource may be stored, e.g., for statistical or maintenance reasons. I left out the corresponding aspect diagram for the sake of simplicity. Figure 16.7 provides the behavioral view on this aspect. It specifies that if the Verifier Machine received a presentation token, then the machine can store that token in the lexical domain Used Presentation Tokens.

7.4 Weaving: The sequence diagram shown in Figure 16.9 shows how the three aspects are weaved into the base problem (see Figure 16.3). First, the arbitrary Before behavior of the base problem takes place, and the User requests a Resource via his or her User Agent. The User Agent

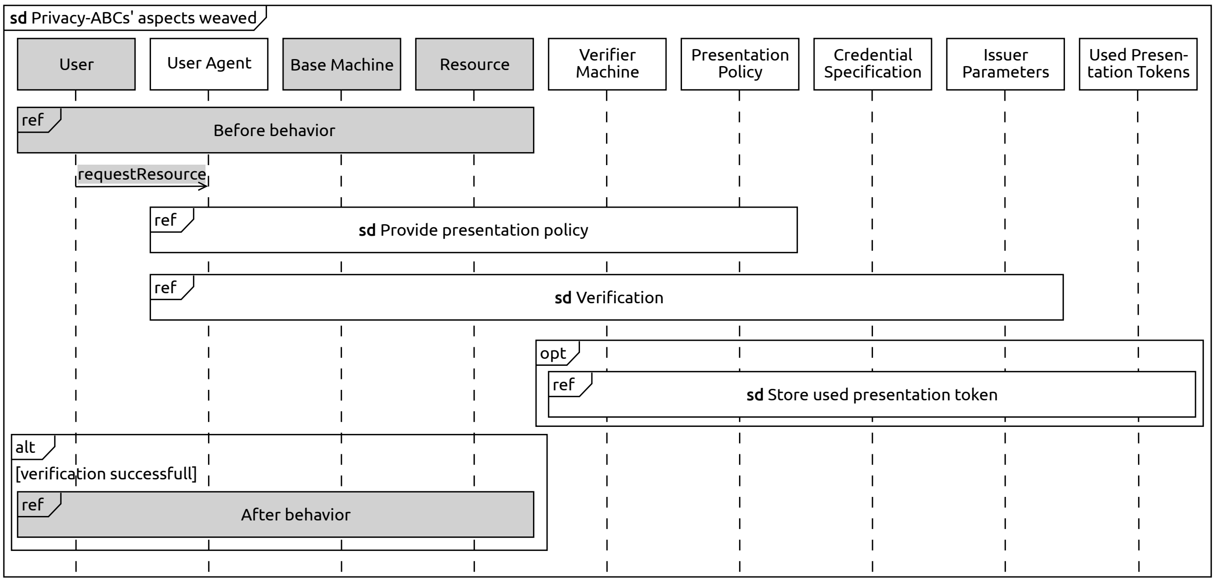


Figure 16.9.: *Weaving of Privacy-ABCs' aspects into base problems*

then requests the Presentation Policy (see Figure 16.6) to be able to generate the presentation token. Thereafter, the User Agent sends the generated presentation token that the Verifier Machine shall verify (see Figure 16.8). Optionally, the used presentation token can be stored by the Verifier Machine (see Figure 16.7). Iff the Verifier Machine reports a successful verification, the Base Machine executes the After behavior, i.e., the User gets access to the requested Resource.

7.5 Base Problems: Privacy-ABCs introduce an additional requirement that does not crosscut functionalities of the software-to-be. This is, the Verifier shall be able to specify his or her Presentation Policy that specifies which properties a User's Presentation Token must have to get access to a specific Resource. The Presentation Policy is based on the Credential Specification and the Issuer Parameters. The problem diagram for this additional base problem is shown in Figure 16.10. If it is expected that further Issuer Parameters and Credential Specification from other Issuers need to be added or that the Issuer changes these in the future, then similar base problems need to be introduced that are concerned with the management of the lexical domains Issuer Parameters and Credential Specification. For the sake of simplicity, I omit the behavioral views for these base problems.

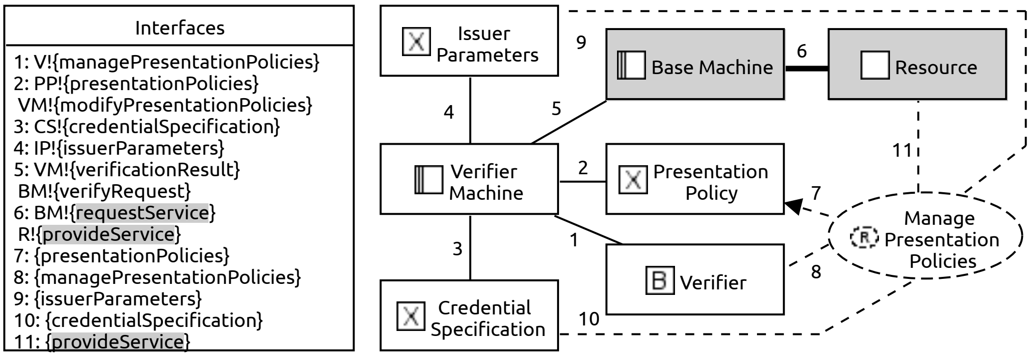

Interfaces
1: V!{managePresentationPolicies}
2: PP!{presentationPolicies}
VM!{modifyPresentationPolicies}
3: CS!{credentialSpecification}
4: IP!{issuerParameters}
5: VM!{verificationResult}
BM!{verifyRequest}
6: BM!{requestService}
R!{provideService}
7: {presentationPolicies}
8: {managePresentationPolicies}
9: {issuerParameters}
10: {credentialSpecification}
11: {provideService}
X Issuer Parameters
Base Machine
Resource
Verifier Machine
X Presentation Policy
B Verifier
X Credential Specification
R Manage Presentation Policies
1
2
3
4
5
6
7
8
9
10
11

Figure 16.10.: *Problem diagram for the management of presentation policies*

8 Design Issues If an existing Privacy-ABCs' infrastructure is used, there are not many design issues because most algorithms, protocols, and formats are prescribed. Only the presentation policy needs to be specified properly and the interface between the User and the Base Machine has to be refined with the User Agent (cf. Figure 16.4). If an own infrastructure shall be developed, several design decisions concerning algorithms, protocols, and formats have to be made. For the sake of simplicity, I omit the details on these issues and refer to Camenisch et al. (2011).

9 Privacy Benefits

9.1 Data Confidentiality: ABCs can be used to reveal personal data that shall be kept confidential only partially or to prove that the personal data satisfy a certain condition without revealing them. For example, it could be proved that a user is older than 18 without revealing his or her exact age or date of birth.

9.2 Integrity: Issuers guarantee that the credentials they issue contain only authentic and correct data (with respect to the date these where issued). It is cryptographically ensured that

1. no entities except the issuers can create credentials,
2. the credentials cannot be modified to contain other data, and
3. the presentation tokens created from a credential can contain only information from this credential or proofs about its properties.

9.4 Anonymity: Presentation tokens are not linkable to their user (unless attribute values or other data outside the scope of Privacy-ABCs allow linking).

9.5 Data unlinkability: Presentation tokens are unlinkable to each other (unless attribute values or other data outside the scope of Privacy-ABCs allow linking and if it has not been explicitly specified that pseudonyms are used to be able to link specific presentation tokens to each other).

9.7 Pseudonymity: Privacy-ABCs can be used to implement transaction pseudonyms for presentation tokens, i.e., a new pseudonym is created for each presentation token. The presentation policy can also specify that specific presentation tokens are linkable to each other if the issuer parameters allow that. Hence, it is possible to implement other kinds of pseudonyms, e.g., role pseudonyms (cf. Pfitzmann and Hansen (2010)).

9.8 Collection information: The service provider has to specify a presentation policy that is used to generate the user's presentation tokens. This policy specifies which information needs to be encoded into the presentation tokens. The presentation policy can be assessed by the user via his or her user agent before a respective token is created. Note that if the revealed attributes do not allow the verifier to link them back to the individual they belong to, then the elicited information is not considered as personal data and needs no further protection according to the EU General Data Protection Regulation (European Commission, 2016).

9.12 Data subject intervention: If the revealed attributes do not allow the verifier to link them back to the individual they belong to, then the verifier does not need to provide specific intervention options to users.

10 General Benefits

10.1 End-user friendliness: If an existing Privacy-ABCs infrastructure can be used and the potential users already have appropriate credentials, then users do not need to explicitly register to use the system and they do not have to input their personal data explicitly again. Users have to register only once at the issuer.

10.3 Costs: A Privacy-ABCs' infrastructure can be shared among several controllers that need to process the same or similar personal data, or an existing infrastructure provided by an identity provider may be used. For example, the German eID card can be used by authorized and certified controllers to check whether a user's age is below or above a specified value (Deutscher Bundestag, 2009).

10.4 Impact on functionality: It is possible that Privacy-ABCs replace another planned authentication mechanism and hence, make it unnecessary to manage user accounts and the like.

10.5 Abuse of PET: It is cryptographically ensured that corrupted tokens can be detected. Furthermore, the issuer guarantees that the data contained in the issued credentials are correct and belong to the user.

11 Privacy Liabilities

11.1 Data Confidentiality: The presentation policy specifies which information can be accessed by the verifier. It has to be specified in a way that only those personal data are revealed that are necessary to carry out the verifier's duties.

11.2 Integrity: Some personal data may change overtime, e.g., contact address. Hence, it may be necessary for users to request new credentials from an issuer and to invalidate the old credential. This issue is addressed by Privacy-ABCs with Revocation Authority (Camenisch et al., 2011).

11.4 Anonymity: The presentation policy specifies which information can be accessed by the verifier. If some provided information or other data outside the scope of Privacy-ABCs allow for linking, then anonymity may be broken.

11.5 Data unlinkability: The presentation policy specifies which information can be accessed by the verifier and whether the verifier is able to link presentation tokens to each other. The policy has to be specified in a way that presentation tokens can be linked to each other only if this is necessary.

11.7 Pseudonymity: The needed kind of pseudonym has to be specified in the verifier's presentation policy.

11.8 Collection information: The presentation policy specifies which personal data are collected, however, verifiers may still need to inform users about the purpose for which the revealed information is used.

11.12 Data subject intervention: Under specific circumstances, it may be necessary to integrate a mechanism that allows users to order the deletion of presentation tokens or to restrict the processing of them (cf. Article 11 of the EU General Data Protection Regulation European Commission (2016)).

12 General Liabilities

12.1 End-user friendliness: The usage of Privacy-ABCs has some issues concerning the end-user friendliness. First, users need to get credentials from an issuer that they need to trust. Second, users have to use a user agent for managing their credentials and generating presentation tokens. Hence, the perceived user-friendliness strongly depends on the properties of this user agent.

12.2 Performance: Depending on the complexity of the properties that need to be proved, the response time for the user could be higher than with a classical authentication mechanism.

12.3 Costs: The creation of an own Privacy-ABCs infrastructure, including issuing credentials and the development of user agents that generate presentations tokens, will be too expensive in most cases. If an existing infrastructure is used instead, it is possible that certain parts of the software need to be certified. Such a certification also raises costs.

12.4 Impact on functionality: It has to be ensured that the personal data necessary to provide the requested services are collected and that (if necessary) users' interactions can be linked to each other.

12.5 Abuse of PET: If the software-to-be can be misused, e.g., to commit a crime or to damage the service provider, it is hardly possible to identify the malicious user (cf. **9 Privacy Benefits**). This threat can be mitigated by the Privacy-ABCs variant with Inspector (Camenisch et al., 2011).

12.6 Revocation: The basic Privacy-ABCs implementation provides no revocation options, but there are two extensions that provide different revocation options. The first extension allows revocation of credentials. That is, once issued credentials can be made invalid by a revocation authority. The second extension introduces the role of an inspector. The inspector is able to reveal the exact personal data contained in a credential from a given presentation token or to uncover the individual to whom the presentation token belongs. The verifier shall only be allowed to request this inspection under specified circumstances that are also part of the verifier's presentation policy.

13 Examples When I apply Privacy-ABCs to the cigarette vending machine example, then the join point Base Machine (cf. Figure 16.2) is instantiated with the vending machine, the Resource with the cigarettes, and the User with the customer who wants to buy cigarettes. In Germany, the existing Privacy-ABCs infrastructure of the German eID card (Deutscher Bundestag, 2009) can be used. In this case, the Issuer (cf. Figure 16.4) is the German state and the User Agent is the eID card. The Credential contains information such as the customer's name, address, date and place of birth. The Presentation Policy of the vending machine specifies that the generated Presentation Token only needs to contain a proof that the customer is older than 18.

14 Related PET Patterns Privacy-ABCs with Revocation Authority, Privacy-ABCs with Inspector

16.3.2. Data Anonymization

In the following, I present the PET pattern Data Anonymization that can be used with different data anonymization schemes as those proposed by Sweeney (2002); Machanavajjhala et al. (2006).

1 Name Data Anonymization

2 Motivation Researchers want to use medical data managed by an electronic health system for clinical research, but it is not allowed to use the identifiable medical data for this purpose and it is infeasible to gain the consent of the data subjects. Hence, the data provided to researchers must not allow them to identify the data subjects the data belong to.

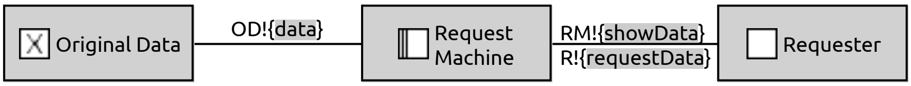


Figure 16.11.: *Context of Data Anonymization*

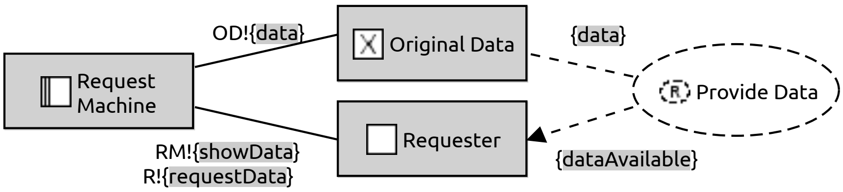


Figure 16.12.: *Basic structure of base problems addressed by Data Anonymization*

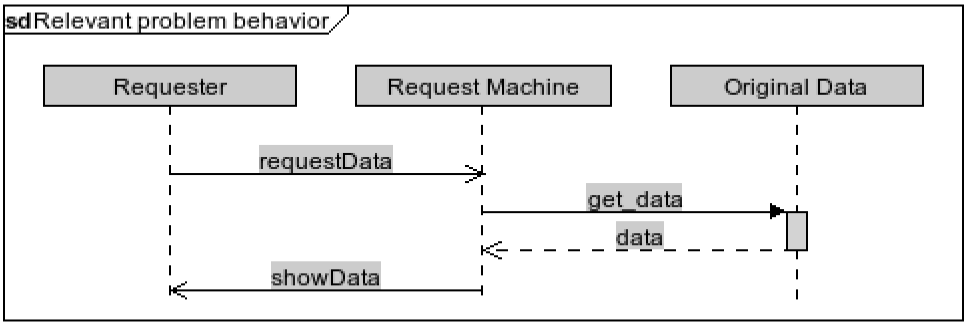


Figure 16.13.: *Behavior of base problems addressed by Data Anonymization*

3 Context A software shall be developed that processes personal data that were originally collected for another purpose. To allow the processing of these personal data for the new purpose, the personal data have to be anonymized, i.e., the data have to be transformed in a way that the identification of the data subjects to whom the personal data belong is not possible. Figure 16.11 shows a context diagram that consists of the core elements of systems the PET shall be integrated into. The context diagram shows the Requester that can request the data contained in the lexical domain Original Data, which were originally collected for a different purpose than the Requester needs the data for, from the Request Machine.

4 Problem A mechanism is needed to transform the originally collected personal data to anonymized data that no longer allow to identify the data subjects to whom the data originally belonged to. Figure 16.12 shows the kind of base problems Data Anonymization might be integrated into. The problems have in common that a Requester requests data managed by a lexical domain Original Data from a Request Machine. The Request Machine shall not return the personal data contained in the Original Data to the Requester, but only anonymized data. Figure 16.13 shows the relevant behavior of the base problems. The interaction starts with the data request of the Requester. The Request Machine then retrieves the data from the lexical domain Original Data, and provides them in an anonymized form to the Requester.

5 Privacy Forces

5.4 Anonymity: The data received by the requester shall not allow any counterstakeholder to whom these data are available to identify the data subjects to whom these data belong.

5.8 Collection information: The collected personal data shall only be used in their identifying form for the purpose they were originally collected for. The data subjects shall be informed about how their personal data are anonymized and further processed.

5.12 Data subject intervenability: The processing of the anonymized personal data shall be possible without the consent of the data subjects and without possibilities to intervene in the processing.

6 General Forces

6.1 End-user friendliness: The mechanism shall anonymize the data in a way that they are still useful for the end-users, e.g., the requester him- or herself.

6.2 Performance: The anonymization process is in most cases not time critical, but it has to be ensured that the anonymized data are provided to the requesters in a time frame in which the data are still useful for the requesters.

6.3 Costs: The costs, also in the sense of effort, to implement, integrate, deploy, and maintain the PET shall be appropriate in comparison to its benefits.

6.4 Impact on functionality: The integration of an anonymization mechanism into the base problems shall not negatively influence other system functionality.

6.6 Revocation: In certain situations, it may be wished to be able to reidentify the individual data subject to whom the anonymized personal data belong. For example, if specific health risks of a group of data subjects are identified during a clinical research project.

7 Solution

7.1 General Overview: Figure 16.14 shows the context diagram for a system implementing Data Anonymization. The gray domains originate from the base problem Data Anonymization shall be integrated into, and the white domains are introduced by the PET Data Anonymization. The machine that needs to be built is the Anonymizer. This machine is responsible for creating Anonymized Data from the Original Data that the Requester want to access through the Request Machine.

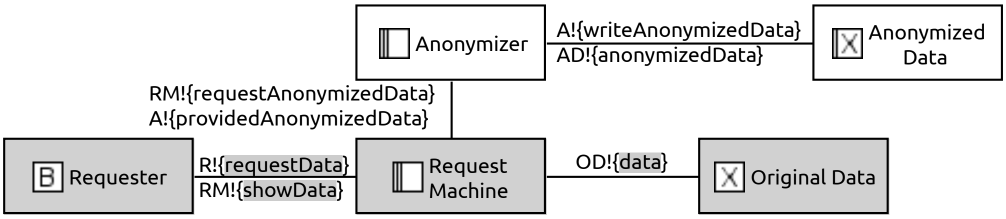


Figure 16.14.: *Context diagram of Data Anonymization*

7.2 Assumptions: There are no assumptions that need to be made for Data Anonymization.

7.3 Aspects: Data Anonymization contains only one aspect that needs to be integrated into base problems which are concerned with requests of anonymized data. The aspect diagram in Figure 16.15 shows the aspect Anonymize Data that refers to the Requester requesting data, and the Original Data to be anonymized. These requested data shall be anonymized by the Anonymizer and stored as such in the lexical domain Anonymized Data. Additionally, the Request Machine shall only provide the anonymized data to the Requester (cf. constrains dependencies in Figure 16.15). The behavioral view on this aspect including its pointcut scenario is given in Figure 16.16. After retrieving the data from the lexical domain Original Data in response to a Requester's request, the Request Machine asks the Anonymizer to anonymize these data. If that data were not anonymized before, or only older versions of the data were anonymized, then the Anonymizer needs to create the needed Anonymized Data. Then the Anonymizer provides the Anonymized Data to the Request Machine, which then can provide these to the Requester.

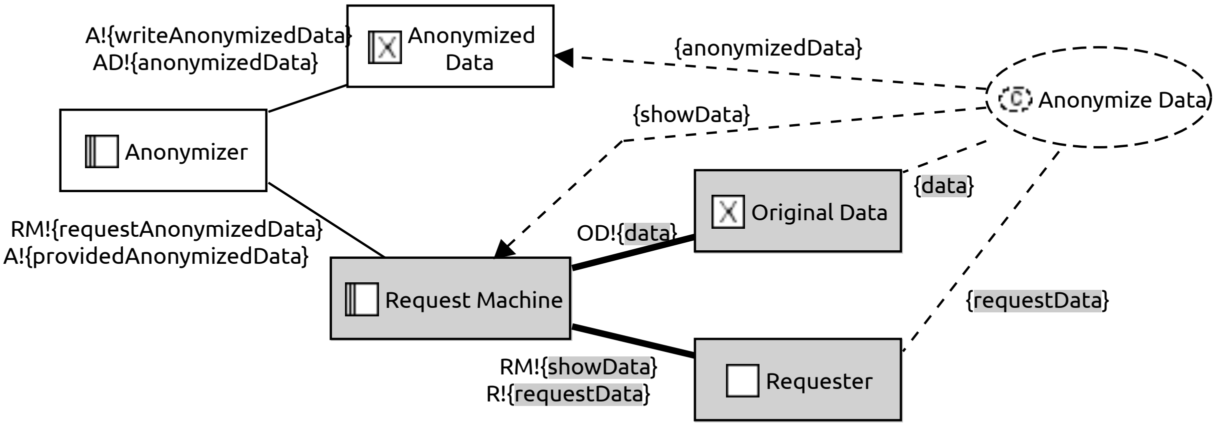


Figure 16.15.: *Aspect diagram for anonymizing data*

7.4 Weaving: The sequence diagram shown in Figure 16.16 describes already sufficiently how the anonymization aspect has to be integrated into the base problem. Namely, after retrieving the data to be anonymized and before the data are shown to the requester.

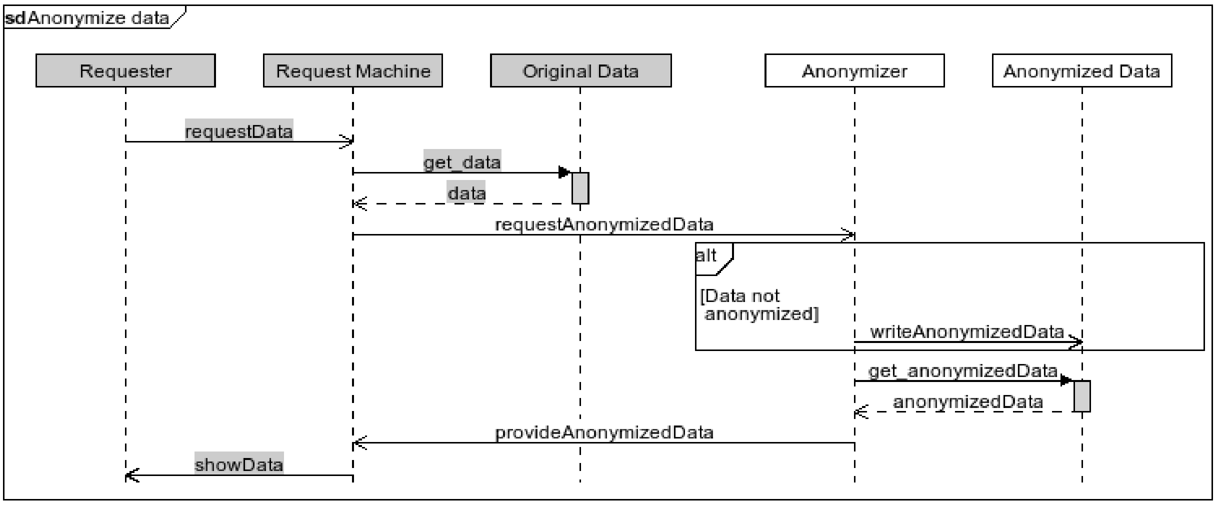


Figure 16.16.: *Sequence diagram for anonymizing data*

7.5 Base Problems: Data anonymization algorithms, e.g., k-anonymity (Sweeney, 2002) and l-diversity (Machanavajjhala et al., 2006) may be configured in their parameters k and l, and also by specifying the quasi-identifier, sensitive attributes, and which and how data fields are generalized or suppressed to obtain the needed degree of anonymization (see also **8 Design Issues**). However, I do not expect that this configuration needs to be changed (often) during the lifetime of the system, but only be configured once. Hence, I do not introduce this configuration as an additional base problem.

8 Design Issues The concrete anonymization algorithm and its configuration has to be selected carefully. For example, for k-anonymity (Sweeney, 2002) the quasi-identifier, the suppressed and generalized attributes of the quasi-identifier, and the value k have to be chosen. The quasi-identifier is the set of attributes that may allow to uniquely identify an individual in the data set. The techniques suppression and generalization are used to transform data belonging to the quasi-identifier in a way that these no longer allow the linkage to the data subject they belong to. Suppression means that parts of an attribute may be left out, e.g., the last two digits of a postal code are not shown. Generalization means, e.g., that a range of values is given instead of the concrete value. An example is to generalize the age of persons to intervals instead of using the exact age. The parameter k specifies how many entries in an anonymized data set (using suppression and generalization techniques on the data of the quasi-identifier) are allowed to have the same quasi-identifier values. Specifying a too small quasi-identifier or k may lead to an identification of data subjects to which the data belong. A too large quasi-identifier or k may make the anonymized data unuseful for the further processing of them, as the data then may lack information necessary for the further processing.

9 Privacy Benefits

9.4 Anonymity: The original data can be anonymized such that a counterstakeholder is not able to identify the data subject from the anonymized data.

9.8 Collection information: The processing of anonymized data is not constrained by privacy and data protection legislation, and hence, does not affect the collection information requirements of the system concerned with the original data. However, data subjects can still be informed about how their personal data are anonymized and further processed.

9.12 Data subject intervention: If only anonymized data are provided to requesters, then no intervention options need to be provided to data subjects.

10 General Benefits

10.1 End-user friendliness: By selecting appropriate suppression or generalization techniques, the anonymization mechanism can be adapted to the purpose for which the anonymized data shall be used. For example, the intervals for age generalization may be customized to the needs for the further processing.

10.2 Performance: The anonymization of the data may be performed offline. That is, the original data may only be anonymized once or in regular intervals, but not necessarily every time the anonymized data are needed.

10.3 Costs: The anonymization mechanism has in most cases only to be implemented once, possibly building on existing frameworks or implementations, and no or only low maintenance costs are expected.

10.4 Impact on functionality: Only low impact on the system's functionality is expected.

10.6 Revocation: In general, it is possible for the controller of the original data to reidentify the data subjects to whom the anonymized data belongs by matching the original data fields with the anonymized.

11 Privacy Liabilities

11.4 Anonymity: The anonymization mechanism has to be chosen carefully to ensure that the disclosed anonymized personal data do not allow to identify their data subjects (cf. **8 Design Issues**).

11.8 Collection information: The data subjects should be informed about the secondary uses of their anonymized data and how the data were anonymized.

12 General Liabilities

12.1 End-user friendliness: The suppression and generalization techniques of the anonymization mechanism have to chosen in a way that the anonymized data are still useful for the purpose they are needed.

12.3 Costs: The anonymization mechanism has to be designed carefully once in collaboration with a privacy expert, which may cause relatively high costs.

12.4 Impact on functionality: It has to be ensured that the anonymization process itself does not affect the system. For example, if the anonymization is performed offline, then it shall be performed when the system's workload is expected to be low.

12.6 Revocation: A reidentification of a data subject shall only be performed by the controller of the original data if this is in the interest of the data subject.

13 Examples Taking the example from **2 Motivation**, the join points are instantiated as follows. The Requesters are the researchers, the Original Data are the health records of patients, and the Request Machine is the machine of the electronic health system. The Anonymizer has then to be configured to the needs of the researchers who want to further process the health records in an anonymized form, while these anonymized data shall not allow to reidentify the data subjects they belong to.

14 Related PET Patterns Data Pseudonymization, k-Anonymity , l-Diversity

16.4. Discussion

Harrison (2006) states: *"The patterns community has been justly criticized for rehashing previously published material."* With this work, I want to emphasize that *"rehashing previously published material"* is necessary if the audience of the material is changed from researchers or developers of PETs to practical requirements engineers who want to apply PETs, and beneficial if the rehashing leads to a homogeneous representation of PETs which makes it easier to identify and compare different solutions for the same privacy requirements with each other. However, my work yet lacks evidence that the proposed presentation of PETs as patterns using an aspect-oriented notion really helps requirements engineers to address questions 1.-3. introduced in Section 16.1. In future work, it may be empirically evaluated how much requirements engineers benefit from a catalog of PET patterns. I also expect that such an evaluation would return valuable feedback from the participants of the experiments to further improve the pattern format, e.g., by adding further pattern (sub)sections to it, and to improve the presentation of PETs as aspects on the requirements level.

In addition to the question whether requirements engineers are willing to use a catalog of PET patterns, the question "who will provide the PET patterns and maintain such a catalog?" arises. I would prefer an open platform, similar to existing platforms for privacy patterns (cf. Section 16.5), to which people who have experience with a certain PET can add a respective PET pattern, and that can be browsed by everyone who is interested in the application of PETs.

In this chapter, I have shown how Privacy-ABCs and Data Anonymization can be presented as PET patterns supported by an aspect-oriented notation. I supervised a master (Gao, 2017) and two bachelor theses (Stöhr, 2018; Werger, 2018) in which the students created PET patterns for different privacy, transparency, and intervenability enhancing technologies. These initial results have shown that both simple and more complex PETs can be formulated as aspects, and that the proposed pattern format is suitable to represent them and their properties.

Actually, the proposed pattern format is independent of the problem frames notation and aspect-orientation. That is, any other notation or only plain text may be used to describe the **Context**, **Problem**, and **Solution**. But I believe that the provided context, problem, aspect, and sequence diagrams help to illustrate the **Context**, **Problem**, and **Solution** of a PET. Especially, the *Weaving* shall support requirements engineers to understand into which base problems of the system-to-be a PET needs to be integrated and how.

My proposed representation of PETs is dedicated to the requirements engineering phase, to support an early consideration of PETs and an integration of these into the other requirements of the system-to-be. Hence, my proposed pattern for Privacy-ABCs lacks information concerning concrete algorithms and other implementation details, e.g., in which format presentation tokens or policies are stored. This is intended, because the focus in the requirements engineering phase should be on understanding the problem of building the software-to-be rather than on implementation details (Jackson, 2000). Hence, the goal is to understand which additional entities (domains) have to be considered if a PET is selected, how these entities are related to each other and the software-to-be, how the software-to-be needs to be adapted to integrate the

PET, under which assumptions the PET functions, and with which benefits and liabilities the PET comes.

I created the aspects contained in the PET patterns using the aspect frames introduced in Chapter 15. The Provide presentation policy aspect (see Figure 16.6) does not fit to one of the aspect frames introduced in Chapter 15. This is, because it does not represent an instance of the Independent behavior aspect frame(cf. Section 15.3.4), because the behavior is necessary for the further functionality. Hence, an additional aspect frame, called *Necessary behavior aspect* may be identified from this aspect as an additional class of aspects. The Store used presentation token aspect (see Figure 16.7) is an instance of the Independent behavior aspect frame (see Section 15.3.4). This is, because it is concerned with adding an additional functionality, namely the storage of used presentation tokens, to the other functionality. The Verification aspect (see Figure 16.8) is an instance of the Decision aspect frame (see Section 15.3.1). This is, because the base machine requests from the verifier machine the verification of the request to decide whether the access to the requested resource may be granted or not. The information sources are the presentation policy, the credential specification, and the issuer parameters. The aspect Anonymize data (see Figure 16.16) is an instance of the Transform before transmission aspect (Section 15.3.3). This is, because the original data are transformed by anonymizing them before they are sent to the requesters.

16.5. Related Work

Hafiz (2013) presents a pattern language for developing PETs consisting of 12 patterns. Each pattern describes a solution to achieve a certain privacy property. The goal of the pattern language is to assist developers of PETs. Lobato et al. (2009) propose patterns that support the presentation of privacy policies to users. Schumacher (2003) presents two privacy patterns that describe best-practices for end-users to protect their privacy. Romanosky et al. (2006) identified three privacy patterns for online interactions. These contain patterns for best-practices for end-users and best-practices for the design of privacy-friendly software. Porekar et al. (2008) propose organizational privacy patterns. These privacy patterns shall help to address privacy issues already on the organizational level by providing corresponding solutions. The solution description is enhanced with Secure Tropos diagrams. Drozd (2016) developed a catalog containing nine privacy patterns that shall help to implement the privacy principles and guidelines proposed in ISO 29100 (ISO/IEC, 2011). In comparison to my proposed PET patterns, Drozd's patterns are very condensed. For example, Drozd does not make explicit how and when the patterns have to be integrated into the system, and under which assumptions this integration will be successful. Additionally, she only briefly describes the consequences using one or two sentences that only refer to positive effects of applying the pattern to the privacy principle the pattern belongs to. In addition to the previously mentioned works, there are two websites[2] that provide catalogs of privacy patterns similar to the mentioned works.

The above mentioned research mainly focuses on patterns to support the development of PETs, or on patterns that support data subjects to protect their privacy. In contrast to these works, I propose to express PETs themselves as patterns to support requirements engineers to select appropriate PETs and to integrate them into the software-to-be to address identified privacy requirements. The consideration of PETs as early aspects is a novel contribution of this chapter.

[2] `https://privacypatterns.org` (accessed on 20 June 2018) and `https://privacypatterns.eu` (accessed on 20 June 2018)

16.6. Conclusion

In this chapter, I have proposed a pattern format to represent PETs as early aspects. I have illustrated how a PET pattern shall look like using the rather complex PET Privacy-ABCs and the simpler PET Data Anonymization. Initial results have shown that a variety of PETs can be expressed as PET patterns. The PET patterns shall help requirements engineers to identify the PETs that address the privacy requirements that they need to integrate into the software-to-be, then to select the PET that best fits to the needs without having too much impact on the software-to-be, and finally to integrate the PET's requirements into the requirements of the software-to-be.

In future work, a larger catalog of PET patterns may be set up. Using this catalog, an empirical evaluation may be performed to assess how much requirements engineers benefit from PET patterns. This is, it may be investigated whether the catalog helps them to identify, select, and integrate PETs into a system. Additionally, it may be investigated whether the pattern format can also be used or adapted to document non-technical privacy measures and their properties. Furthermore, it could be studied how the effect of PETs to privacy threats and risks could be documented in the PET pattern format. Currently, the link between a privacy threat and the PETs that may be used to mitigate the threat is established by the type of privacy requirement which the privacy threat harms and which the PET has positive consequences on. An explicit consideration of privacy threats mitigated by a PET may support the analysis team to more effectively select privacy measures (cf. Section 17.3).

In the next chapter, I introduce the step Privacy Measure Integration of the ProPAn method. In this step, the analysis team has to select and integrate privacy measures that address the system's privacy requirements identified in the step Privacy Requirements Identification (see Chapter 12) and mitigate the unacceptable privacy risks identified in the step Privacy Risk Analysis (see Chapter 13) of the ProPAn method. The step Privacy Measure Integration is supported by the usage of PET patterns, as these support the analysis team to identify and select appropriate PETs as privacy measures, and to integrate these into the system to implement the system's privacy requirements and to mitigate its privacy risks.

Chapter 17

Selection and Integration of Privacy Measures into a Problem Frame Model

In this chapter, I introduce the step Privacy Measure Integration of the ProPAn method. In this step, the analysis team selects technical and non-technical privacy measures that implement the privacy requirements identified in the step Privacy Requirements Identification (see Chapter 12), and mitigate the privacy risks identified identified in the step Privacy Risk Analysis (see Chapter 13). Having selected the privacy measures, the analysis team has to integrate these into and align them with the functional requirements of the system, so that it can be reasoned that the integrated privacy measures satisfy the privacy requirements, also with respect to the identified privacy risks. I propose that the analysis team selects privacy measures based on PET patterns that I introduced in Chapter 16, which are based on the aspect-oriented problem frames notation introduced in Chapter 14.

The content of this chapter has not yet been published and hence, presents a novel contribution of my thesis.

In Section 17.1, I provide an introduction to this chapter. I sketch how the analysis team may prioritize privacy requirements and risks in Section 17.2. How the analysis team may select privacy measures is discussed in Section 17.3. I explain how the analysis team can integrate the selected privacy measures into the problem frame model in Section 17.4. How the analysis team can document why the taken privacy measures sufficiently implement the privacy requirements, also with respect to the identified privacy risks, is explained in Section 17.5. In Section 17.6, I compare the step Privacy Measure Integration of the ProPAn method with the state of the art methods. This chapter is concluded in Section 17.7.

17.1. Introduction

Having identified the privacy requirements that a system has to address and the threats that might lead to a violation of these, the analysis team has to select and integrate privacy measures that operationalize the privacy requirements or mitigate the identified risks implied by the threats to the privacy requirements. The selected technologies and non-technical measures that shall be integrated into the system may have severe consequences on the system, its functionality and properties, and the costs and effort to implement the system. Hence, the analysis team has to balance these factors during the selection of privacy measures.

The selected privacy measures only realize privacy requirements and mitigate privacy risks if they are correctly integrated into the functional requirements of the system. Hence, the analysis team has to specify how the measures are aligned with the system's functional requirements, and to provide arguments why this integration leads to a satisfaction of the privacy require-

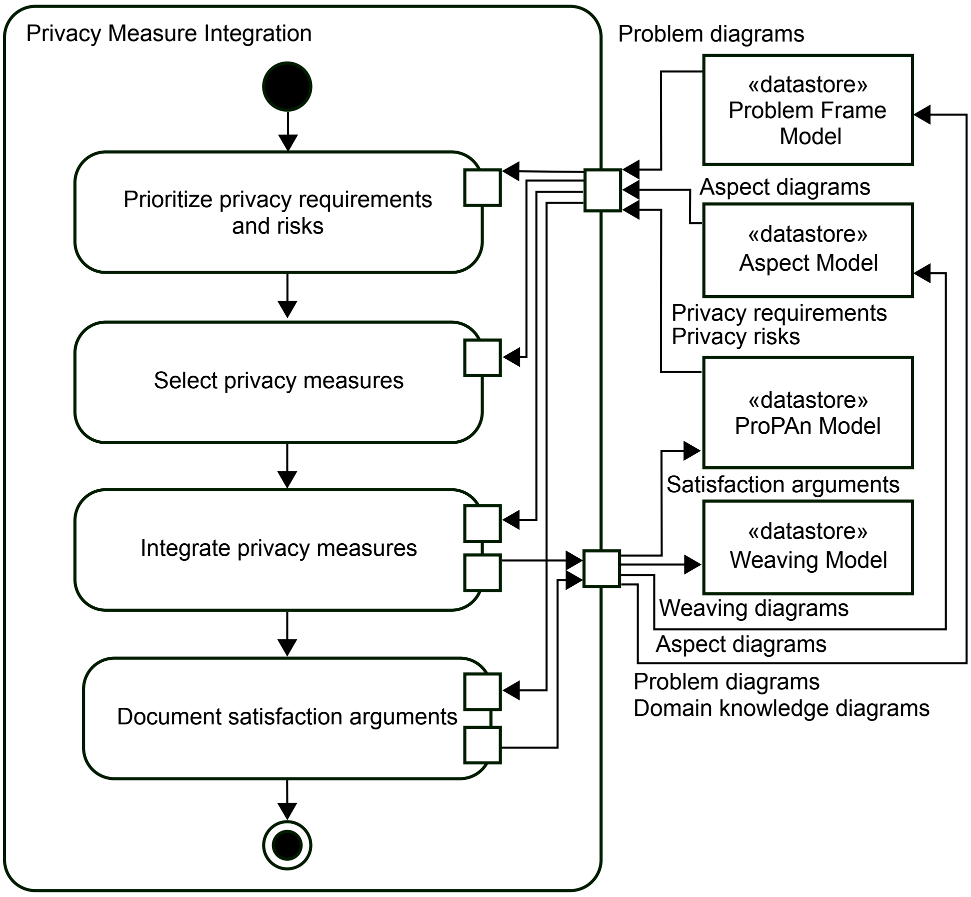

Privacy Measure Integration
Prioritize privacy requirements and risks
Select privacy measures
Integrate privacy measures
Document satisfaction arguments
Problem diagrams
«datastore» Problem Frame Model
Aspect diagrams
«datastore» Aspect Model
Privacy requirements
Privacy risks
«datastore» ProPAn Model
Satisfaction arguments
«datastore» Weaving Model
Weaving diagrams
Aspect diagrams
Problem diagrams
Domain knowledge diagrams

Figure 17.1.: *Detailed view on the step* Privacy Measure Integration *of the ProPAn method*

ments, also considering the risks that these might be violated. These satisfaction arguments are similar to the entailment relation relationship that Zave and Jackson (1997) propose to reason that the specification of the machine together with the domain knowledge about the machine's environment satisfies the functional requirements (see Section 2.1).

Figure 17.1 shows the substeps of the step Privacy Risk Analysis. In the first substep Prioritize privacy requirements and risks, the analysis team has to decide which privacy requirements and risks have to be addressed to which degree. This is, because privacy requirements may be addressed to different degrees and it may be unfeasible to address all of them to the highest level of satisfaction. For example, a data confidentiality requirement may specify that data shall not be disclosed to a third party. However, the data confidentiality requirement may still be consdidered as satisfied when the third party rarely gets accidentially access to single data records. To specify the degrees of satisfaction, I propose that the analysis team specifies the highest acceptable risk level for each privacy requirement considering the consequences scales for privacy requirements, the likelihood scales, and the risk matrix introduced in Chapter 13. A highest acceptable risk level for a privacy requirement implies the satisfaction criteria for implementing it. That is, the satisfaction criteria for a privacy requirement are the combinations of the consequence and likelihood values of the highest acceptable risks given by the risk matrix. Having prioritized the privacy requirements by assigning them their highest acceptable risk level, the analysis team has to select privacy measures to reach the needed satisfaction level of the privacy requirements and to mitigate the previously identified privacy risks in the step Select privacy measures. This step may be supported by a catalog of PET patterns (see Chapter 16) that describe the properties of PETs and their structure and behavior in the form of aspect diagrams. Note that the link between a privacy risk and a PET pattern that may be used to mitigate the risk is established by the type of privacy requirement which the privacy risk harms and which the PET has positive consequences on. The analysis team has then to integrate the selected measures into the problem frame model in the substep Integrate privacy measures. This is done by specifying weaving diagrams that describe how the selected aspects are weaved into the functional requirements of the system, and adding problem and domain knowledge diagrams to the problem frame model that specify not cross-cutting technical measures and non-technical measures, respectively. Finally, the analysis team has to reason why the selected and integrated privacy measures satisfy the previously specified satisfaction criteria by specifying a satisfaction argument in the step Document satisfaction arguments.

17.2. Prioritize Privacy Requirements and Risks

The input for the step Prioritize privacy requirements and risks are the privacy requirements and risks identified in the steps Privacy Requirements Identification (see Chapter 12) and Privacy Risk Analysis (see Chapter 13). To prioritize the privacy requirements, the analysis team has to specify the highest acceptable risk level for each privacy requirement of the system. That is, the analysis team decides which risk levels (introduced in Figure 13.8 on page 263) may be acceptable for a privacy requirement. For the selection of the risk levels, the analysis team shall consider for each privacy requirement the corresponding consequences scale, the likelihood scale, and the risk matrix specified in the step Privacy Risk Analysis (see Chapter 13). Privacy requirements for which acceptable, tolerable, and unacceptable risks are acceptable are considered to be important, medium prioritized, and unimportant, respectively. To document the highest acceptable risk level, I added the attribute highestRiskLevel to the class *PrivacyRequirement* (see Figure 17.2).

In this way, the satisfaction criteria for a privacy requirement are given by the likelihood and consequence scales (see Tables 13.8 and 13.9 on page 262), and the risk matrix (see Table 13.10 on page 264) defined in the step Privacy Risk Analysis (see Chapter 13). The risk matrix specifies which combinations of likelihood and consequence are acceptable, tolerable, or unacceptable. Consequently, the highest acceptable risk level of a privacy requirement specifies which most severe violations (given by the consequences) of the privacy requirement may occur and how often (given by the likelihoods) these may occur at most.

If no consequence scale was yet created by the analysis team for a privacy requirement, then the analysis team has to specify a consequence scale for the requirement in this substep. This can be the case if no threats to a class of privacy requirements were identified in the step Privacy Risk Analysis. The reuse of the previously specified likelihood and consequence scales, and the risk matrix lead to a consistent application of both refinement- and prevention-based privacy requirements operationalization (cf. Chapter 3). This is, because the satisfaction criteria of the privacy requirements and the privacy risks are formulated in the same terminology using the same scales.

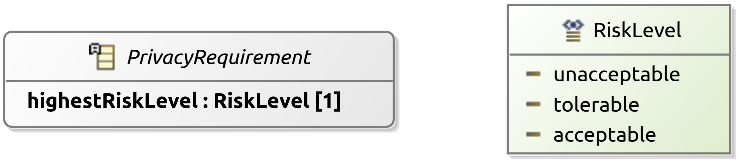
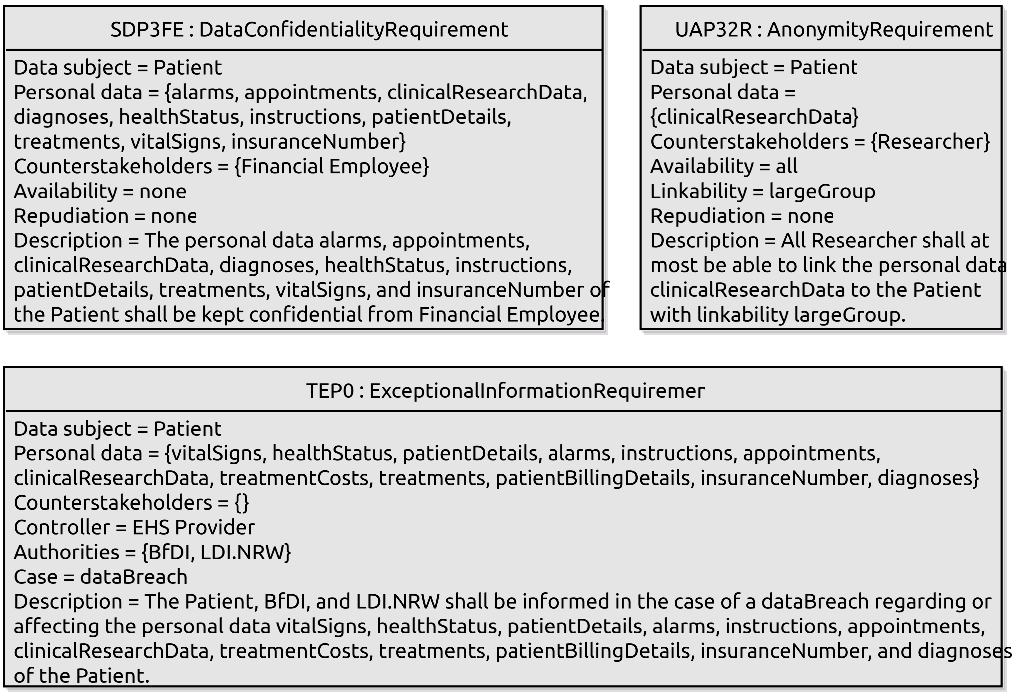


Figure 17.2.: *Adding an attribute to document highest acceptable risk level to the class PrivacyRequirement*



SDP3FE : DataConfidentialityRequirement

Data subject = Patient
Personal data = {alarms, appointments, clinicalResearchData, diagnoses, healthStatus, instructions, patientDetails, treatments, vitalSigns, insuranceNumber}
Counterstakeholders = {Financial Employee}
Availability = none
Repudiation = none
Description = The personal data alarms, appointments, clinicalResearchData, diagnoses, healthStatus, instructions, patientDetails, treatments, vitalSigns, and insuranceNumber of the Patient shall be kept confidential from Financial Employee.

UAP32R : AnonymityRequirement

Data subject = Patient
Personal data = {clinicalResearchData}
Counterstakeholders = {Researcher}
Availability = all
Linkability = largeGroup
Repudiation = none
Description = All Researcher shall at most be able to link the personal data clinicalResearchData to the Patient with linkability largeGroup.

TEP0 : ExceptionalInformationRequiremer

Data subject = Patient
Personal data = {vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, diagnoses}
Counterstakeholders = {}
Controller = EHS Provider
Authorities = {BfDI, LDI.NRW}
Case = dataBreach
Description = The Patient, BfDI, and LDI.NRW shall be informed in the case of a dataBreach regarding or affecting the personal data vitalSigns, healthStatus, patientDetails, alarms, instructions, appointments, clinicalResearchData, treatmentCosts, treatments, patientBillingDetails, insuranceNumber, and diagnoses of the Patient.

Figure 17.3.: *Privacy requirements of the EHS that are operationalized in this chapter*

Table 17.1.: *Satisfaction criteria for the selected privacy requirements of the EHS*

Requirement	Satisfaction Criteria
TEP0	Less than once per ten years the information about an occurred data breach is only available with manual effort. Less than once per two years the information about an occurred data breach is provided, but in an insufficient manner. Less than twice per year the information about an occurred data breach is provided in a sufficient manner, but not recognized.
SDP3FE	Less than once per ten years 21-100 records are disclosed to financial employees. Less than once per two years 5-20 records are disclosed to financial employees. Less than twice per year 1-4 records are disclosed to financial employees.
UAP32R	Less than once per ten years researchers can link 21-100 records to the corresponding patients. Less than once per two years researchers can link 5-20 records to the corresponding patients. Less than twice per year researchers can link 1-4 records to the corresponding patients.

Application to EHS As the EHS has to comply to the GDPR and processes sensitive personal data, all identified privacy requirements have as highest acceptable risk level the value acceptable. That is, no tolerable or unacceptable risks to the privacy requirements shall exist.

In this chapter, I only show the operationalization of the exceptional information requirement TEP0, the data confidentiality requirement SDP3FE, and the anonymity requirement UAP32R. These requirements (including their textual representation) are shown in Figure 17.3. I use the consequence scale for security-related privacy requirements for SDP3FE and UAP32R, and the transparency-related consequence scale for TEP0, both shown in Table 13.9 on page 262. Taking the likelihood scale shown in Table 13.8 on page 262 and the risk matrix shown in Table 13.10 on page 264, the satisfaction criteria listed in Table 17.1 can be derived. Note that I only list the three weakest satisfaction criteria that represent acceptable risks. These correspond to the risk matrix entries (**Rare**, **Moderate**), (**Unlikely**, **Minor**), and (**Possible**, **Insignificant**) in Table 13.10.

17.3. Select Privacy Measures

In the substep Select privacy measures, the analysis team has to decide how the privacy requirements shall be addressed in the system by integrating technical and non-technical privacy measures. A catalog of PET patterns (see Chapter 16) or other sources describing privacy enhancing technologies or non-technical measures may serve as an input for the analysis team. A catalog of PET patterns would be most preferable, as the PET patterns are designed to provide all the information about a PET as privacy measure that the analysis team needs to decide whether the PET shall be selected and how it can be integrated into the problem frame model. Note that the PET patterns yet do not explicitly contain information about whether and how a PET mitigates a privacy risk. However, the link between a privacy risk and a PET pattern that may be used to mitigate the risk is established by the type of privacy requirement which the privacy risk harms and which the PET has positive consequences on. Also other sources documenting privacy measures may be used by the analysis team. For example, Drozd (2016) provides a catalog of privacy patterns to address ISO 29100's privacy principles (ISO/IEC, 2011). Most of the privacy patterns contained in Drozd's catalog help to address transparency and intervenability requirements. As I derived ProPAn's transparency and intervenability requirements taxonomies also based on ISO 29100 (see Chapters 5 and 6), I expect that Drozd's catalog can also be used by the analysis team in this step. Additional catalogs of privacy patterns the analysis team may use are provided by two websites[1], and by other privacy requirements engineering methods (see Section 17.6). The international standard ISO 29151 (ISO/IEC, 2017b) lists several high-level controls for the protection of personal data with guidelines for their implementation. These controls can also be considered by the analysis team as a source of privacy measures. The EU General Data Protection Regulation (GDPR) (European Commission, 2016) also contains some references to possible privacy measures. For example, the GDPR suggests the use of standardized and machine-readable icons to inform data subjects about the processing of their personal data in Article 12, and the application of pseudonymization and encryption techniques to protect personal data in Article 32.

To select appropriate privacy measures, the analysis team needs to check in which context a privacy measure may be applied, and which problem and privacy requirements it addresses. This information is contained in the pattern sections **Context**, **Problem**, and **Forces** (cf. Chapter 16). If the privacy measure's context, problem, and forces fit to the system and its privacy requirements, the analysis team has to check whether it is feasible to integrate the technology into the system (using the pattern section **Solution**), and whether the provided solution is sufficient to address the system's privacy requirements (using the pattern sections **Privacy/General Benefits**), and whether the negative consequences (given in the pattern sections **Privacy/General Liabilities**) are acceptable and do not lead to a violation of other functional or quality requirements.

[1] `https://privacypatterns.org` (accessed on 20 June 2018) and `https://privacypatterns.eu` (accessed on 20 June 2018)

Application to EHS For the privacy requirement SDP3FE, I identified the risk that financial employees may deduce personal data that shall not be disclosed to them from the treatment costs and patient billing details that are available to them (see Figure 13.4 on page 259). I consider this risk as the only possibility that financial employees gain access to personal data of patient's that they are not allowed to access. As the unwanted incident leading to this risk also represents a data breach, the exceptional information requirement TEP0 about informing patients about data breaches may be violated, because the EHS provider is not able to detect such a disclosure to financial employees, and would hence be unable to inform the patients about this kind of data breaches. To reduce the likelihood of the unwanted incident that harms both SDP3FE and TEP0 (shown in Figure 13.4 on page 259 and in the following called FED), I decided to introduce an obligation to confidentiality and the performance of security trainings for financial employees as non-technical measure. This measure shall reduce the likelihood that a financial employee deduces additional personal data from the treatment costs and patient billing details available to him or her from *possible* to *rare* (cf. Figure 13.4 on page 259 and Figure 13.7 on page 261).

An additional risk evaluation with this non-technical measure shows that TEP0 is still not sufficiently addressed (see Figure 13.7 on page 261 and risk FED2_TEP0 in Table 13.10 on page 264), because the EHS provider may still not be able to know whether a data breach in the context of the financial application happened, and is consequently not able to inform affected patients about it. Hence, I introduce an obligation to monitor the occurrence of data breaches in the context of the financial application for the provider of it and to report these to the EHS provider. Data breaches can be detected, e.g., by monitoring a computer or network for suspicious usage of it. Knowing that a data breach happened, the EHS provider has to inform the patients about it. This could be realized using a technical measure, e.g., sending a message to the patient's mobile device, or a non-technical measure, e.g., the EHS provider sends a postal notification to the affected patient. As not all patients in the EHS necessarily use a mobile device, I decided to select the non-technical measure of a postal notification about occurred privacy breaches.

To ensure that the clinical research data provided to researchers only allow to link data records to large groups of patients (anonymity requirement UAP32R), the PET pattern Data Anonymization (see Section 16.3.2) is selected. This is, because it addresses the problem of anonymizing personal data, fits to the described context (cf. the problem diagram for R7 in Figure 4.8 on page 70), its solution may be integrated into the related subproblem for requirement R7, its benefits address the anonymity requirement UAP32R, and its liabilities are not expected to affect the other functional and privacy requirements.

17.4. Integrate Privacy Measures

The analysis team has to integrate the selected privacy measures into the system in the substep Integrate privacy measures. Non-technical privacy measures can be formulated as assumptions about the effected domains and can be represented in domain knowledge diagrams (see Section 2.2.4) that are added to the problem frame model describing the system-to-be. Technical privacy measures, possibly described as PET patterns, may consist of cross-cutting and not cross-cutting functional requirements, and assumptions that need to be guaranteed by the system in order to implement the privacy requirements. The cross-cutting functional requirements are described as aspects in aspect diagrams (see Section 14.3). To integrate these into the functional requirements of the system, weaving diagrams (see Section 14.4) have to be created that describe with which domains and phenomena of the problem diagrams the aspects' join points are instantiated, and consequently how the aspects are integrated into the functional requirements. The not cross-cutting functionalities are added as functional requirements represented by problem diagrams to the problem frame model. The assumptions needed for the technical

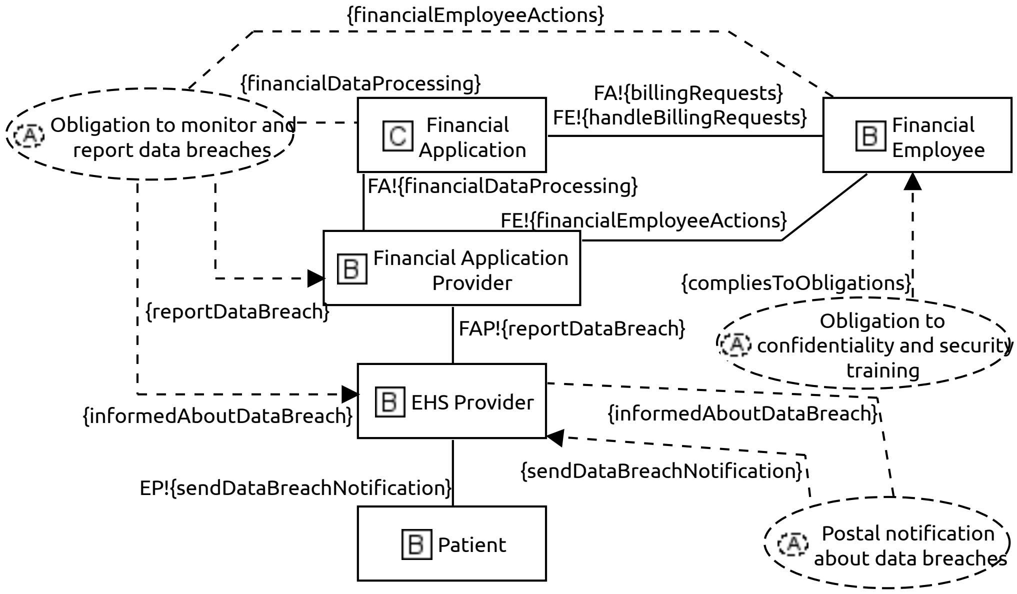

{financialEmployeeActions}
{financialDataProcessing}
A Obligation to monitor and report data breaches
C Financial Application
FA!{billingRequests}
FE!{handleBillingRequests}
B Financial Employee
FA!{financialDataProcessing}
FE!{financialEmployeeActions}
B Financial Application Provider
{compliesToObligations}
{reportDataBreach}
FAP!{reportDataBreach}
A Obligation to confidentiality and security training
B EHS Provider
{informedAboutDataBreach}
{informedAboutDataBreach}
{sendDataBreachNotification}
EP!{sendDataBreachNotification}
B Patient
A Postal notification about data breaches

Figure 17.4.: *Non-technical privacy measures to address the privacy requirements* SDP3FE *and* TEP0 *represented as assumptions*

privacy measures are added as domain knowledge diagrams to the problem frame model, just as the assumptions which represent the non-technical privacy measures.

Note that the analysis team can now add the selected privacy measures to the respective collection, storage, and flow information requirements (see attribute measures in Figure 7.6 on page 112). In this way, the need to inform the data subjects about the technical measures integrated into the system is documented.

Application to EHS The non-technical measures to address the data confidentiality requirement SDP3FE and the exceptional information requirement TEP0 are integrated into the problem frame model as three assumptions represented in the domain knowledge diagram shown in Figure 17.4. The assumption Obligation to confidentiality and security training constrains the FinancialEmployee to comply to the obligation and hence to not deduce further personal data from the information available to him or her. The assumption Obligation to monitor and report data breaches refers to the actions of the Financial Employee and the data processing of the Financial Application, which are both observable by the Financial Application Provider. The assumption further constrains the Financial Application Provider to report occurred data breaches to the EHS Provider, who is consequently constrained to be informed about the reported data breaches. Finally, the assumption Postal notification about data breaches constrains the EHS Provider to send data breach notifications to the Patient when the EHS Provider is informed about a data breach concerning the respective Patient.

The aspect Anonymize data of the PET pattern Data anonymization (see Section 16.3.2), has to be integrated into the problem diagram for functional requirement R7 shown in Figure 4.8 on page 70. The join point instantiation for this aspect and problem diagram is shown in Table 17.2. Based on this join point instantiation, the sequence diagram shown in Figure 17.5 can be obtained that describes how the aspect has to be integrated into the behavior to address R7. As specified by the sequence diagram shown in Figure 16.16 on page 322, the anonymization is integrated after the retrieval of the health records that shall be anonymized, and before the requested data are sent to the Research Database Application. The concrete anonymization algorithm has to be specified in collaboration with the researchers who are interested in the clinical research data. This has to be done to ensure that the clinical research data are useful for the researchers, but

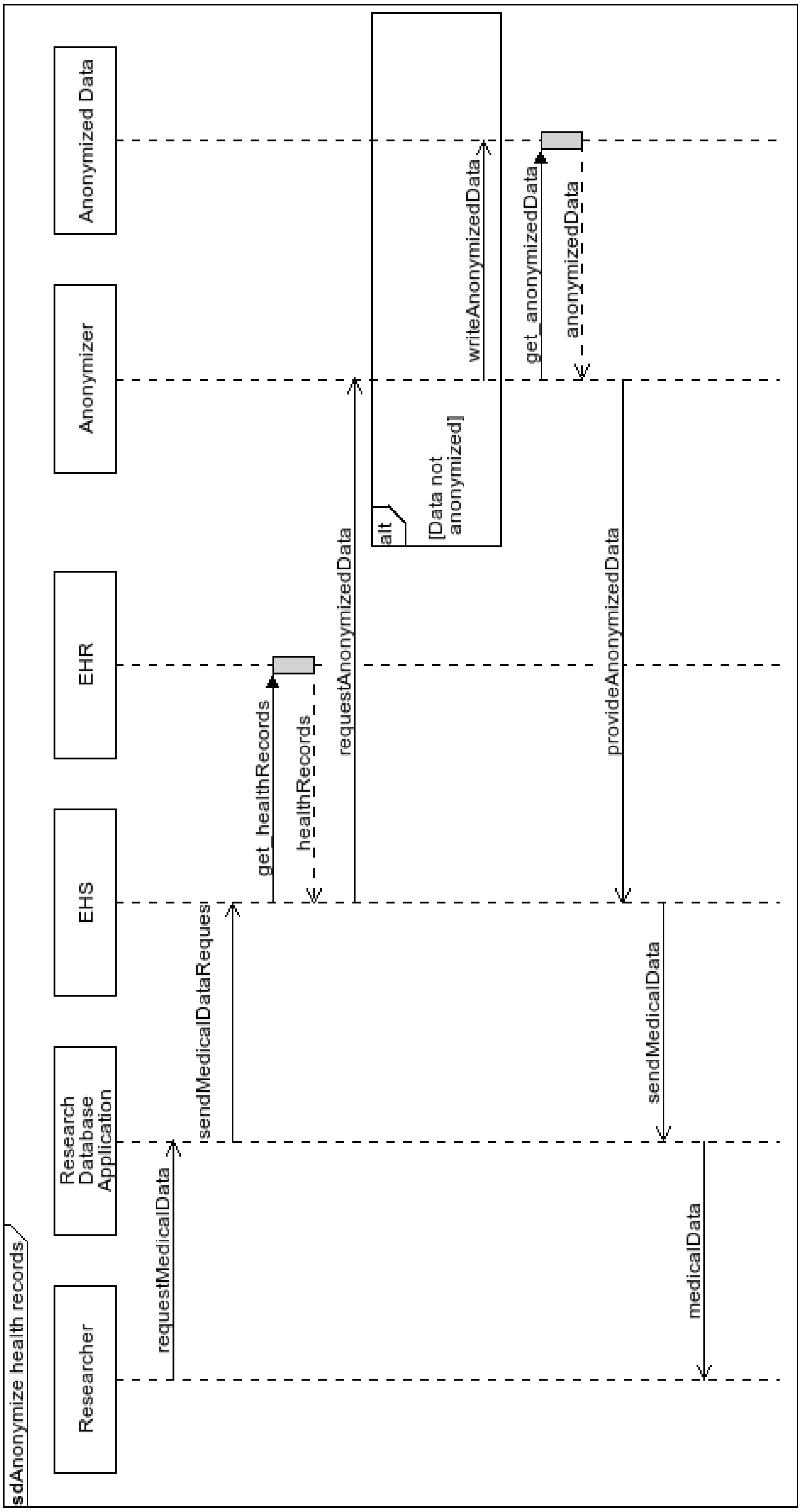


Figure 17.5.: *The* Anonymize data *aspect weaved into the behavior to satisfy R7*

Table 17.2.: *Join point instantiations for the integration of the* Anonymize data *aspect into R7 of the EHS*

Join Point	Instantiation
Requester	Research Database Application
Request Machine	EHS
Original Data	EHR
data	healthRecords
requestData	sendMedicalDataRequest
showData	sendMedicalData

still do not allow to reidentify patients. This specification needs a concrete data model of the health records and also depends on the amount of health records managed by the EHS. However, these considerations are mainly part of the design phase of a software development process and hence, out of the scope of my thesis.

17.5. Document Satisfaction Arguments

Having selected and integrated privacy measures into the system, the analysis team has to reason why these measures sufficiently implement of the privacy requirements, also in relation to the documented privacy risks. For this, I propose to use satisfaction arguments similar to the entailment relationship proposed by Zave and Jackson (1997) (see also Section 2.1) to reason that the specification of the machine S, and the domain knowledge on the machine's environment K suffice to realize the system's requirements R, which is denoted as $S, K \vdash R$. In my case, it has to be reasoned that the privacy requirement P is entailed by, and the privacy risks U are mitigated by the non-technical privacy measures N, and the technical privacy measures T that complement and are integrated into the system's requirements R and the domain knowledge K. This can be denoted as $N, T, R, K \vdash P, U$.

Figure 17.6 shows the metamodel for the documentation of the satisfaction argument. The class SatisfactionArgument is a subclass of *DocumentableElement*, and has hence a name and description. The description shall contain the actual argument, why the statements referenced as measures that are integrated into the functional requiremens and domain knowledge of the system-to-be referenced as statements mitigate the referenced risks (represented by unwanted incidents), and implement the privacy requirement referenced as clause. The privacy requirement contains the satisfaction argument, because the a satisfaction argument belongs only to one specific privacy requirement. The measures of the satisfaction argument shall reference all non-technical and technical privacy measures, and the reference statements all functional requirements and domain knowledge of the system that are relevant to implement the respective privacy requirement. As risks, all unwanted incidents that harm the privacy requirement shall be linked to the satisfaction argument, and the statements should include all statements from which the risks where identified. This is specified by the OCL invariant shown in Listing 17.1.

The analysis team shall create for each privacy requirement exactly one satisfaction argument that references all unwanted incidents that harm the privacy requirement. The statements shall

Listing 17.1: *OCL invariant for the class* SatisfactionArgument

```
context SatisfactionArgument
inv: self.risks = self.clause.harmedBy.source and
     self.statements→includesAll(self.risks.origin)
```

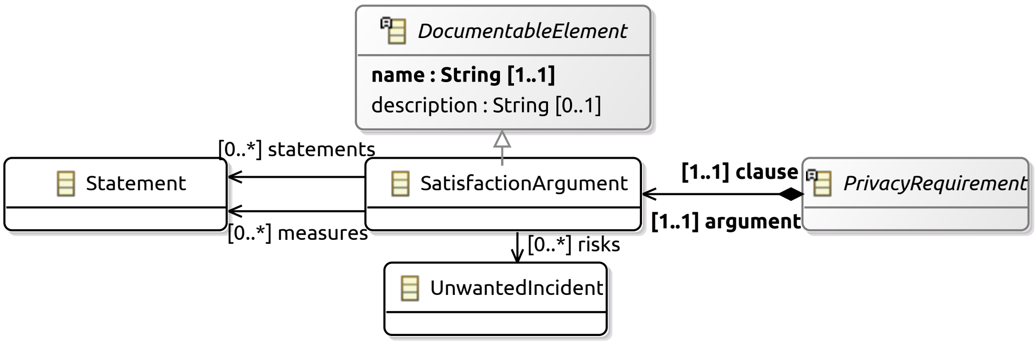


Figure 17.6.: *Part of the ProPAn metamodel that introduces the class* SatisfactionArgument

Listing 17.2: *OCL invariant for the class* Cures

```
context Cures
inv: SatisfactionArgument.allInstances()→exists(sa|
  sa.measures→includes(cures.statement) and
  Set{cures.cause}→closure(c|c.causes.target)→intersection(
    sa.risks)→notEmpty())
```

reference all statements about the system-to-be and the measures all privacy measures that are needed to reason that all privacy risks are mitigated and the privacy requirement is satisfied. The description of the satisfaction argument shall document the reason why the measures lead to the mitigation of the risks and the satisfaction of the clause when these are integrated into the statements.

The documentation of the satisfaction arguments allows to document the decisions of the analysis team and hence, to later evaluate the measures taken by the analysis team to address the system's privacy requirements and risks to these. This may be necessary or valuable, e.g., for privacy audits and certifications.

After the privacy measure integration, the analysis team has to perform the step Privacy Risk Analysis (see Chapter 13) again (cf. Chapter 8). During this repeated application of the Privacy Risk Analysis, the analysis team evaluates whether the selected privacy measures may introduce privacy risks themselves, by assessing deviations of their expected behavior. Additionally, the analysis team shall add the selected privacy measures to the threat diagrams to document which privacy measures mitigate the privacy risks and to assess to which degree these reduce the privacy risks.

The satisfaction arguments provide support for the analysis team for the latter task, as they link the privacy measures to the unwanted incidents they mitigate. The OCL invariant on the class Cures (cf. Figure 13.6 on page 260) given in Listing 17.2 specifies that the documentation of the privacy measures in the threat diagrams shall be consistent to the satisfaction arguments. That is, for each cures relation, a satisfaction argument has to exist that explains why the statement referenced by a cures relation mitigates the referenced cause. The referenced cause should be on a causes chain (see Section 13.3) to an unwanted incident that is referenced as risk by the respective satisfaction argument.

When no further unacceptable privacy risks are identified, and all satisfaction arguments are approved by the analysis team, the privacy analysis is finished and an initial privacy impact assessment (PIA) report can be created based on the information documented in the ProPAn model in the step PIA Report Creation (see Chapter 18). Otherwise, the step Privacy Measure Integration described in this chapter has to be performed again to select or adapt privacy measures that mitigate the unacceptable privacy risks.

Application to EHS The satisfaction arguments for the considered privacy requirements TEP0, SDP3FE, and UAP32R are listed in Table 17.3.

17.6. Comparison to the State of the Art

In this section, I discuss briefly the state of the art privacy requirements engineering methods that I introduced in Chapter 3 and that support the operationalization of privacy requirements.

Many methods (Kung et al., 2011; Degeling et al., 2016; Lentzsch et al., 2017; Yee, 2017; Murukannaiah et al., 2016; Islam et al., 2010; Oliver, 2016; Feth et al., 2017; Gürses et al., 2011; Vicini et al., 2016; Drgon et al., 2016) contain a privacy measure selection step and describe how these could be documented. Hong et al. (2004); Murukannaiah et al. (2016); Senarath et al. (2017) propose further support by providing elicitation questions to identify relevant privacy measures.

The methods proposed by Dennedy et al. (2014); Colesky et al. (2016); Antón and He (2003); Fisk et al. (2015); Bellotti and Sellen (1993); Deng et al. (2011); Oetzel and Spiekermann (2014); van Blarkom et al. (2003); Crespo et al. (2015); Spiekermann and Cranor (2009); Kalloniatis et al. (2008) propose catalogs of privacy measures that can be used by the analysis team to operationalize privacy requirements or mitigate privacy risks. These measures are in most cases linked to specific privacy requirements, principles, or threats. In comparison to the PET patterns (see Chapter 16) that I propose to document privacy enhancing technologies, the catalogs of the state of the art methods lack details on the privacy measures that are needed to decide which measures can be integrated into the system, to which degree the measure satisfies the system's privacy requirements or treats the system's privacy risks, and which other effects the integration of the measure may have on the system. However, it could be further investigated whether the privacy measures proposed by the other state of the art methods can be captured in the form of PET patterns in future work.

Yu and Cysneiros (2002); Liu et al. (2003); Jensen et al. (2005) integrate privacy measures as tasks into the system that is represented as a goal model. In comparison to my work that is based on the problem frames approach (Jackson, 2000), the goal models are on a higher abstraction level and do not provide as much detail about the entities (machines and domains) and their relations (interfaces and phenomena) as problem frame models do. However, a combination of these approaches with the ProPAn method might be investigated in future work.

Kalloniatis et al. (2008); Diamantopoulou et al. (2017); Argyropoulos et al. (2016) propose the integration of privacy measures in the form of process patterns that can be integrated into the business processes of the system. Their process patterns describe high-level privacy enhancing technologies and could serve as categories for a hierarchical catalog of PET patterns. The process patterns can also be seen as very high-level aspects that are weaved into the business processes of the system, which also indicates that these may be combined with my method for the aspect-oriented integration of privacy measures into a system.

Guarda et al. (2017); Ranise and Siswantoro (2017); Ahmadian and Jürjens (2016) propose to model the system using message sequence charts and UML models that are annotated with privacy relevant information. Privacy requirements and measures are formulated as privacy policies, e.g., in the form of attribute-based access control rules, in first order logic. These policies can then be checked for consistency to the annotated system model. A similar automatic consistency checking is also integrated into the ProPAn tool, as it can check the validity of satisfaction arguments, and the consistency between the satisfaction arguments and the privacy measures added to the threat diagrams.

I can conclude that there are several methods that provide larger catalogs of privacy measures than I do in my thesis. However, the pattern format for PET patterns proposed in my thesis specifies which information is needed by the analysis team to select and integrate privacy

Table 17.3.: *Satisfaction arguments for the selected privacy requirements of the EHS*

Clause	Risks	St.	Measures	Argument
TEP0	FED	R3, R4, A6	Obligation to confidentiality and security training, Obligation to monitor and report data breaches, Postal notification about data breaches	Violations to TEP0 may occur when the EHS provider is not able to recognize data breaches and hence, not able to inform data subjects about these. This risk is mitigated by introducing an Obligation to confidentiality and security training for financial employees who could be a source of data breaches as personal data of patients may flow to them due to R3, R4, and A6. Additionally, an Obligation to monitor and report data breaches for the provider of the financial employee, who is hence responsible to monitor his or her system and employees, and to inform the EHS provider about occurred data breaches. This monitoring is also expected to reduce the likelihood that a financial employee gains more personal data than intended and discloses these to others. The EHS provider shall provide a Postal notification about data breaches. In this way, it is ensured that patients are informed about breaches of their personal data in a sufficient manner.
SDP3FE	FED	R3, R4, A6	Obligation to confidentiality and security training, Obligation to monitor and report data breaches	Violations to SDP3FE may occur when the the financial employees are able to deduce additional personal data from the personal data treatment costs and patient billing details available to them due to R3, R4 and A6 and documented by the unwanted incident FED. This risk is mitigated by introducing a Obligation to confidentiality and security training for financial employees that shall ensure that the financial employees not maliciously deduce additional personal data, e.g., diagnoses or treatments received by patients. Additionally, an Obligation to monitor and report data breaches for the provider of the financial employee, who is hence responsible to monitor his or her system and employees. This monitoring is also expected to reduce the likelihood that a financial employee maliciously derives personal data that is not intended to be derived by them. In this way, it is ensured that financial employees only get to know the personal data treatment costs and patient billing details.
UAP32R	-	R7	Anonymize Data	To ensure that the clinical research data provided to researchers via the research data application derived from the health records (R7) do only allow the researchers to link a data record to a large group of patients it may belong to, the data are anonymized using an appropriate anonymization algorithm that provides the needed degree of linkability between the clinical research data and the related patients.

measures. This information is only partially available in the catalogs of the other methods. I expect that the privacy measures proposed by the other state of the art methods can also be presented as PET patterns. My proposed integration of the selected privacy measures into the system as domain knowledge, functional requirements, and aspects is a novel contribution to the state of the art, as it provides a more detailed system and requirements model than goal models do. Furthermore, no other privacy requirements engineering method make the reasoning why the selected privacy measures suffice to mitigate the privacy risks and realize the privacy requirements explicit.

17.7. Conclusions

In this chapter, I have explained the substeps of the step Privacy Measure Selection. For the first substep, I described how the analysis team may prioritize the identified privacy requirements and risks by defining which risk level is acceptable for each privacy requirement. In this way, I also obtain satisfaction criteria that can be used for a refinement-based operationalization of privacy requirements using the same terminology that is used for the risk analysis. This is, because I propose to use of the same likelihood and consequence scales. Hence, the ProPAn method supports both the refinement-based and prevention-based operationalization of privacy requirements (cf. Chapter 3) and synchronizes these by using the same terminology. I further explained how the analysis team may select privacy measures that mitigate the privacy risks identified in step Privacy Risk Analysis (see Chapter 13) and operationalize the privacy requirements identified in the step Privacy Requirements Identification (see Chapter 12), possibly supported by a catalog of PET patterns (see Chapter 16). Thereafter, I explained how the selected privacy measures may be integrated into the problem frame model representing the system-to-be as part of the third substep Integrate privacy measures. This integration is supported by the description of the weaving in the PET patterns. Finally, I propose to document satisfaction arguments that provide an (informal) reasoning why the system's privacy requirements are sufficiently operationalized by the selected and integrated privacy measures, also with respect to the identified privacy risks.

In future work, it could be further assessed how privacy requirements can be prioritized considering other quality requirements, the costs and effort to implement these into the system, and potentially other relevant factors. This prioritization may be based on existing prioritization techniques, e.g., the MoSCoW method (Clegg and Barker, 1994) that categorizes requirements into the categories *Must have*, *Should have*, *Could have*, and *Won't have (this time).* Furthermore, the selection process for privacy measures may be further supported by using optimization models considering the above mentioned factors. Pavlidis et al. (2017) propose such an approach for the selection of security measures in Secure Tropos.

When no further risks are identified after an additional application of the step Privacy Risk Analysis (see Chapter 13) and all privacy requirements are considered as sufficiently refined, then the privacy analysis part of the ProPAn method is finished and the analysis team can create an initial privacy impact assessment (PIA) report in the final step of the ProPAn method PIA Report Creation that is described in the following chapter (see also Chapter 8).

Part V.

Evaluation

Chapter 18

Using ProPAn to Assist Privacy Impact Assessments

In this chapter, I describe the final step of the ProPAn method called PIA Report Creation. In this step, the analysis team can create an initial privacy impact assessment (PIA) report based on the information collected and stored in the ProPAn model during the previous steps of the ProPAn method.

Initial ideas for the content of this chapter are published in the paper (Meis and Heisel, 2015) and its extended version (Meis and Heisel, 2016a). I am the main author of both papers, and Maritta Heisel provided valuable feedback to the content of the papers that helped me to improve them.

In Section 18.1, I provide an introduction to this chapter. The relations between the ProPAn method and the steps to be performed during a PIA are established in Section 18.2. The artifacts produced during the ProPAn method are related to the elements that PIA reports shall contain in Section 18.3. I explain how the information documented in the ProPAn model can be used to (automatically) generate an initial PIA report in Section 18.4. In Section 18.5, I assess how the state of the art privacy requirements engineering methods support an analysis team to conduct a privacy impact assessment. This chapter concludes in Section 18.6.

18.1. Introduction

Several countries prescribe or advise government departments and organizations to conduct a so-called privacy impact assessment (PIA) for systems that process personal data. Wright et al. (2011) define a PIA as follows: *"A privacy impact assessment is a methodology for assessing the impacts on privacy of a project, policy, programme, service, product or other initiative which involves the processing of personal information and, in consultation with stakeholders, for taking remedial actions as necessary in order to avoid or minimise negative impacts."* Wright et al. reviewed the PIA methods of seven countries, namely Australia, Canada, Hong Kong, Ireland, New Zealand, the United Kingdom, and the United States of America for the EU project PIAF. This project had the goal to provide recommendations on how a regulation for a PIA in the EU should look like. Article 35 of the EU general data protection regulation (GDPR) (European Commission, 2016) prescribes to conduct a data protection impact assessment (DPIA) when the processing of personal data *"is likely to result in a high risk to the rights and freedoms of natural persons"*. A DPIA is a PIA and hence, I use the term PIA to also refer to DPIAs in my thesis. The international standard ISO 29134 (ISO/IEC, 2017a) describes guidelines for conducting PIAs and creating PIA reports. This standard is also referenced by the Article 29 Data Protection Working Party (2017)[1] as a source of guidelines for conducting DPIAs and

[1] The Article 29 Data Protection Working Party was an advisory body consisting of representatives from the EU Member State's data protection authorities, and has been replaced by the European Data Protection Board.

creating corresponding reports.

In this chapter, I explain how the ProPAn method is related to PIAs, and how artifacts produced by the ProPAn method can be used to create an initial PIA report. As basis, I use the guidelines provided by ISO/IEC (2017a), because these are also referenced by the Article 29 Data Protection Working Party (2017) that provide guidelines for the conduction of DPIAs that are required by the GDPR.

18.2. Relation of ProPAn to PIAs

The ProPAn method consists of several steps that also have to be performed during a PIA. That is, following a privacy-by-design method like ProPAn contributes to PIAs, because the steps and artifacts to be produced overlap (see also Section 18.5). Note that ProPAn method as a privacy requirements engineering method mainly supports steps of a PIA that are related to the requirements engineering phase. Details of the design and architecture of the system are out of the scope of the ProPAn method.

ISO/IEC (2017a) describes in ISO 29134 a method for conducting PIAs. Tables 18.1 and 18.2 list the steps of this method and relate to these the steps and artifacts of the ProPAn method that support these PIA steps.

First, it has to be determined whether a PIA is necessary (No. 1). This step is similar

Table 18.1.: *Mapping between PIA method steps and elements of the ProPAn method (part 1)*

No.	ISO 29134 PIA step	Related ProPAn steps	Supporting ProPAn artifacts
1	Determine whether a PIA is necessary (threshold analysis)	Privacy Context Elicitation, Privacy Threshold Analysis	Context, problem, and domain knowledge diagrams, personal data relations, initial data flow graphs
2	Preparation of PIA	*see subpoints*	*see subpoints*
2.1	Set up the PIA team and provide it with directions	Privacy Risk Analysis	Likelihood and consequence scales, risk matrix
2.2	Prepare a PIA plan to determine the necessary resources for conducting the PIA	-	-
2.3	Describe what is being assessed	Privacy Context Elicitation, Privacy Threshold Analysis, Data Flow Analysis	Context, problem, and domain knowledge diagrams, personal data relations and availability, data flow graphs
2.4	Stakeholder engagement	*see subpoints*	*see subpoints*
2.4.1	Identify stakeholders	Privacy Context Elicitation, Privacy Threshold Analysis, Data Flow Analysis	Context, problem, and domain knowledge diagrams, personal data relations and availability, data flow graphs
2.4.2	Establish consultation plan	-	-
2.4.3	Consult with stakeholders	-	-

to the step Privacy Threshold Analysis (see Chapter 10) that builds on the additional context information elicited in the step Privacy Context Elicitation. The artifacts produced during these steps support the analysis team to decide whether a detailed privacy analysis is necessary, and consequently, whether a PIA is necessary.

Second, the PIA needs to be prepared (No. 2). For this, ISO 29134 proposes to set up the PIA team and to define acceptable risk levels for privacy (No. 2.1). This is done in the ProPAn method in the step Privacy Risk Analysis (see Chapter 13) by creating likelihood and consequence scales, and a risk matrix. The preparation of a PIA plan (No. 2.2) is out of the scope of the ProPAn method. The outputs of the steps Privacy Context Elicitation, Privacy Threshold Analysis, and Data Flow Analysis (see Chapter 11) can be used to describe the system to be assessed (No. 2.3). Especially during the step Data Flow Analysis, the personal data processing of the system is analyzed in depth and documented. Only the substep identify stakeholders (No. 2.4.1) of the step stakeholder engagement (No. 2.4) is supported by the ProPAn method. The outputs of the steps Privacy Context Elicitation, Privacy Threshold Analysis, and Data Flow Analysis provide an overview of the stakeholders (represented as biddable domains), whether these are data subjects, which personal data shall be available to them, in which amount, with which linkability, and for which duration.

Third, the PIA is performed (No. 3). The step Data Flow Analysis of the ProPAn method provides the personal data flows in the system (No. 3.1). An analysis of the implication of human interactions with the system (No. 3.2) is performed in the substep Identify privacy threats from deviations of the step Privacy Risk Analysis. In this substep, the analysis team especially assesses whether deviations of the expected behavior of humans have unintended implications. The relevant privacy requirements (No. 3.3) are identified in the step Privacy Requirements Identification (see Chapter 12) of the ProPAn method, and risks to them (No. 3.4) in the step Privacy Risk Analysis. The PIA steps privacy risk identification (No. 3.4.1), privacy risk analysis (No. 3.4.2), and privacy risk evaluation (No. 3.4.3) can directly be mapped to ProPAn's substeps Identify privacy threats from deviations, Assess privacy threats globally, and Evaluate privacy risks of the step Privacy Risk Analysis, respectively. Similarly, the substeps of the PIA method to prepare for treating privacy risks (No. 3.5) can be mapped to the substeps of the ProPAn's method step Privacy Measure Integration.

Fourth, things have to be done after the actual PIA (No. 4). Only the preparation of the PIA report (4.1) is in the scope of the ProPAn method. I explain which artifacts produced by the ProPAn method can be used to create a PIA report in the following section, and in Section 18.4 I explain how an initial PIA report can be created using artifacts produced during the ProPAn method and stored in a ProPAn model.

Table 18.2.: *Mapping between PIA method steps and elements of the ProPAn method (part 2)*

No.	ISO 29134 PIA step	Related ProPAn steps	Supporting ProPAn artifacts
3	Perform the PIA	*see subpoints*	*see subpoints*
3.1	Identify information flows of PII	Data Flow Analysis	personal data relations and availability, data flow graphs
3.2	Analyse the implications of the use case	Privacy Risk Analysis (Identify privacy threats from deviations)	Privacy threats
3.3	Determine the relevant privacy safeguarding requirements	Privacy Requirements Identification	Privacy requirements
3.4	Assess privacy risk	Privacy Risk Analysis	*see subpoints*
3.4.1	Privacy risk identification	Identify privacy threats from deviations	Privacy threats
3.4.2	Privacy risk analysis	Assess privacy threats globally	Privacy threats
3.4.3	Privacy risk evaluation	Evaluate privacy risks	Likelihood and consequence scales, risk matrix, privacy risks
3.5	Prepare for treating privacy risks	Privacy Measure Integration	*see subpoints*
3.5.1	Choose the privacy risk treatment options	Select privacy measures	List of technical and non-technical privacy measures
3.5.2	Determine controls	Integrate privacy measures	Privacy measures (problem, domain knowledge, aspect, and weaving diagrams)
3.5.3	Create privacy risk treatment plans	Document satisfaction arguments	Satisfaction arguments
4	Follow up the PIA	*see subpoints*	*see subpoints*
4.1	Prepare the report	PIA report creation	Initial PIA report
4.2	Publication	-	-
4.3	Implement privacy risk treatment plans	-	-
4.4	Review and/or audit of the PIA	-	-
4.5	Reflect changes to the process	-	-

18.3. Relation of ProPAn Artifacts to PIA Reports

Article 35 of the GDPR specifies elements that a DPIA should at least contain. These elements are further refined by the Article 29 Data Protection Working Party (2017) by providing a list of criteria for an acceptable DPIA. Tables 18.3 and 18.4 list the criteria for an acceptable DPIA and map the supporting artifacts of the ProPAn method to them.

First, a systematic description of the processing shall be provided (No. 1). For this, the nature, scope, context, and purposes of the processing shall be assessed (No. 1.1). This information is contained in the context, problem, and domain knowledge diagrams that describe the domains, interfaces, requirements, and domain knowledge of the system-to-be. Additionally, the elicited personal data relations and availability describe which personal data are processed and

Table 18.3.: *Mapping between the Article 29 Data Protection Working Party's criteria for an acceptable DPIA and supporting artifacts of the ProPAn method (part 1)*

No.	Art. 29 WP criteria	Supporting ProPAn artifacts
1	a systematic description of the processing is provided (Article 35(7)(a)):	*see subpoints*
1.1	nature, scope, context and purposes of the processing are taken into account (recital 90);	Context, problem, and domain knowledge diagrams, personal data relations and availability, data flow graphs, transparency requirements
1.2	a functional description of the processing operation is provided;	Problem diagrams
1.3	the assets on which personal data rely (hardware, software, networks, people, paper or paper transmission channels) are identified;	Context, problem, and domain knowledge diagrams, personal data relations and availability, data flow graphs
1.4	compliance with approved codes of conduct is taken into account (Article 35(8));	-
2	necessity and proportionality are assessed (Article 35(7)(b)):	*see subpoints*
2.1	measures envisaged to comply with the Regulation are determined (Article 35(7)(d) and recital 90), taking into account measures contributing to the proportionality and the necessity of the processing on the basis of:	*see subpoints*
2.1.1	specified, explicit and legitimate purpose(s) (Article 5(1)(b));	Transparency requirements
2.1.2	lawfulness of processing (Article 6);	Transparency requirements
2.1.3	adequate, relevant and limited to what is necessary data (Article 5(1)(c));	Considered as guidance during the flow analysis
2.1.4	limited storage duration (Article 5(1)(e));	Transparency requirements
2.2	measures contributing to the rights of the data subjects:	*see subpoints*
2.2.1	information provided to the data subject (Articles 12, 13 and 14);	Transparency requirements
2.2.2	right of access and to data portability (Articles 15 and 20);	Data subject intervention requirements
2.2.3	right to rectification and to erasure (Articles 16, 17 and 19);	Data subject intervention requirements
2.2.4	right to object and to restriction of processing (Article 18, 19 and 21);	Data subject intervention requirements
2.2.5	relationships with processors (Article 28);	Flow information requirements
2.2.6	safeguards surrounding international transfer(s) (Chapter V);	Privacy measures, flow information requirements, satisfaction arguments
2.2.7	prior consultation (Article 36).	-

at which domains due to which statements these are available, which is also visualized using data flow graphs and documented using transparency requirements. The functional description

Table 18.4.: *Mapping between the Article 29 Data Protection Working Party's criteria for an acceptable DPIA and supporting artifacts of the ProPAn method (part 2)*

No.	Art. 29 WP criteria	Supporting ProPAn artifacts
3	risks to the rights and freedoms of data subjects are managed (Article 35(7)(c)):	*see subpoints*
3.1	origin, nature, particularity and severity of the risks are appreciated (cf. recital 84) or, more specifically, for each risk (illegitimate access, undesired modification, and disappearance of data) from the perspective of the data subjects:	*see subpoints*
3.1.1	risks sources are taken into account (recital 90);	Threat diagrams
3.1.2	potential impacts to the rights and freedoms of data subjects are identified in case of events including illegitimate access, undesired modification and disappearance of data;	Deviations leading to privacy threats
3.1.3	threats that could lead to illegitimate access, undesired modification and disappearance of data are identified;	Deviations leading to privacy threats
3.1.4	likelihood and severity are estimated (recital 90);	Privacy risks
3.2	measures envisaged to treat those risks are determined (Article 35(7)(d) and recital 90);	Privacy measures
4	interested parties are involved	-
4.1	the advice of the DPO is sought (Article 35(2));	-
4.2	the views of data subjects or their representatives are sought, where appropriate (Article 35(9)).	-

of the processing operations (No. 1.2) is provided by the problem diagrams. The assets on which personal data rely (No. 1.3) are also documented using the personal data relations and availability that describe at which domains of the system the personal data shall be available. The compliance with approved codes of conduct (No. 1.4) is not in the scope of the ProPAn method.

Second, the necessity and proportionality of the personal data processing shall be documented (No. 2). The subpoints of this documentation need are addressed by the transparency and intervenability requirements elicited during the ProPAn method. Especially, because ProPAn's transparency and intervenability requirements taxonomies (see Chapters 5 and 6) and the generation of these requirements for a system (see Chapter 12) are derived from the controllers' obligations to transparency and the data subject rights specified in the GDPR.

Third, the risks to the rights and freedoms of data subjects shall be documented (No. 3). The privacy risks are documented using threat diagrams and privacy risks in a risk matrix in the ProPAn method. These address the subpoints considering privacy risks (No. 3.1). The privacy measures identified and documented using the ProPAn method need also to be part of a PIA report to document the measures envisaged to treat risks (No. 3.2).

Fourth, it shall be documented which and how interested parties were involved into the PIA (No. 3). This is out of the scope of the ProPAn method.

Article 29 Data Protection Working Party (2017) references also the international standard ISO 29134 (ISO/IEC, 2017a) as a source of guidelines for conducting DPIAs and corresponding reports. This standard specifies what a PIA report should contain and how it could be structured. I provide a mapping of ProPAn's artifacts to the PIA report elements they support in Table 18.5.

Introduction (No. 1), conclusion and decisions (No. 6), and PIA public summary (No. 7) may be supported by all artifacts of the ProPAn method. For these sections, the analysis team has to decide which elicited information shall be contained in which detail.

The scope of the PIA (No. 2) shall contain information about the process under evaluation (No. 2.1). The system requirement information (No. 2.1.1) can be provided by the context, problem, and domain knowledge diagrams that describe the system-to-be and also contain its functional requirements. The system design information (No. 2.1.2) can only partially be covered by the ProPAn method, because ProPAn is a privacy requirements engineering method that does not consider the design and architecture of the system in depth. However, the personal data relations and availability, and the data flow graphs may serve as valuable input for this report element. The operational plans and procedures information (No. 2.1.3) can be created based on the information about the personal data availability at the domains of the system, the selected privacy measures represented as problem, domain knowledge, aspect, and weaving diagrams, and the satisfaction arguments provided for the privacy requirements. The scope of the PIA shall also contain the risk criteria (No. 2.2) which are documented in the ProPAn method using likelihood and consequence scales, and a risk matrix. The resources and people involved (No. 2.3), and stakeholder consultations (No. 2.4) are out of the scope of the ProPAn method.

The privacy requirements shall be documented in a PIA report (No. 3). These are stored in the ProPAn model, and the ProPAn tool also generates textual presentations (cf. Chapter 7) of these that may be used in the PIA report.

A PIA report shall also contain information about the performed privacy risk assessment (No. 4). Including the sources of the risks (No. 4.1), the privacy threats and their likelihood to occur (No. 4.2), the consequences of the privacy threats on the privacy requirements and their level of impact (No. 4.3), an evaluation of the privacy risks implied by the threats (No. 4.4), and an analysis whether the identified privacy requirements can still be satisfied considering the identified privacy risks (No. 4.5). These elements can be created based on the threat diagrams created during ProPAn's Privacy Risk Analysis, and the satisfaction arguments created during the Privacy Measure Integration.

To document the risk treatment plan (No. 5), the privacy measures, threat diagrams (including the cures relations), the privacy requirements, and the satisfaction arguments documented in the ProPAn method may be used as input. This is, because these artifacts contain all information about the privacy risks, how these are treated, and why these treatments are expected to sufficiently mitigate the privacy risks and satisfy the privacy requirements.

Table 18.5.: *Mapping between PIA report elements and supporting artifacts of the ProPAn method*

No.	PIA report element	Supporting ProPAn artifacts
1	Introduction	All
2	Scope of PIA	*see subpoints*
2.1	Process under evaluation	*see subpoints*
2.1.1	System requirement information	Context, problem, and domain knowledge diagrams
2.1.2	System design information	Personal data relations and availability, data flow graphs
2.1.3	Operational plans and procedures information	Personal data availability, problem, domain knowledge, aspect, and weaving diagrams, satisfaction arguments
2.2	Risk criteria	Likelihood and consequence scales, risk matrix
2.3	Resources and people involved	-
2.4	Stakeholder consultation	-
3	Privacy requirements	Privacy requirements
4	Risk assessment	*see subpoints*
4.1	Risk sources	Threat diagrams
4.2	Threats and their likelihood	Threat diagrams
4.3	Consequences and their level of impact	Threat diagrams
4.4	Risk evaluation	Privacy risks
4.5	Compliance analysis	Satisfaction arguments
5	Risk treatment plan	Privacy measures (problem, domain knowledge, aspect, and weaving diagrams), threat diagrams, privacy requirements, satisfaction arguments
6	Conclusion and decisions	All
7	PIA public summary	All

18.4. PIA Report Creation

As shown in the previous section, the artifacts produced by the ProPAn method may be used to create an initial PIA report. In this section, I discuss how parts of such an initial PIA report may be filled automatically by the ProPAn tool based on the information available in the ProPAn model. This automatically filled PIA report serves as starting point for the analysis team to create a PIA report. In the following, I describe how textual representations of the information contained in the ProPAn model can be generated, which can be used for the PIA report elements listed in Table 18.5.

18.4.1. System Requirement Information

To document the system requirement information, the context diagram of the system-to-be can be included into the PIA report. As the domains and phenomena contained in the context diagram are *DocumentableElements* (cf. Figure 2.8 on page 21), a list consisting of the domain and phenomenon names, and their description can be added to the PIA report. Also the functional requirements, facts, and assumptions of the system are documentable elements (because these are subtypes of the class *Statement* (cf. Figure 2.10 on page 22). Hence, a list of the system's functional requirements, facts, and assumptions can be added to the PIA report including their textual description.

Application to EHS The functional requirements of the EHS are introduced in Chapter 4, and the domain knowledge on the EHS is partly presented in Chapter 9.

18.4.2. System Design Information

As mentioned before, ProPAn is a requirements analysis method and hence, does not in depth consider the design and architecture of the system-to-be. However, the generated data flow diagrams may be added to the PIA report. Additionally, a textual representation of the available data diagrams (cf. Chapter 11) for each domain of the system may be added to the PIA report. The ProPAn model contains the following information that can be used for the textual representation:

1. personal data available at the domain
2. data subject(s) of personal data
3. sensitivity of the personal data for the data subject(s)
4. how the personal data were collected from the data subject(s)
5. availability of the personal data concerning specific instances of the domain (individual instances, authorized entities, all possible instances; cf. Figure 11.3 on page 176)
6. amount of personal data available at the domain
7. to which degree the domain can link the personal data to the corresponding data subject(s)
8. duration for which the personal data are retained at the domain
9. due to which statements the personal data are available there (origin)
10. due to which statements are needed at the domain (purpose)

As also discussed in Chapters 10, 12, and 13, the ProPAn tool can automatically derive from the previously mentioned information which personal data are collected, stored, further provided to others, and derived from other personal data by the machine. All of this information may be derived from the ProPAn model to be added to the initial PIA report using customized text templates (cf. Section 7.1).

Application to the EHS The data flow graph for the data subject patient is shown in Figure 11.16 on page 201. The personal data of the patients and their properties are shown in the patient's personal data diagram (see Figure 11.13 on page 197), and the available data diagram for the domain EHR is shown in Figure 11.14 on page 198 and the available data diagrams for the domains insurance application and invoice in Figure 11.12 on page 194.

18.4.3. Operational Plans and Procedures Information

The operational plans and procedures concerning the processing and protection of personal data may be documented as privacy measures. Hence, a list of these privacy measures could be created based on the statements that are referred to by a satisfaction argument as measure (cf. Figure 17.6 on page 336).

Application to EHS The privacy measures discussed in Chapter 17 can be added to the operational plans and procedures information. These are the assumptions that

1. financial employees are bound to an obligation to confidentiality and participate in a security training,
2. the financial application provider monitors the financial application and their employees, and reports data breaches concerning the personal data of patients to the EHS provider, and
3. the EHS provider sends a postal notification to the patients whose personal data were involved in a data breach.

Additionally, I introduced the privacy measure data anonymization that is integrated into requirement R7, which is about the provision of electronic health records in an anonymized form to researchers for clinical research.

18.4.4. Risk Criteria

The risk criteria are given by the likelihood and consequence scales, and the risk matrix that are specified in the step Privacy Risk Analysis. The risk matrix is stored in the ProPAn model as shown in Figure 13.8 on page 263. To allow the analysis team to document also the likelihood and consequence scales, I extended the ProPAn metamodel as shown in Figure 18.1. Instances of the classes LikelihoodScale and ConsequenceScale contain for each likelihood and consequence literal one LikelihoodDefinition and ConsequenceDefinition object, respectively. These definition objects specify for a likelihood or consequence literal its meaning in natural language. Consequence scales may be assigned to privacy requirements. The consequence scale for a privacy requirement specifies the meaning of the consequence literals that are assigned to a Harms relation that targets the privacy requirement, and can be used to derive the privacy requirement's satisfaction criteria (see Chapter 17). To each object that has a likelihood, i.e., an instance of the class Causes or subtypes of the abstract class *Cause*, a LikelihoodScale has to be assigned. Note that I recommend in Chapter 13 to specify exactly one likelihood scale for the risk analysis of a system. However, the ProPAn metamodel allows the analysis team to specify different likelihood scales for a project. This may be useful for larger projects in which for different parts of the system different likelihood scales need to be specified.

Having stored the likelihood and consequence scales, and the risk matrix in the ProPAn model, these can automatically added to the PIA report by the ProPAn tool.

Application to EHS The likelihood scale for the EHS is shown in Table 13.8 on page 262, the consequence scales in Table 13.9 on page 262, and the risk matrix in Table 13.10 on page 264.

18.4.5. Privacy Requirements

For each privacy requirement of ProPAn's privacy requirements taxonomy, I specified a template that can automatically be instantiated by the ProPAn tool to derive a textual representation of the respective privacy requirement in Chapter 7. These textual representation can be added to the PIA report to document the identified privacy protection needs of the system.

Application to EHS Examples for textual descriptions of privacy requirements of the EHS can be found in Figure 17.3 on page 330.

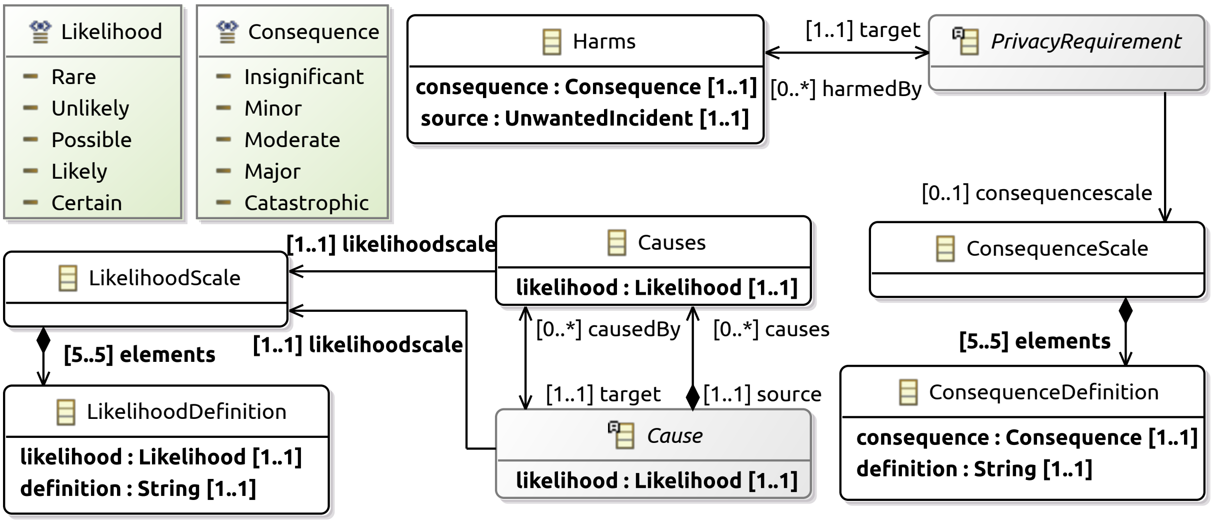


Figure 18.1.: *Metamodel for the documentation of likelihood and consequence scales in the ProPAn model*

18.4.6. Risk Sources, Threats and Their Likelihood, and Consequences and Their Level of Impact

The risk sources, threats and their likelihood, and consequences and their level of impact are documented all together in the threat diagrams created during the privacy risk analysis. Depending on the documentation needs for the PIA report, textual representations may be derived for the identified threat scenarios and unwanted incidents, or just for the identified risks. These may be complemented by the threat diagrams. The textual representations can be based, e.g., on the semantics (in natural language) that Lund et al. (2010) provide for their threat diagrams, because I adopted the notation used by Lund et al. in Chapter 13 for the threat diagrams used in the ProPAn method.

Application to EHS Two examples of threat diagrams for the EHS can be found in Figure 13.4 on page 259 and in Figure 13.5 on page 260.

18.4.7. Risk Evaluation

To document the risk evaluation in the PIA report, the risk matrix of the ProPAn model that specifies which combinations of likelihoods and consequences represent acceptable, tolerable, and unacceptable risks may be added to the PIA report. The identified risks may be added to the respective cells of this risk table.

Application to EHS The risk matrix for the EHS, which is filled with the considered privacy risks, is shown in Table 13.10 on page 264.

18.4.8. Compliance Analysis

The satisfaction arguments created in the step Privacy Measure Integration (see Chapter 17) can be added to the PIA report to document why the system complies to its privacy requirements under the light of the identified privacy risks.

Application to EHS I provide examples for satisfaction arguments in Table 17.3 on page 338.

18.4.9. Risk Treatment Plan

The privacy measures that are introduced to reduce the identified privacy risks to an acceptable level can be derived from the threat diagrams and the satisfaction arguments documented in the ProPAn model. Depending on the level of detail with which the privacy measures shall be documented, the description of the functional requirements, domain knowledge, aspects, and a description of how the aspects are integrated into the functional requirements they cross-cut can be added to the PIA report, or additionally the respective problem, domain knowledge, aspect, and weaving models with the corresponding behavioral views.

Note that some of the measures listed here may already be mentioned in the PIA report section *Operational plans and procedures information.* As a rough guideline, those privacy measures curing privacy risks belong to the section *Risk treatment plan* (prevention-based strategy; cf. Chapter 3), and those privacy measures that refine and implement the identified privacy requirements (refinement-based strategy; cf. Chapter 3) belong to the section *Operational plans procedures information.* The analysis team has to decide which measures shall be listed in which of these sections, or whether specific measures may be listed in both of them. This is, a privacy measure may be obtained following both, the refinement-based and the prevention-based strategies. The description of the satisfaction arguments may also be added to the PIA report to document the reasons why the analysis team considers all identified risks as sufficiently treated.

Application to EHS The non-technical privacy measures mentioned in Section 18.4.3 are also treatments to the privacy risks listed in the EHS' risk matrix and hence, may be listed as such.

18.5. Comparison to the State of the Art

All privacy requirements engineering methods introduced in Chapter 3 support at least parts of a PIA. To illustrate this, I mapped the subtasks of the privacy analysis process (see Figure 3.3 on page 36) to the high-level privacy requirements engineering method introduced in Chapter 3.2. This mapping is shown in Table 18.6.

Determining whether a PIA is necessary or not (No. 1) may be supported by the task Extend req.s specification and system model, as in this step privacy relevant information about the system is gathered, that may also be used for a threshold analysis. The definition of likelihoods, consequences and risks is part of the substep No. 2.1 of the preparation of a PIA. These definitions are normally obtained during the task Elicit privacy risks. To describe what is being assessed (No. 2.3), to identify the stakeholders of the system (No. 2.4.1), and to identify information flows of personal data (No. 3.1), the outputs of the task Extend req. specification and system model can be used. An analysis of the implications of the interactions with users (No. 3.2) could be obtained during the tasks Elicit privacy requirements, and Elicit privacy risks. The relevant privacy requirements (No. 3.3) are obtained during the task Elicit privacy requirements. The privacy risks are assessed (No. 3.4) in the task Elicit privacy risks. The tasks Refine privacy requirements and Treat privacy risks support the step to prepare the treatment of privacy risks (No. 3.5). The steps following up the PIA (No. 4) are not supported by a task of the high-level privacy requirements engineering method.

In the following, I only discuss those state of the art requirements engineering methods that explicitly support conducting a PIA or the creation of PIA reports.

Oetzel and Spiekermann (2014) describe a methodology to support the complete PIA process. Their methodology describes which steps have to be performed in which order to conduct a PIA. Hence, their methodology covers all necessary steps that have to be performed for a PIA. In contrast to our method, Oetzel and Spiekermann's methodology does not give concrete guidance on how to elicit the relevant information needed for a PIA, which is the focus of my work.

Table 18.6.: *Mapping between PIA method steps and subtasks of the high-level privacy requirements engineering method*

No.	ISO 29134 PIA step	Related Subtasks of the High-level Privacy Requirements Engineering Method
1	Determine whether a PIA is necessary (threshold analysis)	Extend req. specification and system model
2	Preparation of PIA	*see subpoints*
2.1	Set up the PIA team and provide it with directions	Elicit privacy risks
2.2	Prepare a PIA plan to determine the necessary resources for conducting the PIA	-
2.3	Describe what is being assessed	Extend req. specification and system model
2.4	Stakeholder engagement	*see subpoints*
2.4.1	Identify stakeholders	Extend req. specification and system model
2.4.2	Establish consultation plan	-
2.4.3	Consult with stakeholders	-
3	Perform the PIA	*see subpoints*
3.1	Identify information flows of PII	Extend req. specification and system model
3.2	Analyse the implications of the use case	Elicit privacy requirements and risks
3.3	Determine the relevant privacy safeguarding requirements	Elicit privacy requirements
3.4	Assess privacy risk	Elicit privacy risks
3.5	Prepare for treating privacy risks	Treat privacy risks
4	Follow up the PIA	-

The high-level privacy engineering methods of Drgon et al. (2016); Crespo et al. (2015) both contain tasks to perform two PIAs. First, an initial PIA after a description of the use cases of the system and a definition of initial privacy properties (e.g., personal data processed and privacy policies). Second, a final PIA after the detailed privacy analysis including a risk assessment and a privacy measure selection. However, Drgon et al.; Crespo et al. provide no further details on how a PIA shall be performed and which information needs to be collected for it. In contrast to these works, I provide guidance on conducting a PIA and the creation of a PIA report by discussing how a PIA is supported by the steps of the ProPAn method (and more generally of a privacy requirments engineering method; cf. Table 18.1), and which artifacts produced during the ProPAn method can be used for a PIA report.

18.6. Conclusions

In this chapter, I have shown how the ProPAn method supports to conduct a PIA by mapping its steps and artifacts to the steps of the PIA method described in ISO 29134 (ISO/IEC, 2017a). Additionally, I discussed which artifacts produced during the ProPAn method and documented in the ProPAn model can be used to fill a PIA report. Most of the PIA steps related to the privacy analysis of the system-to-be are supported by the ProPAn method. Only the involvement of stakeholders is not explicitly mentioned as a step in the ProPAn method. However, these stakeholders may be involved in all steps of the ProPAn method as domain or privacy experts (these roles are introduced in Chapter 3 and part of the analysis team). Furthermore, the planning of the PIA including the selection of the PIA team, the resources necessary to perform the PIA, and the timeframe in which the PIA shall be created are out of the scope of the ProPAn method.

A novel contribution to the state of the art is that the ProPAn tool allows to automatically fill parts of a PIA report using the information stored in the ProPAn model. However, the exact structure and content of the PIA report has to be framed by the analysis team. For example, the analysis team has to decide how detailed the system-to-be, its personal data processing behavior, its privacy requirements, risks to these privacy requirements, and selected privacy measures shall be described. For this, the analysis team may specify own text templates containing placeholders that are filled with the information stored in the ProPAn model. I have shown examples of such text templates in Chapter 7 to specify the meaning of the privacy requirements of my privacy requirements taxonomy.

In future research, it may be investigated whether information needed to be documented in a PIA report is not elicited during the ProPAn method and whether the ProPAn method can be enhanced to also document this information in the ProPAn model. Additionally, it could be evaluated whether analysis teams prefer the automatic generation of initial PIA reports that they have to customize, or whether they prefer to query the ProPAn model, e.g., using OCL expressions, and use the retrieved data to create a PIA report.

This chapter described the final step of the ProPAn method. In the next chapter, I apply the ProPAn method on an additional case study that is concerned with an course evaluation system for a university.

Chapter 19

Case Study (Course Evaluation)

In this chapter, I apply the ProPAn method on a second case study. This case study is concerned with the development of an online course evaluation system for a university. I have adapted this example from the privacy training workshop that was conducted in the context of the EU project PRIPARE[1].

I introduce the scenario of the course evaluation system in Section 19.1. This section presents all inputs provided to the ProPAn method. The application of the ProPAn method is shown in the Sections 19.2-19.9. Finally, I draw conclusions from the application of the ProPAn method on the course evaluation system in Section 19.10.

19.1. Scenario Introduction

In this section, I introduce the course evaluation system that I use as a second case study to evaluate the applicability of the ProPAn method. I give the description of the scenario in Section 19.1.1. Section 19.1.2 shows the context diagram for the course evaluation system. The functional requirements and the corresponding problem diagrams are presented in Sections 19.1.3 and 19.1.4, respectively.

19.1.1. Scenario Description

A course evaluation system shall be developed that allows teachers to receive feedback for the courses they have given. For this, they can create their own evaluation forms for each course they give. At the end of the course, the students who regularly participated in the course are allowed to evaluate it once. Hence, the students have to be able to register their participation in the course. Such a registration of participation shall only be possible for those students who actually participate in the course and that are officially enrolled in the course.

The administration of the university already runs a study management system that manages the students of the university, their contact information, the studies and courses of the university, the studies and courses the students are enrolled in, and the students' trials and grades for the courses they have taken. Additionally, the study management system manages the teachers of the universities, their university contact information, and the courses they give. The students and teachers shall be able to authenticate themselves in the course evaluation system using their credentials for the study management system. Using the study management system, the course evaluation system shall also be able to determine whether a student is enrolled in a course, and whether a teacher gives a specific course.

[1] `http://pripareproject.eu/events/privacy-training-workshop-3/` (accessed on 9 July 2018)

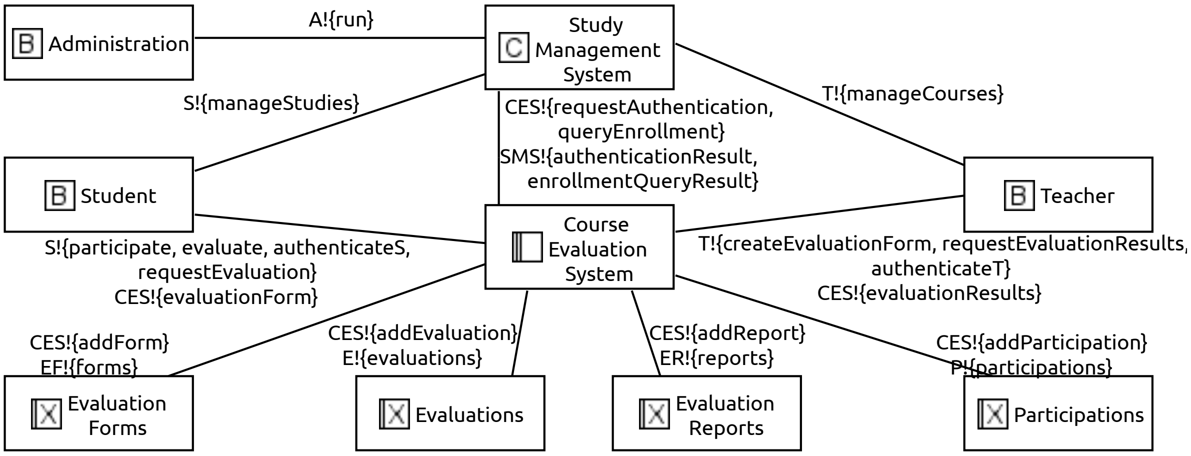


Figure 19.1.: *Context diagram for the course evaluation system*

19.1.2. Context Diagram

The context diagram of the course evaluation system is shown in Figure 19.1. The machine to be built is the Course Evaluation System and it manages the Evaluation Forms that Teachers can create for their courses, the Evaluations performed by Students, the final Evaluation Reports of the courses, and the registered Participations of the students in the courses. Teachers can create evaluation forms and request evaluation results from the Course Evaluation System. A Student can register his or her participation in a course and evaluate courses using the Course Evaluation System. To perform the above mentioned actions, Students and Teachers have to be authenticated. This is done using the existing Study Management System that is run by the Administration of the university. The Students and Teachers use the Study Management System to manage their studies and the courses they give, respectively. Furthermore, the Course Evaluation System can request from the Study Management System whether a Student is officially enrolled in the course he or she wants to participate in to later evaluate it. The Study Management System can also be asked whether a teacher gives a specific course for which he or she wants to create an evaluation form, or wants to access the evaluation results.

19.1.3. Functional Requirements

From the above scenario description, I derived the following functional requirements.

Create Forms Teachers shall be able to create evaluation forms for their courses.

Participate Students enrolled in a course shall be able to register their participation in a course.

Request Evaluation During the evaluation period, the students that regularly participated in a course shall be able to request the evaluation form for the course, unless they have not already evaluated the course.

Evaluate A student can fill out a received evaluation form and submit it to evaluate a course.

Create Report After the evaluation period, the course evaluation system shall create an evaluation report for each course that does not allow to link evaluations to single students.

Show Results Teachers shall be able to access the evaluation reports of their courses.

Authenticate Students and teachers shall authenticate themselves to the course evaluation system using the credentials they also use for the study management system.

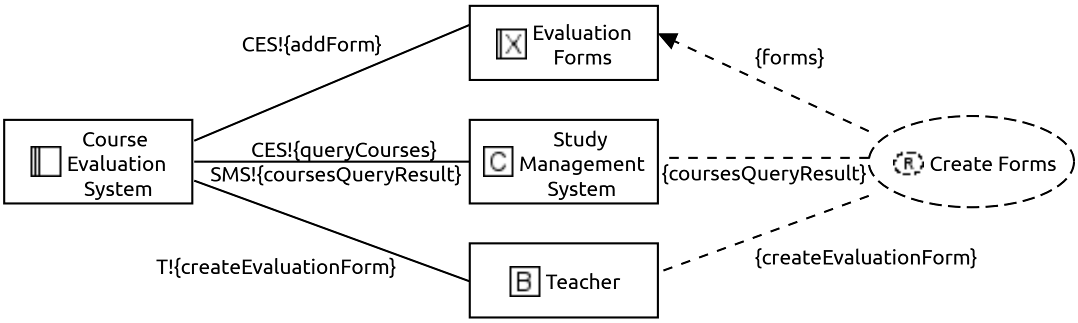


Figure 19.2.: *Problem diagram for requirement* Create Forms *of the course evaluation system*

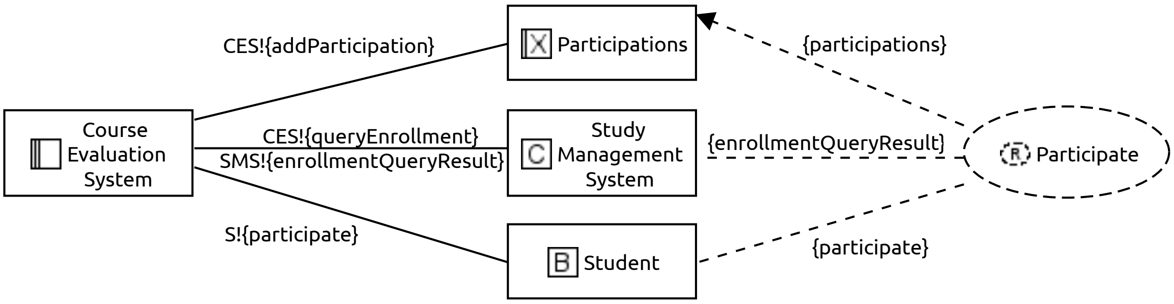


Figure 19.3.: *Problem diagram for requirement* Participate *of the course evaluation system*

19.1.4. Problem Diagrams

The problem diagram for requirement Create Forms is shown in Figure 19.2. Create Forms refers to the event createEvaluationForm controlled by the Teacher and observable by the Course Evaluation System. With this event the Teacher can create an evaluation form for one of his or her courses. To check whether a course is given by the respective Teacher, the Course Evaluation System can query the courses from the Study Management System. If the Teacher gives the course, then the Course Evaluation System shall add the form to the lexical domain Evaluation Forms that stores these. Create Forms constrains that the evaluation form created by the Teacher is stored in the domain Evaluation Forms.

Requirement Participate is concerned with the documentation of the Students' participation in the courses they take (see Figure 19.3). Participate refers to the event participate that is controlled by the Student and observable by the Course Evaluation System. The Course Evaluation System shall check whether the Student is properly enrolled in the course he or she wants to register a participation using the Study Management System. When the enrollmentQueryResult is positive, the Course Evaluation System shall add the participation to the lexical domain Participations, which is constrained by requirement Participate to store it.

In Figure 19.4, I show the problem diagram for requirement Request Evaluation. Request Evaluation refers to the event requestEvaluation controlled by the biddable domain Student, and the participations, forms and evaluations stored in the lexical domains Participations, Evaluation Forms and Evaluations, respectively. All these phenomena are observable by the Course Evaluation System. When the Student sufficiently often participated in the course and he or she did not yet evaluate the course, the evaluation form of the requested course shall be provided to him or her. Requirement Request Evaluation constrains that the Student receives a course's evaluation form when he or she satisfies the requirements to evaluate the course.

Figure 19.5 shows the problem diagram for requirement Evaluate. Evaluate refers to the event evaluate controlled by the biddable domain Student, and the forms stored in the lexical domain Evaluation Forms. The Student is only able to cause the event evaluate when he or she

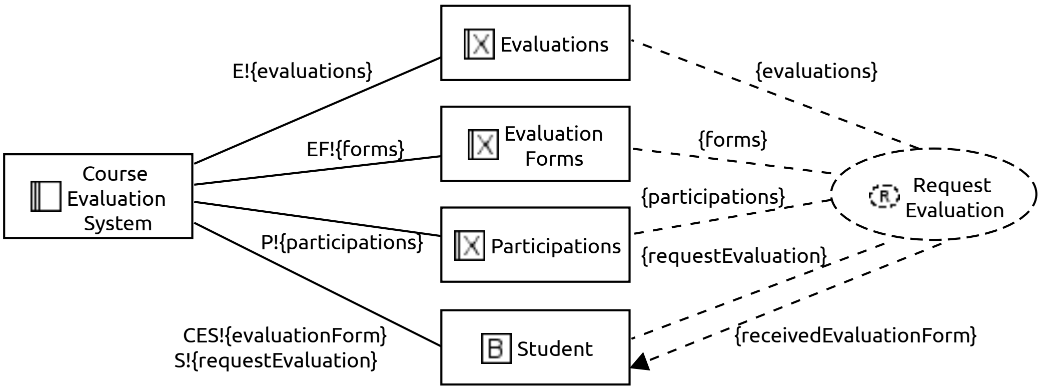


Figure 19.4.: *Problem diagram for requirement* Request Evaluation *of the course evaluation system*

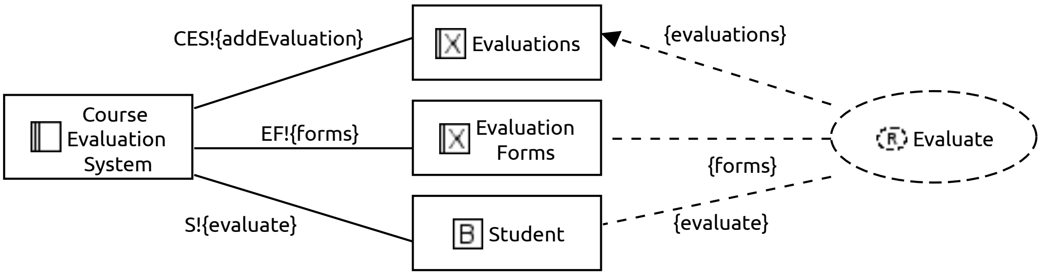


Figure 19.5.: *Problem diagram for requirement* Evaluate *of the course evaluation system*

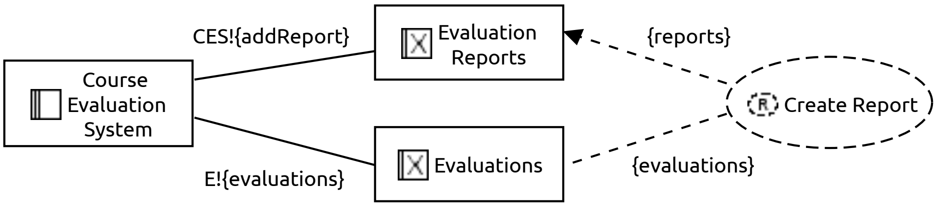


Figure 19.6.: *Problem diagram for requirement* Create Report *of the course evaluation system*

before successfully requested the evaluation of a course (see requirement Request Evaluation; Section 19.4. The filled out evaluation form provided by the Student is added by the Course Evaluation System to the lexical domain Evaluations, which is also constrained by requirement Evaluation to store the Student's evaluation of the course.

The problem diagram for requirement Create Report is shown in Figure 19.6. Create Report refers to the evaluations stored in the lexical domain Evaluations, and constrains that based on these evaluations, reports are created and stored in the lexical domain Evaluation Reports after the evaluation period. To satisfy this requirement, the Course Evaluation System can observe the evaluations and add evaluation reports to the lexical domain Evaluation Reports.

Requirement Show Results is concerned with providing the results of the course evaluation to the teachers of the respective course (see Figure 19.7). Show Results refers to the event requestEvaluationResults that is controlled by the Teacher and observable by the Course Evaluation System. The Course Evaluation System shall check whether the Teacher gave the course for which he or she requested the evaluation results using the Study Management System. When the coursesQueryResult shows that the Teacher gave the respective course, the Course Evaluation System shall provide the evaluationResults to the Teacher. The Course Evaluation System can retrieve the results from the lexical domain Evaluation Reports, which stores these. Requirement Show Results constrains that the Teacher received the requested evaluation results when he or

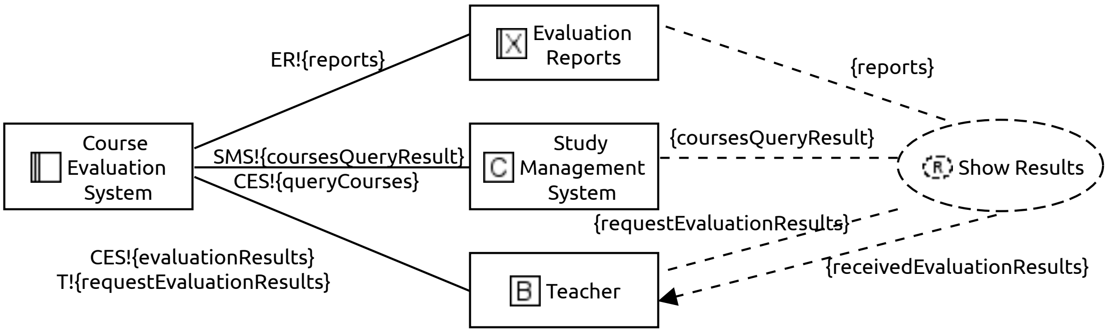


Figure 19.7.: *Problem diagram for requirement* Show Results *of the course evaluation system*

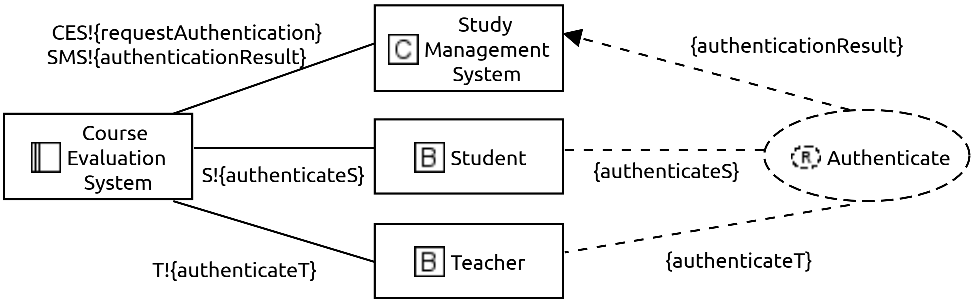


Figure 19.8.: *Problem diagram for requirement* Authenticate *of the course evaluation system*

she gave the associated course.

In Figure 19.8, I show the problem diagram for requirement Authenticate. Authenticate refers to the events authenticateS and authenticteT controlled by the biddable domains Student and Teacher, respectively. Both phenomena are observable by the Course Evaluation System. When one of these authentication requests is observed by the Course Evaluation System, it shall request the authentication from the Study Management System that manages user accounts for the Students and Teachers. In response to this request, the Study Management System provides the Course Evaluation System the result of the authentication. Requirement Authenticate constrains the Study Management System to provide the authenticationResult to the Course Evaluation System.

19.2. Privacy Context Elicitation

In this section, I apply the step Privacy Context Elicitation that I introduced in Chapter 9 on the course evaluation system. I show the application of the substeps Elicit indirect environment, Model indirect environment, Elicit connection domains, and Model connection domains in Sections 19.2.1-19.2.4, respectively.

19.2.1. Elicit Indirect Environment

To identify the privacy relevant indirect environment of the course evaluation system, I filled out the questionnaire for causal and lexical domains (see Table 9.1 on page 136) for the domain Study Management System, and the questionnaire for biddable domains (see Table 9.2 on page 137) for the domains Administration, Teacher, and Student. The answers for these domains are given in Table 19.1. As the lexical domains Participations, Evaluation Forms, Evaluations, and Evaluation Reports are part of the course evaluation system, developed with it, and only accessible through it, I did not identify an indirect environment for these lexical domains.

Table 19.1.: *Answers to the general questionnaires for the course evaluation system*

No.	Study Management System	Administration	Teacher	Student
GE1.1	-	Attacker	Attacker	Attacker
GE1.2	Attacker	Teacher, Student	Administration, Student	Administration, Teacher
GE1.3	-	-	-	-
GE1.4	Administration, Teacher, Student	/	/	/
GE2.1	Administration, Teacher, Student	Teacher, Student	Administration, Student	Administration, Teacher
GE2.2	Administration, Teacher, Student	Teacher, Student	-	-
GE2.3	Administration, Teacher, Student	/	/	/

The Study Management System may be attacked by an *Attacker* who tries to access the data processed by it. I also consider that this attacker may try to get information from the administration, teacher, and student by performing a social engineering attack. Note that different attackers could be modeled, however, I decided to model only one attacker for the course evaluation system who is able to perform different kinds of attacks. Furthermore, the administration, teacher, and student are considered as both possible counterstakeholders and data subjects to which and from which the study management system, administration, teacher, and student may provide and process personal data, respectively. I did not identify further indirect data subjects and counterstakeholders.

19.2.2. Model Indirect Environment

The indirect data subject and counterstakeholder relations elicited in Table 19.1 are documented in the domain knowledge diagrams shown in Figures 19.9 and 19.10.

Figure 19.9 shows the privacy relevant assumptions concerned with the data flows involving the Study Management System (abbreviated as SMS). The domain knowledge diagram lists the assumptions that the study management system stores data of students, teachers, and the administration. Additionally, the students, teachers, and administration can retrieve data from the study management system, which is also documented in the domain knowledge diagram. Finally, Figure 19.9 documents that the Attacker may capture data from the study management system.

The different data flows occurring between the biddable domains of the course evaluation system and the considered attacker are documented as assumptions in Figure 19.10. These data flows occur when the different biddable domains communicate with each other. Hence, students, teachers, and the administration may get data from each other, which is documented by two assumptions for each pair of these biddable domains. Additionally, it was documented in Table 19.1 that the Attacker may be able to perform social engineering attacks on the administration, teacher, and student. The latter is documented using three assumptions that constrain the attacker to get data from the three mentioned biddable domains.

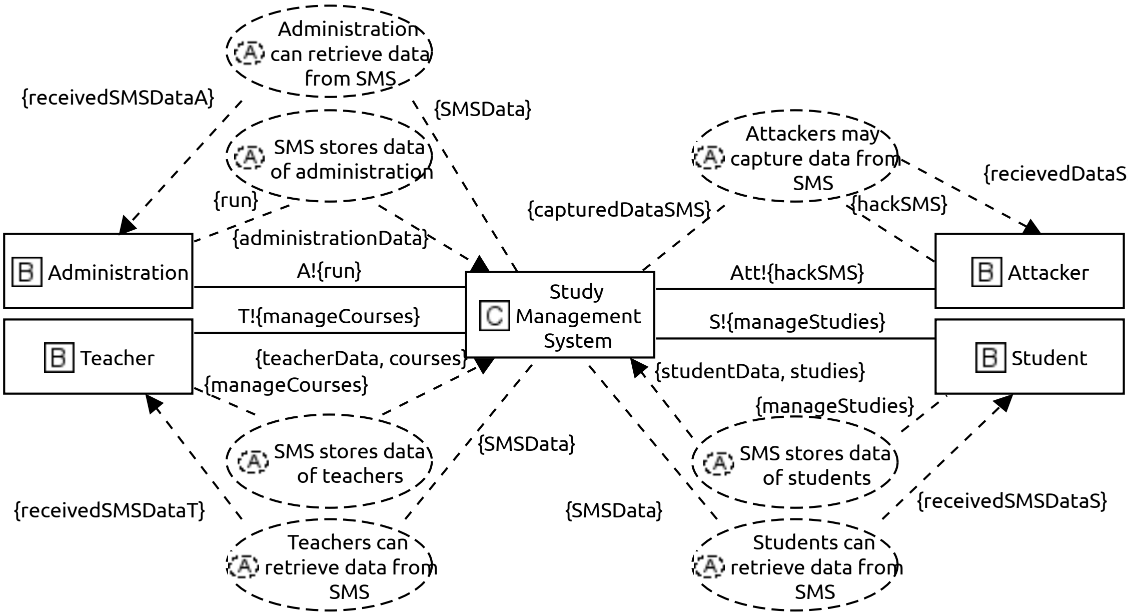


Figure 19.9.: *Domain knowledge diagram for the elicited indirect environment of the study management system*

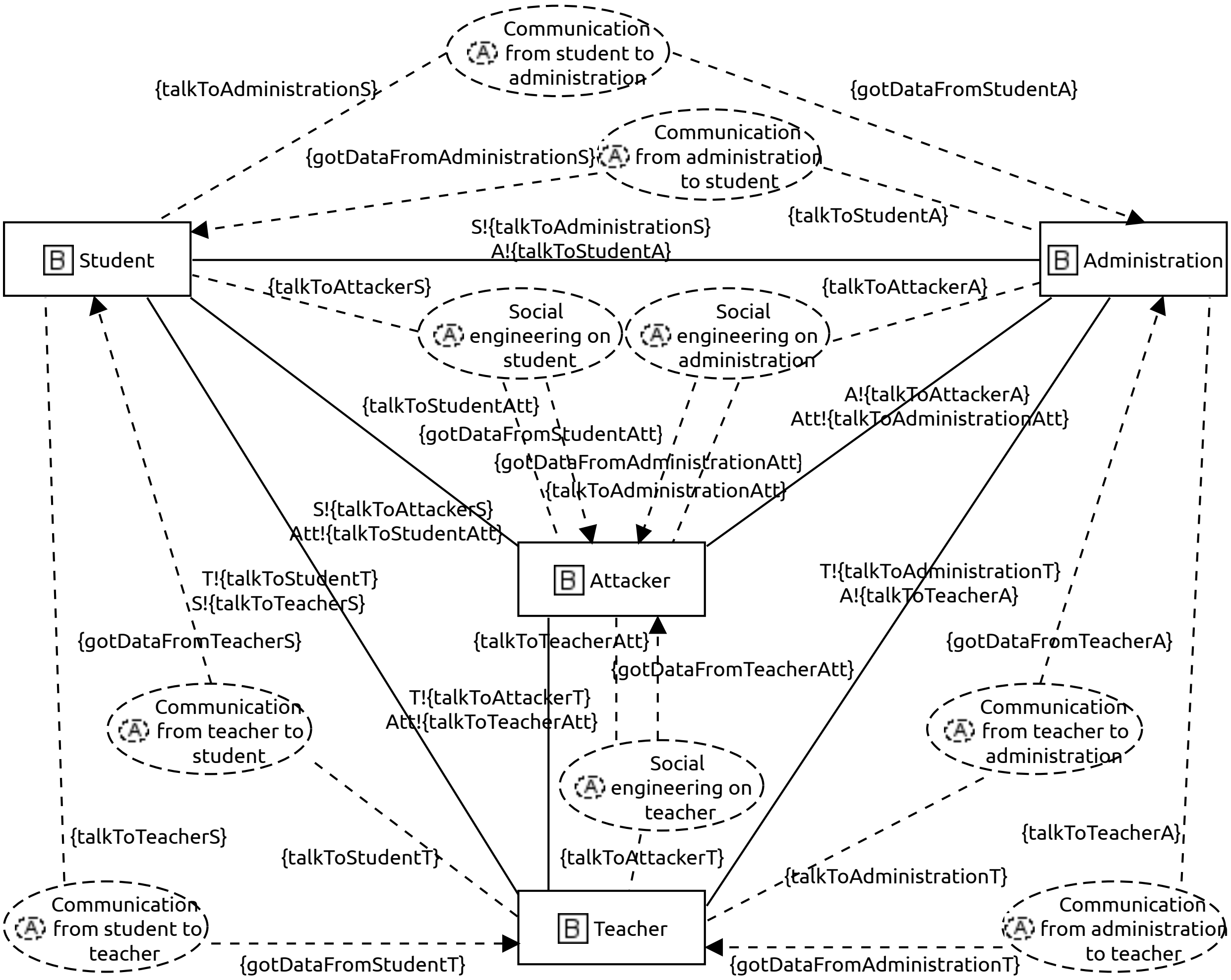


Figure 19.10.: *Domain knowledge diagram for the communication between the biddable domains involved in the course evaluation system*

Table 19.2.: *Answer to the general connection domain questionnaire for the course evaluation system*

Interface	GC1	GC2	GC3
Course Evaluation System - Student	Browser Student	all	Attacker
Course Evaluation System - Teacher	Browser Teacher	all	Attacker
Student Management System - Student	Browser Student	all	Attacker
Student Management System - Teacher	Browser Teacher	all	Attacker

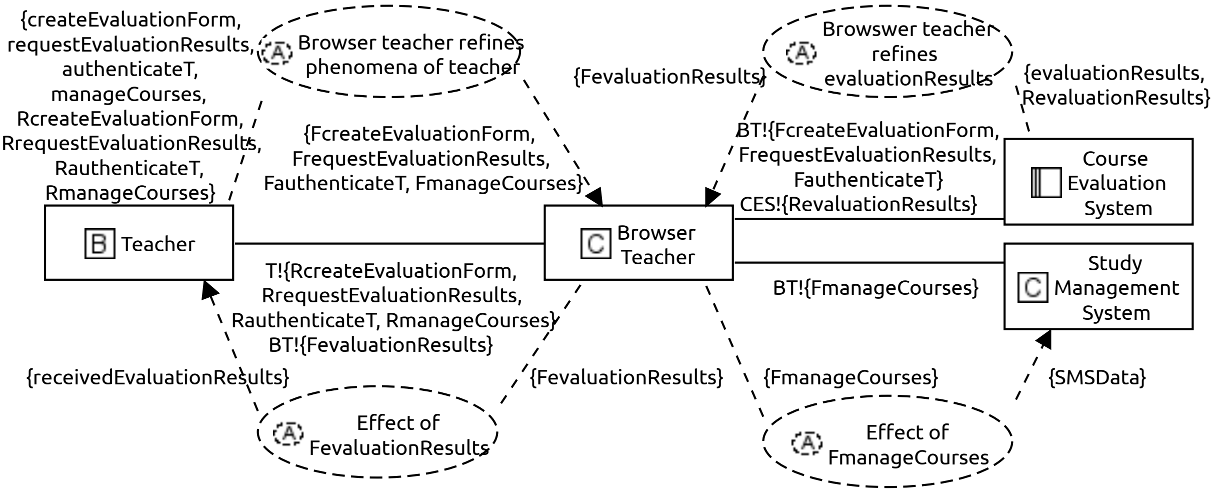


Figure 19.11.: *Domain knowledge diagram for the connection domain* Browser Teacher *in the course evaluation system*

19.2.3. Elicit Connection Domains

The different interfaces of the course evaluation system are shown in the context diagram (see Figure 19.1). The interfaces between the machine and the lexical domains are not further refined, as these lexical domains shall be a part of the machine. I also decided not to refine the interface between the Administration and the Study Management System, as well as the interface between the Study Management System and the Course Evaluation System, because these interfaces shall be realized inside the internal network of the university. For the sake of simplicity, I assume that this network is sufficiently protected.

The Study Management System and the Course Evaluation System shall be accessible for the Student and the Teacher through the internet using a browser. As this may introduce vulnerabilities, I decided to introduce the students' and teachers' browser as connection domains. Table 19.2 shows the answers to the questions listed in Table 9.7 on page 144 for the respective interfaces. The *Browser Student* and *Browser Teacher* refine all phenomena of the interfaces between the course evaluation system and study management system, and the student and teacher, respectively. Both browsers are considered to be vulnerable against attacks of the attacker.

19.2.4. Model Connection Domains

I instantiated the pattern for domain knowledge diagrams for connection domains (see Figure 9.11 on page 146) twice, once for each browser. The resulting domain knowledge diagram for the connection domain Browser Teacher is shown in Figure 19.11. Analogously, I created also a domain knowledge diagram for Browser Student, which I omit for the sake of simplicity. These domain knowledge diagrams document that the browsers mediate between the teacher and student, and the course evaluation system and study management system. This also means that personal data of students and teachers may be available at their browsers at least for a

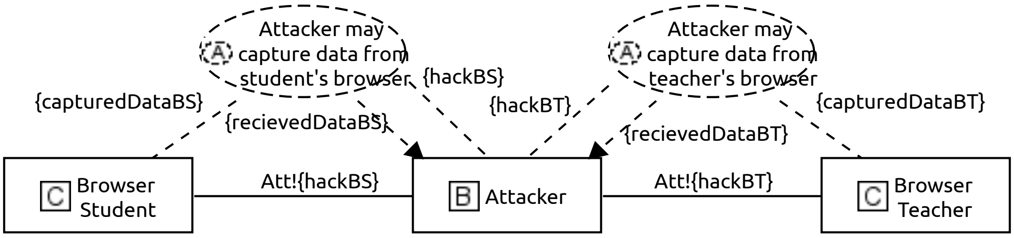


Figure 19.12.: *Domain knowledge diagram for the counterstakeholder Attacker and the connection domains Browser Student and Browser Teacher*

short time.

The domain knowledge diagram shown in Figure 19.12 contains the two assumptions that the Attacker may capture data that are available at the Browser Student and the Browser Teacher.

For the student's and teacher's browser I do not identify further indirect counterstakeholders or data subjects. Hence, the step Privacy Context Analysis is finished.

19.3. Privacy Threshold Analysis

Having identified the privacy relevant indirect environment and connection domains, the step Privacy Threshold Analysis (see Chapter 10) can be performed. I identify and model the personal data processed by the course evaluation system in Sections 19.3.1 and 19.3.2, respectively. The generated initial data flow graphs are shown in Section 19.3.3, and the decision on the privacy threshold for the course evaluation system is given in Section 19.3.4.

19.3.1. Identify Data Subjects and Personal Data

The biddable domains in the course evaluation system are the Administration, Teacher, Student, and Attacker. The phenomena of these domains that are referred to by the functional requirements introduced in Section 19.1, and the assumptions identified in Section 19.2 are listed in Table 19.3. Note that these phenomena are all causal phenomena, more specifically events (cf. Figure 2.9 on page 21). I do not expect that the administration and attacker communicate any personal data of them through the causal phenomena they control. Hence, I only further analyze the biddable domains teacher and student.

Table 19.4 shows the personal data that I identified to be contained or transmitted by the

Table 19.3.: *Overview of phenomena of biddable domains referred to by statements*

Biddable domain	Causal phenomena
Administration	run, talkToAttackerA, talkToStudentA, talkToTeacherA
Teacher	createEvaluationForm, requestEvaluationResults, authenticateT, manageCourses, RcreateEvaluationForm, RrequestEvaluationResults, RauthenticateT, RmanageCourses, talkToAttackerT, talkToStudentT, talkToAdministrationT
Student	participate, requestEvaluation, evaluate, authenticateS, manageStudies, Rparticipate, RrequestEvaluation, Revaluate, RauthenticateS, RmanageStudies, talkToAttackerS, talkToAdministrationS, talkToTeacherS
Attacker	hackSMS, hackBS, hackBT, talkToAdministrationAtt, talkToStudentAtt, talkToTeacherAtt

Table 19.4.: *Data identified from the causal phenomena of the domain* Teacher

Causal phenomenon	Containing or transmitting	Metadata
createEvaluationForm, RcreateEvaluationForm	evaluationForm, courses	usedDeviceAndDateT
requestEvaluationResults, RrequestEvaluationResults	courses, aggregatedEvaluations	usedDeviceAndDateT
authenticateT, RauthenticateT	credentialsT	usedDeviceAndDateT
manageCourses, RmanageCourses	credentialsT, courses, teacherData	usedDeviceAndDateT
talkToStudentT, talkToAdministrationT, talkToAttackerT	courses, teacherData	-

causal phenomena controlled by the biddable domain Teacher. I obtained this table by answering the questions listed in Table 10.1 on page 154. The causal phenomenon createEvaluationForm and its refined version RcreateEvaluationForm transmit the evaluationForm that the teacher creates and the courses for which this form shall be used. The phenomena requestEvaluationResults and RrequestEvaluationResults transmit the courses of which the teacher requests the evaluation results, and they transmit the aggregatedEvaluations back to the teacher. The teacher's credentialsT are transmitted by the causal phenomena authenticateT, and RauthenticateT. During the interaction with the Study Management System teachers may transmit the data credentialsT, courses, and teacherData to it using the causal phenomena manageCourses and RmanageCourses. The data teacherData represent all information about the teacher that are persisted in the Study Management System. For all above listed causal phenomena controlled by the teacher, I expect that the usedDeviceAndDateT are transmitted or observable as metadata for the recipient. Additionally, I expect that teachers may communicate the courses they give, and their teacherData to students, the administration, and (social engineering) attackers through the causal phenomena talkToStudentT, talkToAdministrationT, and talkToAttackerT.

To assess whether the above identified data are personal data of teachers and how they are related to teachers, I answered the questions given in Table 10.2 on page 154. The answers are given in Table 19.5. All identified data are considered as personal data of teachers because these are related to them, and as non-sensitive according to the categories of sensitive personal data provided in Article 9 of the EU General Data Protection Regulation (GDPR) (European Commission, 2016). I expect that the evaluationForm may be linkable to a medium-sized group of potential teachers that might have created it, that the courses and aggregatedEvaluations may be linked to a small group of teachers, and that the teacherData, credentialsT, and usedDeviceAndDateT allow to uniquely identify the corresponding teacher. The evaluationForm, courses, and credentialsT are direct inputs of the teacher to the course evaluation system, the courses, teacherData, and credentialsT are (also) reused from the existing study management system, the aggregatedEvaluations are derived from the students' course evaluations, and the usedDeviceAndDateT are indirectly collected from teachers.

The definition of the literals of the enumeration Linkability (see Figure 10.2 on page 157) in the context of the course evaluation system is given in Table 19.6. The literal single refers to exactly one person. A smallGroup, mediumGroup, and largeGroup is considered to consist of 2-5, 6-50, and 51-200 persons, respectively. Data are considered as anonymous when they are at most linkable to a group of more than 200 persons.

Table 19.7 shows the personal data that I identified to be contained or transmitted by the causal phenomena controlled by the biddable domain Student. I obtained this table by answering the questions listed in Table 10.1 on page 154. The causal phenomenon participate and its

Table 19.5.: *Personal data identified for the data subject* Teacher

Data	Personal?	Sensitive?	Linkability	Collection
evaluationForm	yes	no	mediumGroup	direct
courses	yes	no	smallGroup	direct, reused
aggregatedEvaluations	yes	no	smallGroup	derived
teacherData	yes	no	single	reused
credentialsT	yes	no	single	direct, reused
usedDeviceAndDateT	yes	no	single	indirect

Table 19.6.: *Definition of* Linkability *literals for the course evaluation system*

Linkability literal	Definition
single	1 person
smallGroup	2-5 persons
mediumGroup	6-50 persons
largeGroup	51-200 persons
anonymous	>200 persons

Table 19.7.: *Data identified from the causal phenomena of the domain* Student

Phenomenon	Containing or transmitting	Metadata
participate, Rparticipate	participationDateAndCourse	usedDeviceAndDateS
requestEvaluation, RrequestEvaluation	takenCourses	usedDeviceAndDateS
evaluate, Revaluate	takenCourses, opinionOnCourseAndTeacher	usedDeviceAndDateS
authenticateS, RauthenticateS	credentialsS	usedDeviceAndDateS
manageStudies, RmanageStudies	credentialsS, studies, studentData	usedDeviceAndDateS
talkToTeacherS, talkToAdministrationS, talkToAttackerS	studies, takenCourses, studentData, participationDateAndCourse	-

refined version Rparticipate transmit the participationDateAndCourse the student attended. The phenomena requestEvaluation and RrequestEvaluation transmit the takenCourses for which the student requests to evaluate. With the causal phenomena evaluate and Revaluate the student can evaluate a course he or she successfully requested to evaluate. These phenomena transmit the takenCourses that the student evaluates and opinionOnCourseAndTeacher of the respective student. The students' credentialsS are transmitted by the causal phenomena authenticateS, and RauthenticateS. During the interaction with the Study Management System students may transmit the data credentialsS, studies, and studentData to it using the causal phenomena manageStudies and RmanageStudies. The data studentData represent all information about the student that are persisted in the Study Management System. For all above listed causal phenomena controlled by the student, I expect that the usedDeviceAndDateS are transmitted or observable as metadata for the recipient. Additionally, I expect that students may communicate their studies, takenCourses, participationDateAndCourse, and studentData at least partially to teachers, the administration, and (social engineering) attackers (causal phenomena talkToStudentS, talkToAdministrationS, and talkToAttackerS).

Table 19.8.: *Personal data identified for the data subject* Student

Phenomenon	Personal?	Sensitive?	Linkability	Collection
participationDateAndCourse	yes	no	mediumGroup	direct
takenCourses	yes	no	mediumGroup	direct
opinionOnCourseAndTeacher	yes	yes	mediumGroup	direct
studentData	yes	no	single	reused
studies	yes	no	mediumGroup	reused
credentialsS	yes	no	single	direct, reused
usedDeviceAndDateS	yes	no	single	indirect

To assess whether the above identified data are personal data of students and how they are related to students, I answered the questions given in Table 10.2 on page 154. The answers are given in Table 19.8. All identified data are considered as personal data of students because these are related to them, and only the personal data opinionOnCourseAndTeacher are considered as sensitive personal data that deserve special protection, because teachers shall not be able to link these data to the student who wrote it. I expect that the participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, and studies may be linkable to a medium-sized group of potential students, i.e., those students who take the respective course. I consider the studentData, credentialsS, and usedDeviceAndDateS to uniquely identify the corresponding student. The participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, and credentialsS are direct inputs of the student to the course evaluation system, the studentData, studies, and credentialsS are (also) reused from the excising study management system, and the usedDeviceAndDateS are indirectly collected from students.

19.3.2. Model Personal Data Relations

For the sake of simplicity I omit the personal data diagrams for the student and teacher. The information contained in these is already presented in Tables 19.4, 19.5, 19.7, and 19.8. The personal data diagrams only contain the statements from which the personal data were identified as traceability links (cf. Figure 10.2 on page 157).

19.3.3. Generate Initial Data Flow Graphs

The initial data flow graphs for the Student and Teacher are generated by the ProPAn tool. The initial data flow graphs only differ in the ordering of the nodes. For the sake of simplicity, I only show the initial data flow graph for the Student in Figure 19.13. Both initial data flow graphs show that personal data of students and teachers may flow to each domain of the system and possibly through different paths. For example, the Attacker may obtain personal data directly from a Student or Teacher by performing a social engineering attack, or indirectly by attacking the Administration, Browser Teacher, Browser Student, or Study Management System. Note that this graph is an over-approximation of the actual data flows. It is refined in the step Data Flow Analysis (see Section 19.4) if it is decided that a detailed privacy analysis is necessary.

19.3.4. Decide on Privacy Threshold

Based on the identified personal data and over-approximated data flows, I have now to decide whether a further privacy analysis shall be performed or not. This decision is supported by the answers to questions TA1-TA5 specified in Table 10.6 on page 167, which can automatically be derived from the ProPAn model of the course evaluation system. The derived answers are given in Table 19.9.

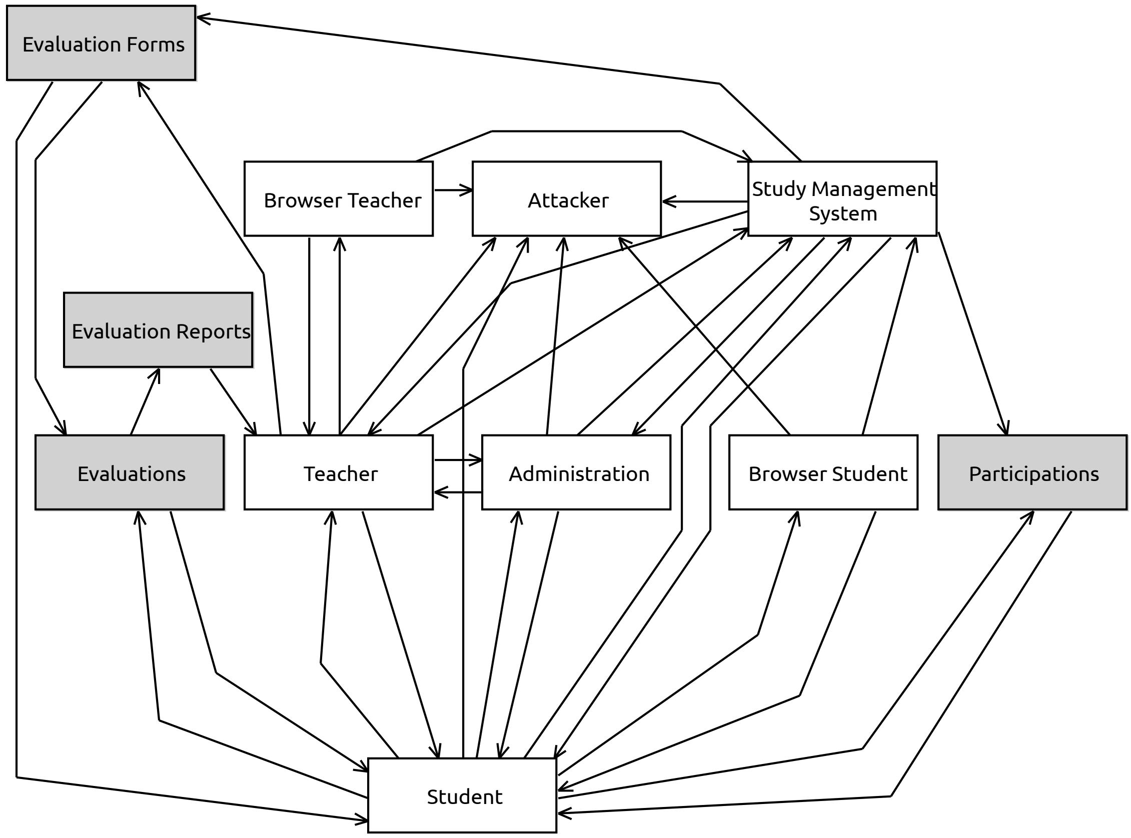


Figure 19.13.: *Initial data flow graph for the data subject Student*

Table 19.9.: *Threshold analysis answers for the EHS*

No.	Personal data
TA1 (collection)	{participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, courses, aggregatedEvaluations, teacherData, evaluationForm, credentialsT, usedDeviceAndDateT}
TA2 (storage)	{participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, courses, aggregatedEvaluations, teacherData, evaluationForm, credentialsT, usedDeviceAndDateT}
TA3 (flow)	{participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, courses, aggregatedEvaluations, teacherData, evaluationForm, credentialsT, usedDeviceAndDateT}
TA4 (identifiable)	{studentData, credentialsS, usedDeviceAndDateS, teacherData, credentialsT, usedDeviceAndDateT}
TA5 (sensitive)	{opinionOnCourseAndTeacher}

The initial data flow graph shown Figure 19.13 is highly connected and hence, all personal data processed by the system may be collected, stored, and provided to other domains of the system by the machine. This is documented in the answers to questions TA1-TA3. The answer to question TA5 shows that the sensitive personal data opinionOnCourseAndTeacher of the Student are processed that need special protection. Additionally, the answer to TA4 shows that the machine processes identifiable personal data of both, teachers and students.

This could lead to situation where a teacher may link the sensitive personal data opinionOnCourseAndTeacher to the corresponding student, which shall not be possible. Hence, I decide to perform a detailed privacy analysis of the course evaluation system for the data subject student.

The personal data of teachers processed by the system is mostly publicly available, because the courses and university contact information of a teacher are in most cases published on their public webpage. The aggregated evaluation that are also personal data of the respective teacher shall only be provided to him or her, and shall not be made publicly available. For the sake of simplicity, I do not pursue a further privacy analysis concerning the teacher's credentials, used devices and access dates, the aggregated evaluation concerning their courses, and their created evaluation forms. The latter I do not consider as privacy relevant, and for the protection of the credentials, used devices and access dates, and the aggregated evaluations the same or similar measures may be taken as for the corresponding personal data of students.

19.4. Data Flow Analysis

In this section, I present the results of the application of the step Data Flow Analysis (see Chapter 11) for the data subject student. For the sake of simplicity, I only show the results of the data flow analysis and no detailed information on the consideration of each statement. I present the final personal data diagram for the data subject student in Section 19.4.1. The available data diagrams that I obtained during the data flow analysis are discussed in Section 19.4.2. Finally, I show the final data flow graph for the student that is generated based on the identified and documented data flows in Section 19.4.3.

19.4.1. Final Personal Data Diagram

Figure 19.14 shows the final personal data diagram for the Student. For the sake of simplicity, I omitted the causal phenomena controlled by the student. In comparison to the student's personal data identified during the step Privacy Threshold Analysis (see Section 19.3), Figure 19.14 additionally lists the personal data studentID, sufficientNumberOfParticipationsForCourse, and aggregatedEvaluations.

The personal data studentID are contained in the studentData, allow to identify a single student, and are considered as non-sensitive. The personal data studentID were identified from functional requirement Participate, as an identifier for each student needs to be stored to know which personal data participationDateAndCourse belong to which student (see also Figure 19.15).

The information whether a student sufficiently often participated in a course in order to evaluate it is represented by the personal data sufficientNumberOfParticipationsForCourse. These data are considered as non-sensitive, and to be linkable to the students enrolled in the respective course, which is considered to be medium-sized group (cf. Table 19.6). Whether a student sufficiently often participated in a course to evaluate it is determined by the machine that is responsible to satisfy the functional requirement Request Evaluation. This is done based on the personal data participationDateAndCourse.

From the personal data studies and opinionOnCourseAndTeacher, the personal data aggregatedEvaluations are derived during the creation of the evaluation reports (functional requirement Create Report). In contrast to the opinionOnCourseAndTeacher, the aggregatedEvaluations are considered as non-sensitive personal data for students. This is, because the aggregatedEvaluations summarize the opinionOnCourseAndTeacher of all evaluations without allowing teachers to know that specific answers originate from the same student. The aggregated evaluations are still linkable to the group of students who participated in the course.

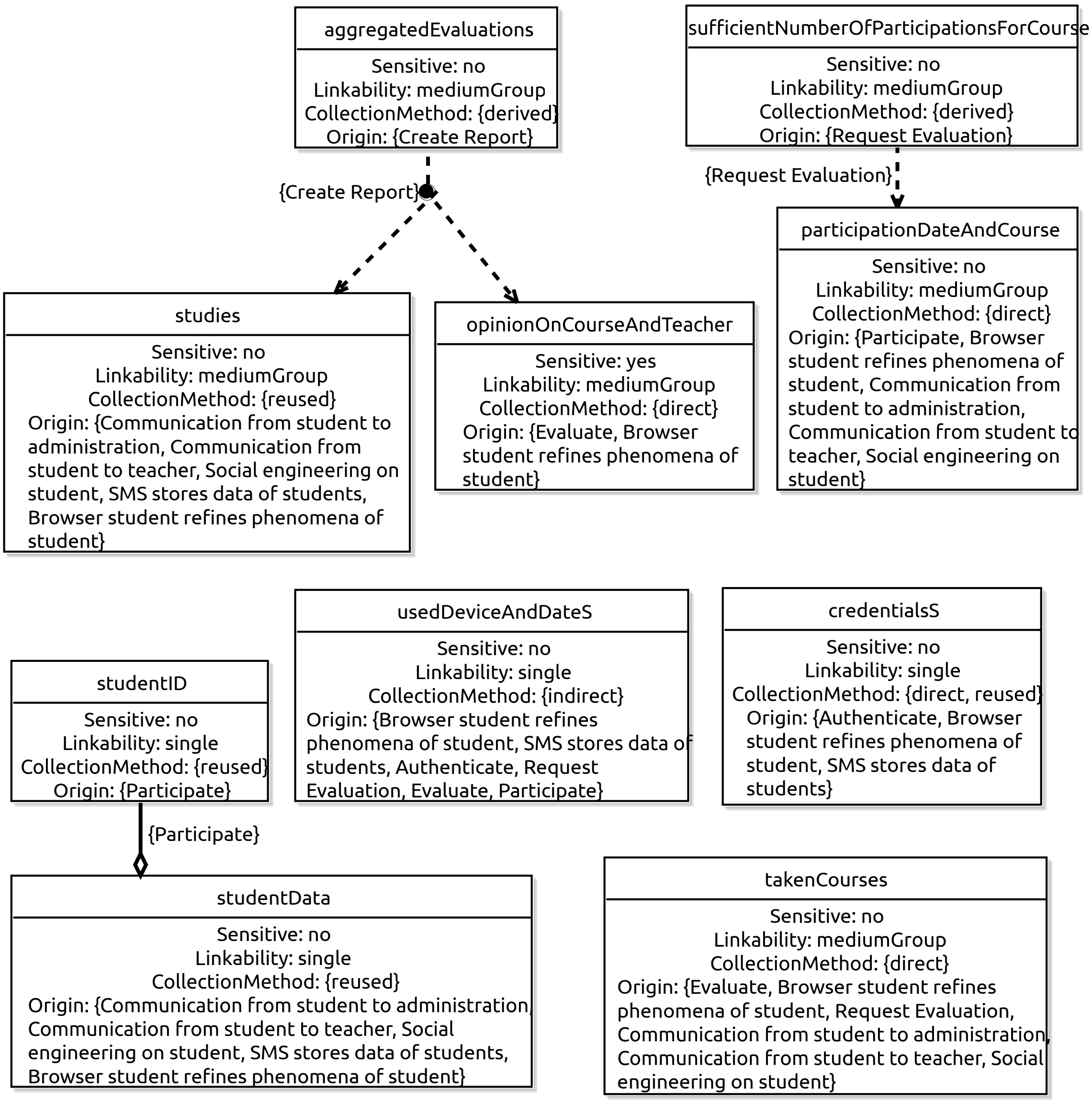

aggregatedEvaluations
Sensitive: no
Linkability: mediumGroup
CollectionMethod: {derived}
Origin: {Create Report}
sufficientNumberOfParticipationsForCourse
Sensitive: no
Linkability: mediumGroup
CollectionMethod: {derived}
Origin: {Request Evaluation}
{Create Report}
{Request Evaluation}
participationDateAndCourse
Sensitive: no
Linkability: mediumGroup
CollectionMethod: {direct}
Origin: {Participate, Browser student refines phenomena of student, Communication from student to administration, Communication from student to teacher, Social engineering on student}
studies
Sensitive: no
Linkability: mediumGroup
CollectionMethod: {reused}
Origin: {Communication from student to administration, Communication from student to teacher, Social engineering on student, SMS stores data of students, Browser student refines phenomena of student}
opinionOnCourseAndTeacher
Sensitive: yes
Linkability: mediumGroup
CollectionMethod: {direct}
Origin: {Evaluate, Browser student refines phenomena of student}
usedDeviceAndDateS
Sensitive: no
Linkability: single
CollectionMethod: {indirect}
Origin: {Browser student refines phenomena of student, SMS stores data of students, Authenticate, Request Evaluation, Evaluate, Participate}
credentialsS
Sensitive: no
Linkability: single
CollectionMethod: {direct, reused}
Origin: {Authenticate, Browser student refines phenomena of student, SMS stores data of students}
studentID
Sensitive: no
Linkability: single
CollectionMethod: {reused}
Origin: {Participate}
{Participate}
studentData
Sensitive: no
Linkability: single
CollectionMethod: {reused}
Origin: {Communication from student to administration, Communication from student to teacher, Social engineering on student, SMS stores data of students, Browser student refines phenomena of student}
takenCourses
Sensitive: no
Linkability: mediumGroup
CollectionMethod: {direct}
Origin: {Evaluate, Browser student refines phenomena of student, Request Evaluation, Communication from student to administration, Communication from student to teacher, Social engineering on student}

Figure 19.14.: *Final personal data diagram for the data subject student*

19.4.2. Available Data Diagrams

In this section, I present the available data diagrams that are relevant for the generation of the system's privacy requirements. These are the available data diagrams of the designed domains (see Section 19.4.2.1), the domains to which personal data flow from a designed domain or due to a functional requirement, and the biddable domains of the system (see Section 19.4.2.2) (cf. Chapter 12). Note that all given domains to which personal data of students flow due to the course evaluation system are biddable domains (see also the final data flow graph for the data subject student shown in Figure 19.20).

19.4.2.1. Designed Domains

At the lexical domain Evaluation Forms no personal data of students are available, because these forms are just templates created by teachers.

The available data diagram for the lexical domain Participations is shown in Figure 19.15. At the Participations, the personal data participationDateAndCourse and studentID are available, and

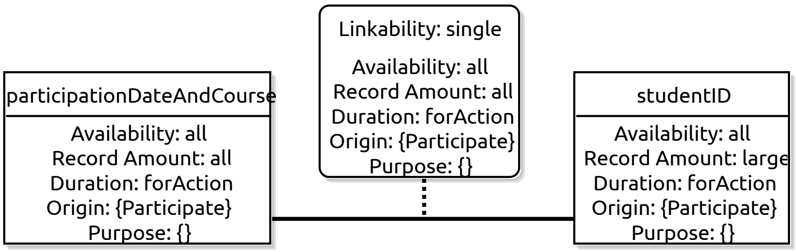


Figure 19.15.: *Available data diagram for the designed domain* Participations

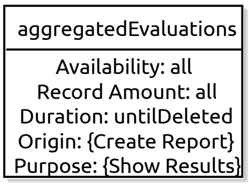


Figure 19.16.: *Available data diagram for the designed domain* EvaluationReports

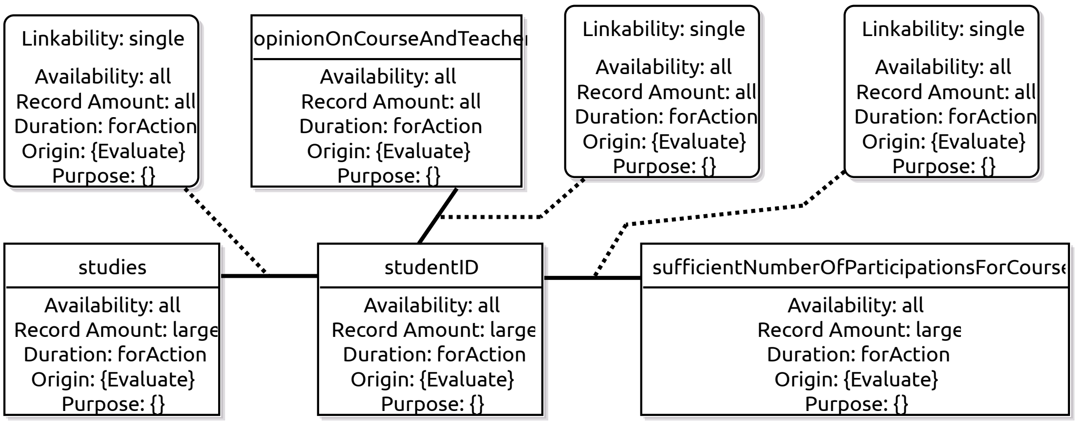


Figure 19.17.: *Available data diagram for the designed domain* Evaluations

linkable to each other due to functional requirement Participate. The link to the studentID is needed to know which student participated when in which course, because the personal data participationDateAndCourse themselves do not contain a link to the corresponding student. For the sake of simplicity, I decided that the studentID contained in the personal data studentData, which is managed by the study management system, is used. A more privacy-friendly option would be to generate identifiers that are only used by the course evaluation system and not (directly) linkable to the personal data studentData. All personal data and links are available to all possible instances of the lexical domain Participations, and they shall only be available there as long as it is necessary for the purpose of the course evaluation. That is, the participations of a student shall be deleted when he or she evaluated a course. At the domain Participations, all personal data participationDateAndCourse that are processed by the system shall be available, and potentially a large number of instances of the personal data studentID. All these personal data available at the lexical domain shall be linkable to each other. The purpose of all personal data and links is empty, because the personal data available at the domain Participations do not flow themselves to another domain, but only the information that a student sufficiently often participated in a course (see sufficientNumberOfParticipationsForCourse in Figure 19.14).

Figure 19.17 shows the available data diagram for the lexical domain Evaluations. An evaluation consists of the opinionOnCourseAndTeacher of the evaluating student, his or her studies, and (implicitly) the information that he or she has a sufficientNumberOfParticipationsForCourse. All these data are linkable to the student's studentID to ensure that a student only evaluates a course once and that the evaluation data are accurate. The student's studies are added to the evaluation to give the teacher information on the different studies the participating students are enrolled in and to possibly group the feedback of students by their studies. The latter shall only be considered in the aggregated evaluation results if the smallest group of students enrolled in the same study has at least a medium size to ensure that the aggregated evaluations are at

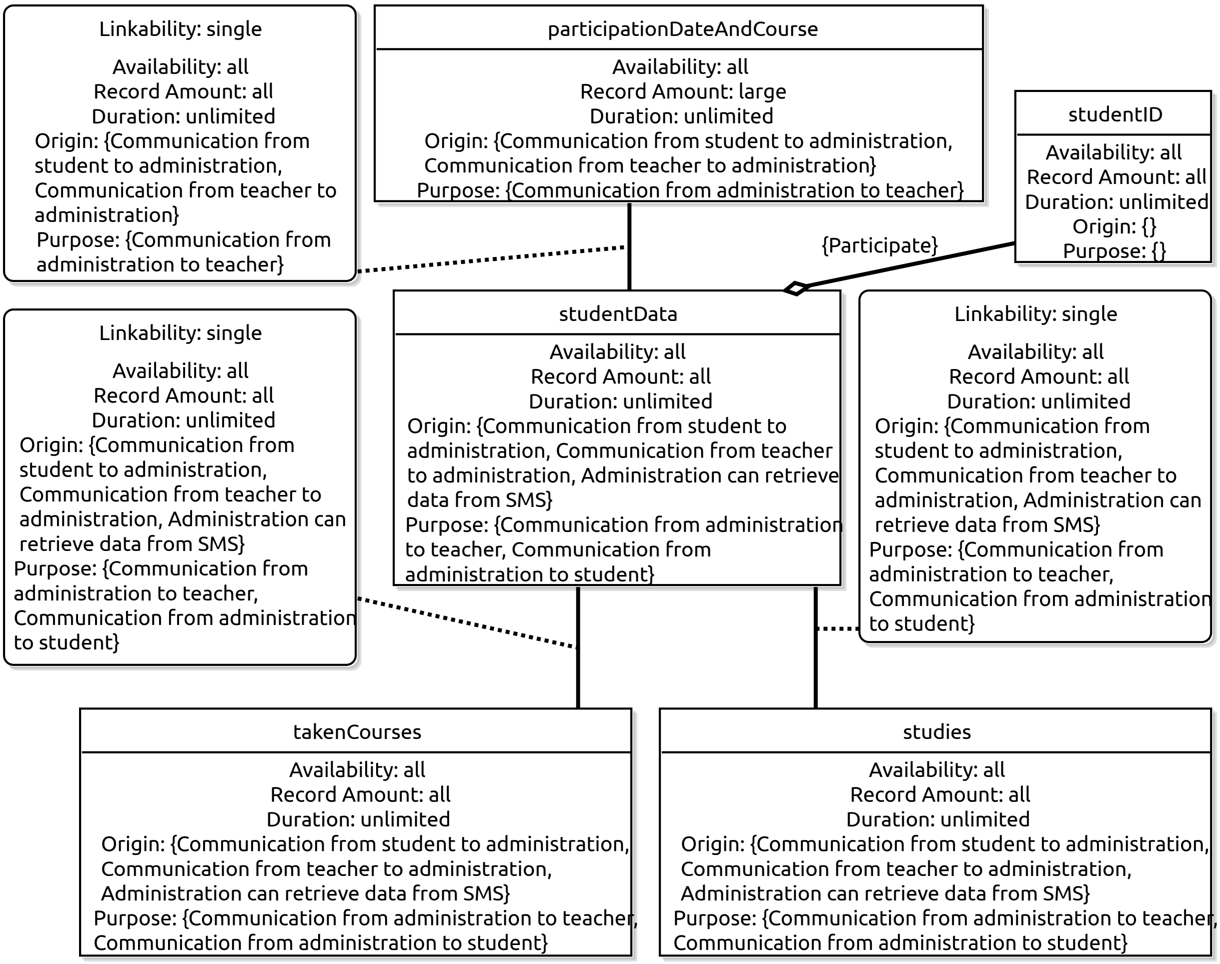

Linkability: single
Availability: all
Record Amount: all
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration}
Purpose: {Communication from administration to teacher}
participationDateAndCourse
Availability: all
Record Amount: large
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration}
Purpose: {Communication from administration to teacher}
studentID
Availability: all
Record Amount: all
Duration: unlimited
Origin: {}
Purpose: {}
{Participate}
Linkability: single
Availability: all
Record Amount: all
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration, Administration can retrieve data from SMS}
Purpose: {Communication from administration to teacher, Communication from administration to student}
studentData
Availability: all
Record Amount: all
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration, Administration can retrieve data from SMS}
Purpose: {Communication from administration to teacher, Communication from administration to student}
Linkability: single
Availability: all
Record Amount: all
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration, Administration can retrieve data from SMS}
Purpose: {Communication from administration to teacher, Communication from administration to student}
takenCourses
Availability: all
Record Amount: all
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration, Administration can retrieve data from SMS}
Purpose: {Communication from administration to teacher, Communication from administration to student}
studies
Availability: all
Record Amount: all
Duration: unlimited
Origin: {Communication from student to administration, Communication from teacher to administration, Administration can retrieve data from SMS}
Purpose: {Communication from administration to teacher, Communication from administration to student}

Figure 19.18.: *Available data diagram for the biddable domain Administration*

most linkable to a medium-sized group of students. All data available at the lexical domain Evaluations flow there because of functional requirement Evaluate. The personal data and the links between them shall be available to all instances of the domain Evaluations for the duration they are needed there. That is, after the evaluation report for a course was generated, the corresponding evaluations shall be deleted. All opinionOnCourseAndTeacher processed by the system shall be available at the domain Evaluations, and potentially a large number of instances of the personal data studies, studentID, and sufficientNumberOfParticipationsForCourse. All personal data instances available at Evaluations shall be linkable to the corresponding instances of the personal data studentID. The purpose of all personal data and links is empty, because the personal data available at the domain Evaluations do not flow themselves to another domain, but the are used to create the aggregated evaluation results (see aggregatedEvaluations in Figures 19.14 and 19.16).

I present the available data diagram for the lexical domain EvaluationReports in Figure 19.16. The EvaluationReports shall only contain the aggregatedEvaluations that are created due to functional requirement Create Report and further provided to teachers due to the functional requirement Show Results. All aggregatedEvaluations processed by the system shall be available at all instances of the domain EvaluationReports until it is necessary to delete them.

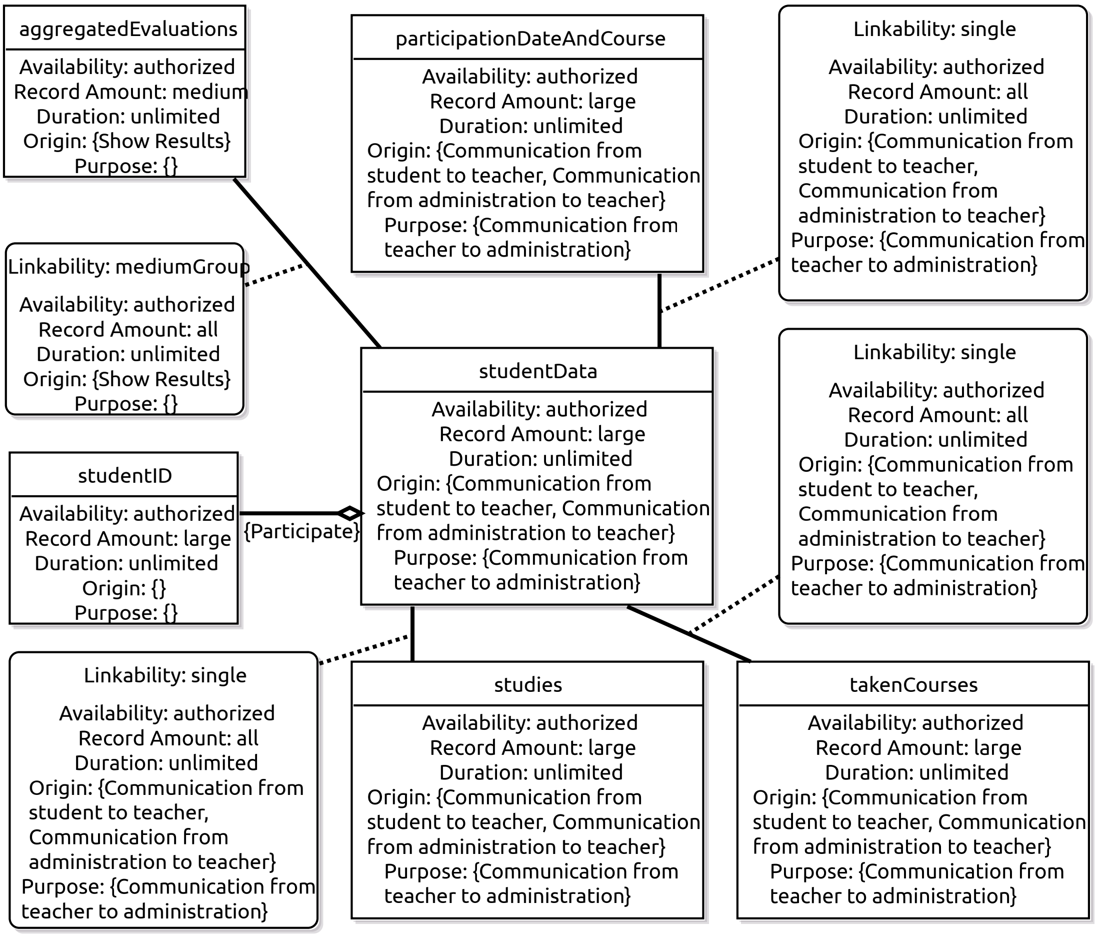

aggregatedEvaluations
Availability: authorized
Record Amount: medium
Duration: unlimited
Origin: {Show Results}
Purpose: {}
participationDateAndCourse
Availability: authorized
Record Amount: large
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}
Linkability: single
Availability: authorized
Record Amount: all
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}
Linkability: mediumGroup
Availability: authorized
Record Amount: all
Duration: unlimited
Origin: {Show Results}
Purpose: {}
studentData
Availability: authorized
Record Amount: large
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}
Linkability: single
Availability: authorized
Record Amount: all
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}
studentID
Availability: authorized
Record Amount: large
Duration: unlimited
Origin: {}
Purpose: {}
{Participate}
Linkability: single
Availability: authorized
Record Amount: all
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}
studies
Availability: authorized
Record Amount: large
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}
takenCourses
Availability: authorized
Record Amount: large
Duration: unlimited
Origin: {Communication from student to teacher, Communication from administration to teacher}
Purpose: {Communication from teacher to administration}

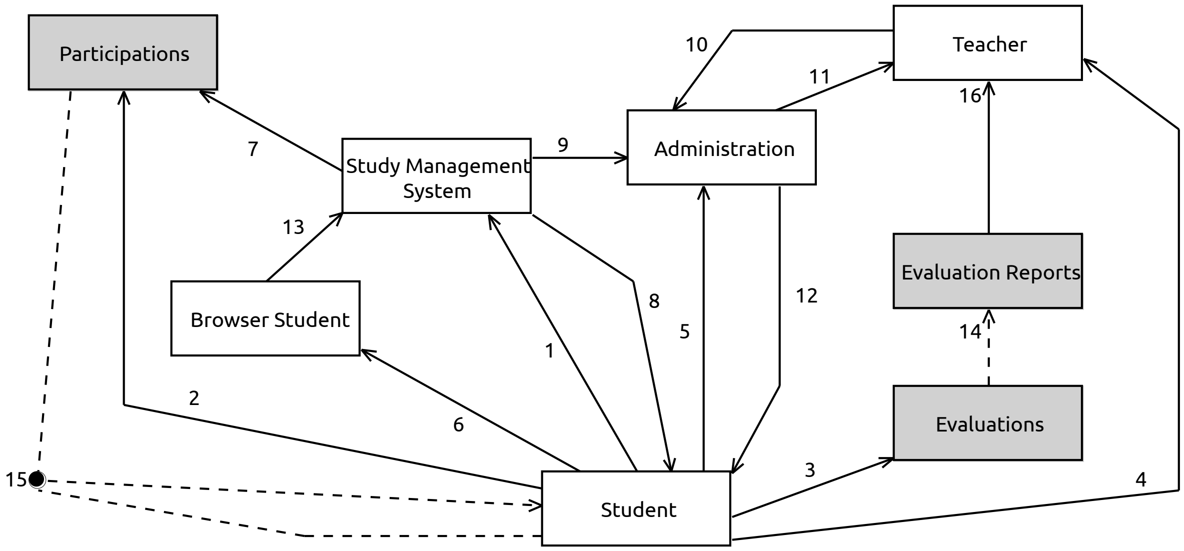
Figure 19.19.: *Available data diagram for the biddable domain* Teacher

19.4.2.2. Biddable Domains

At the individual instances of the biddable domain Student, all personal data related to them and listed in Figure 19.14, excluding the aggregatedEvaluations, are available. This is, because the aggregatedEvaluations are only provided to the teachers of the respective courses. The individual students are of cause able to link all personal data about them to each other. As for all biddable domains, I expect that the personal data and links available to them are available to them for an unlimited duration. As the available data diagram for students is similar to their personal data diagram shown in Figure 19.14, I omit it for the sake of simplicity.

No personal data of students shall be available to the biddable domain Attacker. Hence, the available data diagram for it is empty. The situations in which the Attacker unintendedly gets access to personal data are identified and evaluated during the Privacy Risks Analysis performed in Section 19.8.

The available data diagram for the biddable domain Administration is shown in Figure 19.18. All personal data available to the Administration are available there, because of the domain knowledge elicited during the step Privacy Context Elicitation (see Section 19.2), i.e. the communication with students and teachers, and the use of the student management system (SMS). From the student management system, the administration may retrieve all studentData, takenCourses, and studies processed by the system, and the links between these data. From the communication with students and teachers, the administration may get to know a large number of instances of the personal data participationDateAndCourse with the link to the corresponding studentData.



Labels

1: Data = {credentialsS, studentData, studies, takenCourses, usedDeviceAndDateS} Statements = {Authenticate, SMS stores data of students}
2: Data = {participationDateAndCourse, studentID} Statements = {Participate}
3: Data = {opinionOnCourseAndTeacher, studentID, studies, sufficientNumberOfParticipationsForCourse} Statements = {Evaluate}
4: Data = {participationDateAndCourse, studentData, studies, takenCourses} Statements = {Communication from student t teacher}
5: Data = {participationDateAndCourse, studentData, studies, takenCourses} Statements = {Communication from student t administration}
6: Data = {credentialsS, opinionOnCourseAndTeacher, participationDateAndCourse, studentData, studies, takenCourse usedDeviceAndDateS} Statements = {Browser student refines phenomena of student}
7: Data = {studentID} Statements = {Participate}
8: Data = {credentialsS, studentData, studies, takenCourses} Statements = {Students can retrieve data from SMS
9: Data = {studentData, studies, takenCourses} Statements = {Administration can retrieve data from SMS
10: Data = {participationDateAndCourse, studentData, studies, takenCourses} Statements = {Communication from teacher to administration}
11: Data = {participationDateAndCourse, studentData, studies, takenCourses} Statements = {Communication from administration to teacher}
12: Data = {studentData, studies, takenCourses} Statements = {Communication from administration to student}
13: Data = {credentialsS, studentData, studies, takenCourses, usedDeviceAndDateS} Statements = {Effect of FmanageStudies}
14: Data = {aggregatedEvaluations} Statements = {Create Report}
15: Data = {sufficientNumberOfParticipationsForCourse} Statements = {Request Evaluation}
16: Data = {aggregatedEvaluations} Statements = {Show Results}

Figure 19.20.: *Final data flow graph for the data subject student*

Figure 19.19 shows the available data diagram for the biddable domain Teacher. The personal data participationDateAndCourse, studentData, studies, and takenCourses may be available at authorized teachers, i.e., the teachers of the courses to which these data are related, in a large amount due to the communication with students and the administration, including the links between these data. Additionally, the personal data aggregatedEvaluations may be available to authorized teachers in a medium amount, i.e., teachers shall only access the evaluation results of their courses. A teacher may be able to link the aggregatedEvaluations to a medium-sized group of students of whom the aggregated evaluation results are, i.e., the students that regularly participated in the course.

19.4.3. Final Data Flow Graph

The final data flow graph for the data subject Student is shown in Figure 19.20. It provides the overview of the intended data flows in the course evaluation system and refines the initial data flow graph shown in Figure 19.13.

The personal data flows annotated with the numbers 4, 5, 8, 9, 10, 11, 12, and 13 all exist in the environment of the course evaluation system independently of the existence of the machine.

Due to the functional requirement Authenticate the personal data credentialsS and usedDeviceAndDateS flow from the Student to the Study Management System (see flow number 1). The other personal data annotated at this edge flow due to the students' usage of the study management system. As the interfaces between the course evaluation system and the study management system are refined by the Browser Student, all personal data that flow to these systems also flow to the domain Browser Student (see data flow 6).

From requirement Participate, I identified that the personal data participationDateAndCourse, and studentID flow from the data subject Student to the designed domain Participations (data flow edge 2; see also Figure 19.15), and that the studentID flows from the Study Management System to the designed domain Participations (data flow edge 7). The studentID is considered to flow from both, the Student and the Study Management System, because the students provide it (implicitly due to their previous authentication) and it is matched with the studentID available in the Study Management System.

The data flow edge 3 documents that due to requirement Evaluate, the personal data opinionOnCourseAndTeacher, studentID, studies, and sufficientNumberOfParticipationsForCourse flow from the Student to the designed domain Evaluations (see also Figure 19.17).

The derived data flow edge 14 documents that the aggregatedEvaluations that are derived from the studies and opinionOnCourseAndTeacher (cf. Figure 19.20) flow due to requirement Create Report to the designed domain Evaluation Reports (see also Figure 19.17). The data flow edge 16 documents that the aggregatedEvaluations further flow from the Evaluation Reports to the Teacher due to functional requirement Show Results.

Due to functional requirement Request Evaluation, the student gets the information that he or she has a sufficientNumberOfParticipationsForCourse (see derived data flow edge 15) based on the personal data provided by the student him- or herself and the personal data participationDateAndCourse available at the designed domain Participations (see also Figure 19.14).

19.5. Privacy Requirements Identification

Based on the identified flows of personal data, the ProPAn tool is able to generate the privacy requirements for the course evaluation system, which I then adjust. That is, I apply the step Privacy Requirements Identification that I introduced in Chapter 12 on the course evaluation system.

In the following sections, I present the automatically generated privacy requirements, and I discuss how I adjusted them for the different categories of privacy requirements of my taxonomy (cf. Chapter 7). Section 19.5.1 presents the integrity and availability requirements for the course evaluation system. I present all privacy requirements that are refinements of the abstract class *ConfidentialityRequirement* (including unlinkability-related requirements) in Section 19.5.2. The transparency and intervenability requirements identified for the course evaluation system are presented in Sections 19.5.3 and 19.5.4, respectively. Section 19.5.5 discusses the validation of the adjusted privacy requirements.

19.5.1. Integrity and Availability Requirements

As I explained in Section 12.2.1.2, the ProPAn tool generates for the personal data processed by the machine the integrity requirement SIS, and the availability requirement SAS. Both are generated for the personal data participationDateAndCourse, opinionOnCourseAndTeacher, studies, studentID, sufficientNumberOfParticipationsForCourse, and aggregatedEvaluations that are processed by the machine, and for the counterstakeholders Student, Administration, Teacher, and Attacker that shall not be able to negatively influence the availability and integrity of the personal data.

19.5.2. Confidentiality Requirements

I present the generated and adjusted confidentiality requirements using tables that list the attribute values of the confidentiality requirements. Note that I use the abstract type *ConfidentialityRequirement* in my privacy requirements taxonomy for confidentiality requirements about data (called data confidentiality requirements), and also for unlinkability-related requirements, as these are all concerned with keeping the existence of personal data, or links between personal data and the data subject confidential (see also Section 7.2.3). The presented tables can automatically be generated from the ProPAn model by the ProPAn tool.

In the following sections, I present the confidentiality requirements for the four counterstakeholders administration (see Section 19.5.2.1), teacher (see Section 19.5.2.2), student (see Section 19.5.2.3), and attacker (see Section 19.5.2.4). In each of these sections, I first present the automatically generated confidentiality requirements, and then I discuss how I adjusted them.

19.5.2.1. Counterstakeholder Administration

19.5.2.1.1. Automatically Generated Confidentiality Requirements Table 19.10 shows the confidentiality requirements for the counterstakeholder Administration that the ProPAn tool generated from the elicited data flows. The undetectability, data confidentiality, anonymity, and data unlinkability requirements listed in the first four rows reflect that all instances of the biddable domain Administration are allowed to detect, know, and link the personal data studentID, studies, takenCourses, studentData, and participationDateAndCourse to the individual student they belong to and to each other, respectively. The last row contains the undetectability requirement UUS3A that states that the personal data opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, sufficientNumberOfParticipationsForCourse, and aggregatedEvaluations shall not be detectable for any instance of the biddable domain Administration.

19.5.2.1.2. Adjusted Confidentiality Requirements The latter undetectability requirement is too strong in the context of the course evaluation system, because the administration may be allowed to know that these data exist, but they shall not be allowed to access the students' opinionOnCourseAndTeacher, credentialsS, and usedDeviceAndDateS. For the personal data sufficientNumberOfParticipationsForCourse, and aggregatedEvaluations even a data confidentiality requirement may be too strong. This is, because the administration may know whether a student sufficiently often participated in a course to evaluate it, because it is documented in the data confidentiality requirement SDS2A that the administration is allowed to know about the participation of students in a course, and the number of participations that are necessary to evaluate a course may also be known by the administration. To document this, I added the administration to the domains that are able to derive the personal data sufficientNumberOfParticipationsForCourse from the personal data participationDateAndCourse available to them (cf. Figure 11.6 on page 182 and Figure 19.14). Consequently, the personal data sufficientNumberOfParticipationsForCourse are available at the biddable domain Administration. The aggregatedEvaluations could also be considered to be made public and hence, not be considered as confidential personal data

Table 19.10.: *Automatically generated confidentiality requirements for the data subject* Student *and the counterstakeholder* Administration

Name	Type	Avail.	Linkab.	Personal data
UUS2A	Undetectability requirement	all		studies, takenCourses, studentData, participationDateAndCourse, studentID
SDS2A	Data confidentiality requirement	all		studies, takenCourses, studentData, participationDateAndCourse, studentID
UAS02A	Anonymity requirement	all	single	studentID, studies, takenCourses, studentData, participationDateAndCourse
UDS02A	Data unlinkability requirement	all	single	participationDateAndCourse, takenCourses, studies, studentData, studentID
UUS3A	Undetectability requirement	none		opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, sufficientNumberOfParticipationsForCourse, aggregatedEvaluations

Table 19.11.: *Adjusted confidentiality requirements for the data subject* Student *and the counterstakeholder* Administration

Name	Type	Avail.	Linkab.	Personal data
UUS2A	Undetectability requirement	all		studies, takenCourses, studentData, participationDateAndCourse, studentID, sufficientNumberOfParticipationsForCourse, opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, aggregatedEvaluations
SDS2A	Data confidentiality requirement	all		studies, takenCourses, studentData, participationDateAndCourse, studentID, sufficientNumberOfParticipationsForCourse
UAS02A	Anonymity requirement	all	single	studentID, studies, takenCourses, studentData, participationDateAndCourse, sufficientNumberOfParticipationsForCourse
UDS02A	Data unlinkability requirement	all	single	participationDateAndCourse, takenCourses, studies, studentData, studentID, sufficientNumberOfParticipationsForCourse
SDS3A	Data confidentiality requirement	none		opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, aggregatedEvaluations

for the administration. Note that the aggregated evaluations are also considered as personal data of teachers and that from the point-of-view of teachers the aggregated evaluations should be kept confidential. However, I decided that the aggregatedEvaluations shall only be made available to the teachers of the respective course and I left our the consideration of the data subject teacher. For the sake of simplicity, I also do not consider a specific repudiation type for the identified confidentiality requirements. Hence, I omit the repudiation type in the respective tables. Table 19.11 shows the adjusted confidentiality requirements obtained following the above discussed considerations.

Table 19.12.: *Automatically generated confidentiality requirements for the data subject* Student *and the counterstakeholder* Teacher

Name	Type	Availab.	Linkab.	Personal data	Rep.
UUS1T	Undetectability requirement	authorized		studies, takenCourses, studentData, participationDateAndCourse, studentID, aggregatedEvaluations	none
SDS1T	Data confidentiality requirement	authorized		studies, takenCourses, studentData, participationDateAndCourse, studentID, aggregatedEvaluations	none
UAS01T	Anonymity requirement	authorized	single	studentID, studies, takenCourses, studentData, participationDateAndCourse	none
UDS01T	Data unlinkability requirement	authorized	single	participationDateAndCourse, takenCourses, studies, studentData, studentID	none
UAS21T	Anonymity requirement	authorized	mediumGroup	aggregatedEvaluations	none
UDS21T	Data unlinkability requirement	authorized	mediumGroup	studentData, aggregatedEvaluations, takenCourses, studentID, studies, participationDateAndCourse	none
UUS3T	Undetectability requirement	none		opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, sufficientNumberOfParticipationsForCourse	none

19.5.2.2. Counterstakeholder Teacher

19.5.2.2.1. Automatically Generated Confidentiality Requirements The automatically generated confidentiality requirements for the counterstakeholder Teacher are shown in Table 19.12. The first two rows document the undetectability and data confidentiality requirement that specify that only authorized teachers shall be able to detect and know the personal data studies, takenCourses, studentData, participationDateAndCourse, studentID, and aggregatedEvaluations. The following two anonymity and data unlinkability requirements document that authorized teachers are allowed to link the personal data studies, takenCourses, studentData, participationDateAndCourse, and studentID to each other and to the student they belong to. In contrast, the anonymity requirement UAS21T and the data unlinkability requirement UDS21T state that authorized students shall only be able to link the personal data aggregatedEvaluations to a medium-sized group of students, and to a medium-sized number of instances of the personal data studentData, takenCourses, studentID, studies, and participationDateAndCourse. The last row contains the undetectability requirement UUS3T, which documents that the personal data opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, sufficientNumberOfParticipationsForCourse shall not be detectable for teachers.

Table 19.13.: *Adjusted confidentiality requirements for the data subject* Student *and the counterstakeholder* Teacher

Name	Type	Availab.	Linkab.	Personal data	Rep.
UUS2T	Undetectability requirement	all		studies, takenCourses, studentData, participationDateAndCourse, studentID, aggregatedEvaluations, sufficientNumberOfParticipationsForCourse, opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS	none
SDS1T	Data confidentiality requirement	authorized		studies, takenCourses, studentData, participationDateAndCourse, studentID, aggregatedEvaluations, sufficientNumberOfParticipationsForCourse	none
UAS01T	Anonymity requirement	authorized	single	studentID, studies, takenCourses, studentData, participationDateAndCourse, sufficientNumberOfParticipationsForCourse	none
UDS01T	Data unlinkability requirement	authorized	single	participationDateAndCourse, takenCourses, studies, studentData, studentID, sufficientNumberOfParticipationsForCourse	none
UAS21T	Anonymity requirement	authorized	mediumGroup	aggregatedEvaluations	none
UDS21T	Data unlinkability requirement	authorized	mediumGroup	studentData, aggregatedEvaluations, takenCourses, studentID, studies, participationDateAndCourse, sufficientNumberOfParticipationsForCourse	none
SDS3T	Data confidentiality requirement	none		opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS	none

19.5.2.2.2. Adjusted Confidentiality Requirements As all teachers may know about the existence of the personal data of students, I introduce the undetectability requirement UUS2T shown in the first row of Table 19.13 that replaces the generated undetectability requirements UUS1T and UUS3T. For the generated undetectability requirement UUS3T, I created the data confidentiality requirement SDS3T, because teachers are allowed to know that the personal data opinionOnCourseAndTeacher, credentialsS, usedDeviceAndDateS, and aggregatedEvaluations exists, but they shall not be able to access them. Note that similar to the administration also teachers may know the personal data sufficientNumberOfParticipationsForCourse of students, because the teachers are allowed to know the personal data participationDateAndCourse. As the threshold for sufficiently many participations in a course to evaluate may be known to teachers, they are also added to the domains that may be able to derive this information. Hence, I added the personal data sufficientNumberOfParticipationsForCourse to the confidentiality requirements SDS1T, UAS01T, UDS01T, and UDS21T in Table 19.13.

19.5.2.3. Counterstakeholder Student

19.5.2.3.1. Automatically Generated Confidentiality Requirements Table 19.14 shows the generated confidentiality requirements for the counterstakeholder Student. The first four rows show the undetectability, data confidentiality, anonymity, and data unlinkability requirements that specify that students are allowed to detect, know, and link the personal data participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse of themselves to themselves. These confidentiality requirements also specify that students are not allowed to detect, know, and link the personal data of other students to these or to each other (see also the semantics for confidentiality requirements given in Sections 7.2.2.2 and 7.2.3.2). The last row shows an undetectability requirement that specifies that the existence of the aggregatedEvaluations shall not be detectable for students.

19.5.2.3.2. Adjusted Confidentiality Requirements I weaken the generated undetectability requirements as shown in Table 19.15. I changed the availability of the undetectability requirement UUS0S to all, because all students may know that the respective personal data also exist for other students. In this way, I obtained the undetectability requirement UUS2A. Furthermore, I translated the undetectability requirement UUS3S to the the data confidentiality requirement SDS3S and I added the personal data aggregatedEvaluations to the undetectability requirement UUS2S to document that students are allowed to know that aggregatedEvaluations exist, but that they shall not be able to access them.

19.5.2.4. Counterstakeholder Attacker

19.5.2.4.1. Automatically Generated Confidentiality Requirements The generated confidentiality requirements for the counterstakeholder Attacker are shown in Table 19.16. For the attacker only the undetectability requirement UUS3Att is generated that documents that no personal data of students shall be detectable for him or her.

19.5.2.4.2. Adjusted Confidentiality Requirements The automatically generated undetectability requirement is again too strong, and I weaken it to the data confidentiality requirement listed in Table 19.17, because attackers may know about the existence of the personal data, but they are not allowed to access the data.

19.5.3. Transparency-related Requirements

I present the transparency requirements for the course evaluation system in the following sections. Section 19.5.3.1 presents the collection information requirements, Section 19.5.3.2 the storage information requirements, Section 19.5.3.3 the flow information requirements, Section 19.5.3.4 the exceptional information requirements, and Section 19.5.3.5 the presentation requirements. In each of these sections, I first present the automatically generated transparency requirements, and then I discuss how I adjusted them.

Table 19.14.: *Automatically generated confidentiality requirements for the data subject* Student *and the counterstakeholder* Student

Name	Type	Availab.	Linkab.	Personal data	Rep.
UUS0S	Undetectability requirement	individual		participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse	none
SDS0S	Data confidentiality requirement	individual		participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse	none
UAS00S	Anonymity requirement	individual	single	opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, credentialsS, usedDeviceAndDateS, studentID, studies, takenCourses, studentData, participationDateAndCourse	none
UDS00S	Data unlinkability requirement	individual	single	takenCourses, opinionOnCourseAndTeacher, credentialsS, sufficientNumberOfParticipationsForCourse, studies, participationDateAndCourse, studentData, usedDeviceAndDateS, studentID	none
UUS3S	Undetectability requirement	none		aggregatedEvaluations	none

19.5.3.1. Collection Information Requirements

19.5.3.1.1. Automatically Generated Collection Information Requirements The generated collection information requirements for the data subject Student are presented in Table 19.18. Table 19.18 shows which personal data are collected from the students, due to which statements this collection happens (origin), for which purpose the data are collected, and how the data are collected from the students (method). I omitted the values of the attributes mandatory, grounds, and controlOptions in Table 19.18, because all collection information requirements have the attribute mandatory set to true, the attribute grounds to a singleton set with the element consent, and the attribute controlOptions is set to the intervenability requirements generated for the respective collection information requirement allowing to doNotConsent, withdrawConsent, object, and requestDataCopy. Note that the purpose is empty for all generated collection information requirements, because the respective personal data are not further provided to other domains. The purposes listed in *italics* in Table 19.18 are added during the adjustment of the generated

Table 19.15.: *Adjusted confidentiality requirements for the data subject* Student *and the counterstakeholder* Student

Name	Type	Availab.	Linkab.	Personal data	Rep.
UUS2S	Undetectability requirement	all		participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse, aggregatedEvaluations	none
SDS0S	Data confidentiality requirement	individual		participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse	none
UAS00S	Anonymity requirement	individual	single	opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, credentialsS, usedDeviceAndDateS, studentID, studies, takenCourses, studentData, participationDateAndCourse	none
UDS00S	Data unlinkability requirement	individual	single	takenCourses, opinionOnCourseAndTeacher, credentialsS, sufficientNumberOfParticipationsForCourse, studies, participationDateAndCourse, studentData, usedDeviceAndDateS, studentID	none
SDS3S	Data confidentiality requirement	none		aggregatedEvaluations	none

collection information requirements.

19.5.3.1.2. Adjusted Collection Information Requirements As I described in Section 12.3.3, not all attributes of processing information requirements can automatically be set by the ProPAn tool. I add as controller to all transparency requirements the biddable domain Administration, because the administration shall be responsible for the processing of personal data in the context of the course evaluation system. Additionally, I documented reasons why the personal data are needed to be processed, but I do not show these reasons for the sake of simplicity. I added the purposes given in *italics* in Table 19.18 to the collection information requirements. These added statements are the functional requirements due to which other personal data are derived from the collected personal data.

Table 19.16.: *Automatically generated confidentiality requirements for the data subject* Student *and the counterstakeholder* Attacker

Name	Type	Availab.	Personal data	Rep.
UUS3Att	Undetectability requirement	none	participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse, aggregatedEvaluations	none

Table 19.17.: *Adjusted confidentiality requirements for the data subject* Student *and the counterstakeholder* Attacker

Name	Type	Availab.	Personal data	Rep.
UUS2Att	Undetectability requirement	all	participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse, aggregatedEvaluations	none
SDS3Att	Data confidentiality requirement	none	participationDateAndCourse, takenCourses, opinionOnCourseAndTeacher, credentialsS, studies, studentData, usedDeviceAndDateS, studentID, sufficientNumberOfParticipationsForCourse, aggregatedEvaluations	none

Table 19.18.: *Collection information requirements for the data subject* Student

Name	Personal data	Origin	Purpose	Method
TCSparticipationDateAndCourse	participationDateAndCourse	Participate	*Request Evaluation, Evaluate*	direct
TCSstudentID	studentID	Participate, Evaluate		reused
TCSopinionOnCourseAndTeacher	opinionOnCourseAndTeacher	Evaluate	*Create Report*	direct
TCSsufficientNumberOfParticipationsForCourse	sufficientNumberOfParticipationsForCourse	Evaluate		derived
TCSstudies	studies	Evaluate	*Create Report*	reused

19.5.3.2. Storage Information Requirements

19.5.3.2.1. Automatically Generated Storage Information Requirements Table 19.19 shows the generated storage information requirements for the data subject Student. More precisely, it shows the personal data with which the storage information requirements are concerned, the origin from which it was identified that these are stored, the purpose for which the personal data need to be stored, and how long these have to be retained. As for the generated collection information requirements, I omit in Table 19.19 the attributes that are all instantiated with the

Table 19.19.: *Storage information requirements for the data subject* Student

Name	Personal data	Origin	Purpose	Retention
TSSparticipation-DateAndCourse	participation-DateAndCourse	Participate	*Request Evaluation, Evaluate*	forAction
TSSopinionOn-CourseAndTeacher	opinionOn-CourseAndTeacher	Evaluate	*Create Report*	forAction
TSSstudies	studies	Evaluate	*Create Report*	forAction
TSSstudentID	studentID	Participate, Evaluate		forAction
TSSsufficientNumberOf-ParticipationsForCourse	sufficientNumberOf-ParticipationsForCourse	Evaluate		forAction
TSSaggregated-Evaluations	aggregatedEvaluations	Create Report	Show Results	untilDeleted

same values. The attributes mandatory, grounds, and controlOptions are set to true, the singleton set containing consent, and the intervenability requirements generated for the respective storage information requirement allowing to doNotConsent, withdrawConsent, object, review, challengeAccuracy, and challengeCompleteness, respectively. Note that the purposes in Table 19.19 that are printed in *italics* are added during the adjustment of the storage information requirements.

19.5.3.2.2. Adjusted Storage Information Requirements I added the purposes given in *italics* in Table 19.19 to the storage information requirements. These added statements are the functional requirements due to which other personal data are derived from the stored personal data. There is no further need to modify the generated storage information requirements in addition to the adjustments that apply to all processing information requirements that I mentioned in Section 19.5.3.1.

19.5.3.3. Flow Information Requirements

19.5.3.3.1. Automatically Generated Flow Information Requirements The automatically generated flow information requirements for the data subject student are listed in Table 19.20. These document which personal data flow to which domains (target), to which concrete targets the personal data shall be available (availability), due to which statements (origin) the data flow, and for which purpose the data need to be available at the target.

19.5.3.3.2. Adjusted Flow Information Requirements In addition to the adjustments already mentioned for collection information requirements, I completed the attribute countries of the flow information requirements with the country Germany. This is, because the course evaluation system shall be used and deployed at the University Duisburg-Essen located in Germany (not shown in Table 19.20).

19.5.3.4. Exceptional Information Requirements

19.5.3.4.1. Automatically Generated Exceptional Information Requirements The ProPAn tool automatically generates for the personal data participationDateAndCourse, opinionOnCourseAndTeacher, studies, studentID, sufficientNumberOfParticipationsForCourse, and aggregatedEvalu-

Table 19.20.: *Flow information requirements for the data subject* Student

Name	Personal data	Origin	Purpose	Target	Availability
TFSsufficient-NumberOf-Participations-ForCourse0S	sufficient-NumberOf-Participations-ForCourse	Request Evaluation	Evaluate	Student	individual
TFScredentials-S2SMS	credentialsS	Authenticate	Students can retrieve data from SMS	Study Management System	all
TFSaggregated-Evaluations1T	aggregated-Evaluations	Show Results		Teacher	authorized

Table 19.21.: *Exceptional information requirements for the data subject* Student

Name	Case	Intervention types
TES0	dataBreach	
TES1	systemChange	
TES2	nonCompliance	suspendDataFlows, orderBanOfProcessing, orderErasure, orderRectification
TES3	authorityRequest	obtainAccess

ations processed by the machine of the course evaluation system the exceptional information requirements shown in Table 19.21 with the corresponding authority intervention requirements.

19.5.3.4.2. Adjusted Exceptional Information Requirements To complete the generated exceptional information requirements, I add as controller the biddable domain Administration, and the *Landesbeauftragte für Datenschutz und Informationsfreiheit Nordrhein-Westfalen* (LDI.NRW) as responsible data protection authority for North Rhine-Westphalia (not shown in Table 19.21).

19.5.3.5. Presentation Requirements

For each transparency requirement, the ProPAn tool generates a presentation requirement. For the collection, storage, and flow information requirements, I set the accessibility to forwarded and onRequest. That is, the students are actively informed about the processing of their personal data, and additionally students can request this information. The languages are set to German and English, because the University Duisburg-Essen is located in Germany and has many international students, who shall also be able to use the course evaluation system. I set the time when students shall be informed about the processing of their personal data to beforeCollection.

For the exceptional information requirements, I also set the accessibility to forwarded and onRequest, and the languages to German and English. As time when students and authorities shall be informed about exceptional cases, I select afterRecognition and onRequest.

19.5.4. Intervenability-related Requirements

In section Section 19.5.4.1, I present the generated and adjusted data subject intervention requirements for the course evaluation system. The authority intervention requirements and intervention information requirements are briefly discussed in Sections 19.5.4.2 and 19.5.4.3, respectively.

Table 19.22.: *Data subject intervention requirements for the data subject* Student

Name	Transpar. Req.	Time	Type	Effect
IDTC⟨personal data⟩01	TC⟨personal data⟩	beforeCollection	doNotConsent	noProcessing
IDTC⟨personal data⟩15	TC⟨personal data⟩	anyTime	withdrawConsent	erasure
IDTC⟨personal data⟩52	TC⟨personal data⟩	anyTime	object	restrictedProcessing, erasure
IDTC⟨personal data⟩66	TC⟨personal data⟩	anyTime	requestDataCopy	dataCopy
IDTS⟨personal data⟩01	TS⟨personal data⟩	atRecording	doNotConsent	noProcessing
IDTS⟨personal data⟩15	TS⟨personal data⟩	anyTime	withdrawConsent	erasure
IDTS⟨personal data⟩52	TS⟨personal data⟩	anyTime	object	restrictedProcessing, erasure
IDTS⟨personal data⟩20	TS⟨personal data⟩	anyTime	review	access
IDTS⟨personal data⟩34	TS⟨personal data⟩	anyTime	challengeAccuracy	correction
IDTS⟨personal data⟩43	TS⟨personal data⟩	anyTime	challengeCompleteness	amendment
IDTF⟨personal data⟩01	TF⟨personal data⟩	beforeTransmission	doNotConsent	noProcessing
IDTF⟨personal data⟩15	TF⟨personal data⟩	anyTime	withdrawConsent	erasure
IDTF⟨personal data⟩52	TF⟨personal data⟩	anyTime	object	restrictedProcessing, erasure
IDTF⟨personal data⟩20	TF⟨personal data⟩	anyTime	review	access
IDTF⟨personal data⟩34	TF⟨personal data⟩	anyTime	challengeAccuracy	correction
IDTF⟨personal data⟩43	TF⟨personal data⟩	anyTime	challengeCompleteness	amendment

19.5.4.1. Data Subject Intervention Requirements

19.5.4.1.1. Automatically Generated Data Subject Intervention Requirements Table 19.22 shows which data subject intervention requirements are generated with which attribute values for the previously introduced collection, storage, and flow information requirements (cf. Section 12.2.4). The placeholder ⟨personal data⟩ represents the personal data of the respective transparency requirement for which the data subject intervention requirement is generated.

19.5.4.1.2. Adjusted Data Subject Intervention Requirements As the storage and flow behavior of the course evaluation system is already known before the needed personal data of students is collected, I changed the time for all storage and flow information requirements to beforeCollection. That is, the students already have to give consent to the storage of their personal data and the further flow of these at the time when the personal data (or the data from which these are derived) are collected.

I consider the right to data portability to be out of the scope of the course evaluation system, because there shall not be other course evaluation systems to evaluate the courses given at the University Duisburg-Essen to which the stored participations and evaluations could be transferred. Hence, I delete the respective data subject intervention requirements IDTC⟨personal data⟩66 from the ProPAn model.

The data aggregatedEvaluations is derived based on the students' opinionOnCourseAndTeacher

Table 19.23.: *Authority intervention requirements for the data subject* Student

Name	Transp. Req.	Type	Effect
IATES207	TES2	suspendDataFlows	suspendedDataFlows
IATES211	TES2	orderBanOfProcessing	noProcessing, restrictedProcessing
IATES221	TES2	orderErasure	noProcessing, erasure, amendment
IATES234	TES2	orderRectification	correction
IATES340	TES3	obtainAccess	access

and studies, and possibly contains these. However, from the aggregatedEvaluations only a medium-sized group of students to whom these may be related can be identified, and once the aggregatedEvaluations are generated, the related evaluations containing the students' opinionOnCourseAndTeacher and studies shall be deleted from the lexical domain Evaluations. Hence, the data aggregatedEvaluations do not allow to identify the respective data subjects and I consequently delete all data subject intervention requirements, except the one with type doNotConsent. That is, students can still not consent to the processing of their personal data by the course evaluation system before their data are collected, but once aggregatedEvaluations are created based on their personal data and consent, they can no longer withdraw this consent, or intervene in another way into the processing of the aggregatedEvaluations.

19.5.4.2. Authority Intervention Requirements

Table 19.23 lists the authority intervention requirements that are generated by the ProPAn tool for the exceptional information requirements (cf. Section 12.2.4). As the course evaluation system shall comply to the GDPR, I keep the authority intervention requirements as they were generated.

19.5.4.3. Intervention Information Requirements

For each data subject intervention requirement, the ProPAn tool also generates an intervention information requirement. As these have no further attributes, I do not show a representation of these.

19.5.5. Validation of the Adjusted Privacy Requirements

The validation of the privacy requirements shows that the validation condition VSA1 (see Section 12.4.2) is violated, because the personal data aggregatedEvaluations shall be available due to the availability requirement SAS, but data confidentiality requirement SDS3S states that the aggregatedEvaluations shall not be accessible for students. I resolve this conflict, by removing the personal data aggregatedEvaluations from the availability requirement SAS. This can be reasoned, because the aggregatedEvaluations are no longer linkable to the individual students (see also Section 19.5.4.1.2) and also contain personal data of other students.

The automatic evaluation of the other validation conditions does neither raise further errors for the adjusted privacy requirements, nor reveal inconsistencies between the privacy requirements.

19.6. Privacy Risk Analysis

In this section, I identify and evaluate the potential risks to the privacy requirements that I identified in the previous section. That is, I apply the step Privacy Risk Analysis described in Chapter 13 to the course evaluation system. In Section 19.6.1, I identify privacy threats from

Table 19.24.: *Mapping of the functional requirements of the course evaluation system to PRIOP's operation categories*

Requirement	Collection	Storage	Flow	Deduction
Authenticate			credentialsS	
Participate	participationDateAndCourse, studentID	participationDateAndCourse, studentID		
Request Evaluation			sufficientNumberOfParticipationsForCourse	sufficientNumberOfParticipationsForCourse
Evaluate	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, studies	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, studies		
Create Report		aggregatedEvaluations		aggregatedEvaluations
Show Results			aggregatedEvaluations	

deviations of the course evaluation system's normal behavior using PRIOP. I create a global model of the identified privacy threats in Section 19.6.2. Based on the global model, I evaluate the implied privacy risks in Section 19.6.3.

19.6.1. Identify Privacy Threats from Deviations

First, I identify potential privacy threats by considering deviations from the normal processing of personal data due to the functional requirements. Table 19.24 shows which functional requirements are concerned with what kind of processing and of what personal data. This information is automatically derived by the ProPAn tool from the information elicited during the Data Flow Analysis. Based on this information, I consider the PRIOP guide words and templates (cf. Table 13.2 on page 251 and Table 13.3 on page 252) for the respective requirements and categories.

The functional requirement Authenticate is concerned with the flow of the personal data credentialsS to the Study Management System to authenticate students who want to use the course evaluation system. The corresponding deviations, possible causes, consequences, and harmed privacy requirements that I identified are listed in Table 19.25.

The deviations that might occur when the personal data participationDateAndCourse and studentID are collected from the Student and the Study Management System for requirement Participate are listed in Table 19.26. The deviations, possible causes, consequences, and harmed privacy requirements that I identified for the storage of these data at the lexical domain Participations are listed in Table 19.27.

For requirement Request Evaluations the personal data sufficientNumberOfParticipationsForCourse are derived by the Course Evaluation System and provided to the Student. The deviations that I identified for the deduction of these personal data are given in Table 19.28, and the deviations for the flow to the student in Table 19.29.

The deviations that might occur when the personal data studentID, opinionOnCourseAndTeacher,

Table 19.25.: *Deviations for the flow of credentialsS to Study Management System in requirement Authenticate*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	credentialsS do not flow to SMS.	Software or network error prevents the flow of the credentials to the study management system.	The student cannot be authenticated and cannot access the course evaluation system.	SAS
INCORRECT	The credentialsS flowing to SMS are incorrect.	1. The student incidentally types in incorrect credentials, or a software or network error changes the provided credentials. 2. The student maliciously tries to authenticate as another student.	1. The student cannot be authenticated and cannot access the course evaluation system. 2. The student has access to the course evaluation system using another student's credentials.	1. SAS 2. SDS0S, UAS00S, UDS00S, TES0
OTHER THAN	credentialsS flow to Attacker.	The attacker may steal the student's credentials by attacking the network or the student's browser	The attacker has access to the course evaluation system using the stolen credentials.	SDS3Att, TES0
AFTER	credentialsS flow after another subsequent operation to SMS.	Course evaluation system allows to register a participation, request the evaluation, or perform an evaluation, without a prior authentication.	Participation and evaluation cannot correctly be assigned to the corresponding student.	SIS

sufficientNumberOfParticipationsForCourse, and studies are collected from the Student and the Study Management System for requirement Evaluate are listed in Tables 19.30 and 19.31. The deviations, possible causes, consequences, and harmed privacy requirements that I identified for the storage of these data at the lexical domain Evaluations are listed in Table 19.32.

For requirement Create Report the personal data aggregatedEvaluations are derived by the Course Evaluation System and stored in the lexical domain Evaluation Reports. The deviations that I identified for the deduction of these personal data are given in Table 19.33, and the deviations for the storage at the lexical domain Evaluation Reports in Table 19.34.

I list the identified deviations, possible causes, consequences, and harmed privacy requirements for the flow of the personal data aggregatedEvaluations to the biddable domain Teacher due to requirement Show Results in Table 19.35.

19.6.2. Assess Privacy Threats Globally

The privacy threats that I identified from the above presented deviations of the system's normal behavior, have to be structured, and their interrelationships have to be documented to con-

Table 19.26.: *Deviations for the collection of participationDateAndCourse and studentID from Student and Study Management System in requirement Participate*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	participationDateAndCourse and studentID are not collected	The student cannot be authenticated and cannot access the course evaluation system.	Participation is not registered.	SAS, SIS
MORE	More participationDateAndCourse and studentID are collected than intended.	Due to a software error, students can register participations for dates and courses they did not participate in.	Additional participation is registered.	SIS
LESS	Less participationDateAndCourse and studentID are collected than intended.	Due to a software error, a student's participation for a date and course is not registered.	Participation is not registered.	SIS
INCORRECT	The collected participationDateAndCourse and studentID are incorrect.	Due to a software error, a participation is registered for a wrong date, course, or student.	1. Participation is not registered 2. Additional participation is registered	SIS
BEFORE	participationDateAndCourse and studentID are collected before another prior operation.	Course evaluation system allows to register a participation without a prior authentication.	Participation and evaluation cannot correctly be assigned to the corresponding student.	SIS

sistently estimate and evaluate the risks they imply. These privacy threats are represented by threat scenarios, unwanted incidents, and the privacy requirements they harm in threat diagrams (see Section 13.3). For the sake of simplicity, I leave out the threat scenarios related to software errors located in the machine. This is, because these threat scenarios can all be mitigated by a correct and robust design and implementation of the machine.

For presentation purposes, I splitted the global threat model into four views, namely a view on the threats concerning the availability and integrity requirements, and three views on the threats concering the counterstakeholders student, teacher, and attacker. I present these four views in the following.

Figure 19.21 shows the threat diagram for the availability requirement SAS and the integrity requirement SIS. These two privacy requirements may both be violated by the three unwanted incidents Student cannot evaluate the course, Participation is not registered, and Evaluation results are not stored, because these situations all represent cases in which personal data of students are not available, and become incomplete or inconsistent. Furthermore, I identified from the previously given tables different chains of causes that may lead to these unwanted incidents and that are helpful to estimate the likelihood of the unwanted incidents to occur. The likelihoods and consequences are already annotated in the given threat diagrams, but they are actually annotated in the following step (see Section 19.6.3).

Table 19.27.: *Deviations for the storage of participationDateAndCourse and studentID at Participations in requirement Participate*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	participationDateAndCourse and studentID are not stored	Due to a software error, a participation cannot be registered	Participation is not stored.	SAS, SIS
INCORRECT	The stored participationDateAndCourse and studentID are incorrect.	Due to a software error, a participation is registered for a wrong date, course, or student.	1. Participation is not stored. 2. Additional participation is stored.	SIS
REVERSE	participationDateAndCourse and studentID are deleted.	Due to a software error, registered participation are deleted.	Participation is not stored.	SIS
AFTER	participationDateAndCourse and studentID are stored after another subsequent operation.	Due to a software error, a participation is stored after the student requests to evaluate a course.	Participation is not stored timely.	SIS

The scenario that students may try to maliciously authenticate as another student and consequently get access to the course evaluation system using another student's credentials is shown in the threat diagram in Figure 19.22. The unwanted incident may harm the data confidentiality requirement SDS0S that states that only the individual students are allowed to access their personal data, and the data unlinkability and anonymity requirements UDS00S and UAS00S that state that only the individual students shall be able to link their personal data to themselves. Additionally, the exceptional information requirement TES0 concerned with the notification of data breaches may be violated, because a violation of the data confidentiality, data unlinkability, and anonymity requirements forms a data breach that the data subject student is possibly not informed about.

The threat diagram in Figure 19.23 shows a chain of causes that starts with the scenario that a student enters self-identifying information in a free text field, and finally leads to the unwanted incident that a teacher can link the evaluation results or parts of it to that student. This unwanted incident harms the data unlinkability requirement UDS21T, and the anonymity requirement UAS21T, because these requirements state the teachers shall only be able to link the aggregated evaluation results to a medium-sized group of students or other personal data of students that are available to teachers. Consequently, the unwanted incident may harm the data confidentiality requirement SDS3T, because teachers may get to know the opinionOnCourseAndTeacher of a student. The possible violations of the data confidentiality, data unlinkability, and anonymity requirements again may lead to data breaches, and consequently the exceptional information requirement TES0 can also be violated by the unwanted incident.

The threat diagram for the counterstakeholder attacker is shown in Figure 19.24. I identified three possible scenarios leading to unwanted incidents that all violate the data confidentiality requirement SDS3Att that states that attackers shall not be able to access personal data of

Table 19.28.: *Deviations for the deduction of sufficientNumberOfParticipationsForCourse by Course Evaluation System in requirement Request Evaluation*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	sufficientNumberOfParticipationsForCourse are not deduced.	Due to a software error, the stored participations cannot be retrieved.	Evaluation form is not provided to student.	SAS, SIS
MORE	More sufficientNumberOfParticipationsForCourse are deduced than necessary.	Additional participation is registered or stored.	Evaluation form is provided to a student who does not satisfy the evaluation requirements.	SIS
LESS	Less sufficientNumberOfParticipationsForCourse are deduced than necessary.	Participation is not registered or stored.	Evaluation form is not provided to student.	SAS, SIS
BEFORE	sufficientNumberOfParticipationsForCourse are or can be deduced before another prior operation.	Due to a software error, the information whether a student sufficiently often participated in a course is computed while the student could still register a participation.	Evaluation form is not provided to student.	SAS, SIS
LATE	sufficientNumberOfParticipationsForCourse are deduced later than intended relative to clock time.	Due to a software error, the information whether a student sufficiently often participated in a course is computed after the evaluation period ended.	Evaluation form is not provided to student.	SAS, SIS

students, and the exceptional information requirement TES0. The three threat scenarios are that the attacker may attack the teacher's browser or the student's browser to eavesdrop personal data, and that the attacker may steal a student's credentials.

19.6.3. Evaluate Privacy Risks

To evaluate the privacy risks of the course evaluation system based on the above given threat diagrams, I need to define likelihood and consequence scales for the different privacy requirements. The likelihood scale for the course evaluation system is given in Table 19.36. For the security- and unlinkability-related privacy requirements, I defined three consequence scales that are shown in Table 19.37. I decided to create different consequence scales for the personal data related to

Table 19.29.: *Deviations for the flow of sufficientNumberOfParticipationsForCourse to Student in requirement Request Evaluation*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	sufficientNumberOfParticipationsForCourse do not flow to Student.	Due to a software error, the information whether a student sufficiently often participated in a course is not computed or provided to the respective student.	Evaluation form is not provided to student.	SAS, SIS
OTHER THAN	sufficientNumberOfParticipationsForCourse flow to Attacker.	The attacker may get to know whether a student sufficiently often participated in a course by attacking the network or the student's browser	The attacker knows whether the student sufficiently often participated in a course.	SDS3Att, TES0

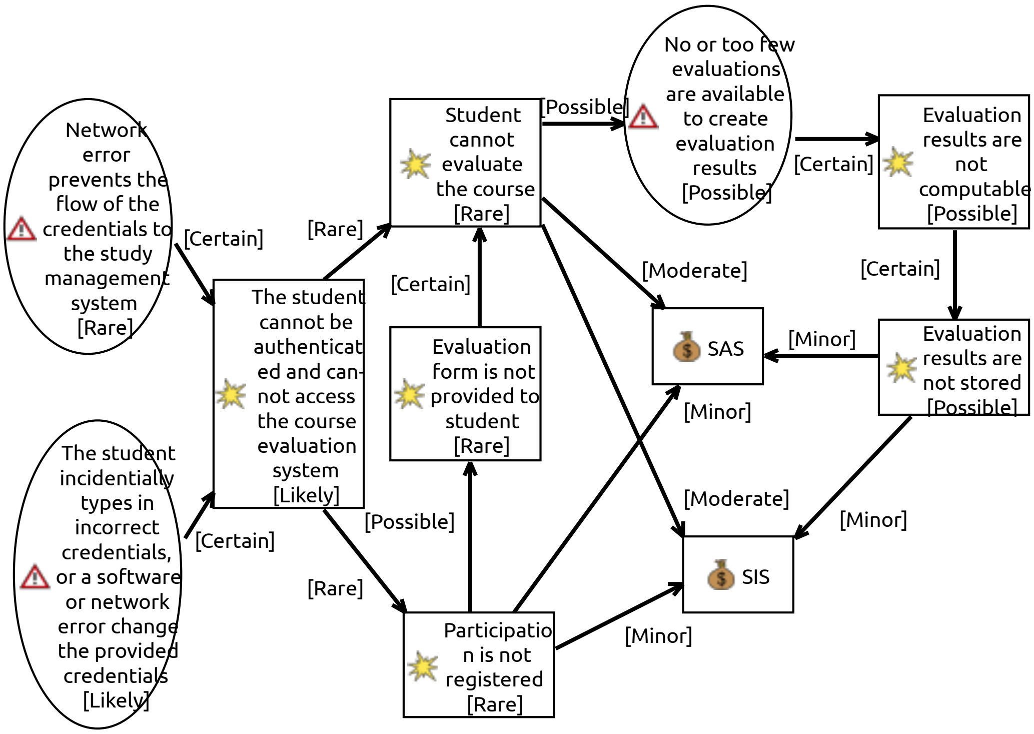

Network error prevents the flow of the credentials to the study management system [Rare]
[Certain]
The student incidentially types in incorrect credentials, or a software or network error change the provided credentials [Likely]
[Certain]
The student cannot be authenticated and cannot access the course evaluation system [Likely]
[Rare]
[Rare]
Student cannot evaluate the course [Rare]
[Certain]
Evaluation form is not provided to student [Rare]
[Possible]
Participation is not registered [Rare]
[Possible]
No or too few evaluations are available to create evaluation results [Possible]
[Certain]
Evaluation results are not computable [Possible]
[Certain]
Evaluation results are not stored [Possible]
[Moderate]
SAS
[Minor]
[Minor]
[Moderate]
SIS
[Minor]
[Minor]

Figure 19.21.: *Threat diagram for the availability requirement SAS and the integrity requirement SIS*

participations of students in courses, evaluations of students for courses, and evaluation results, because these all have a different information content and sensitivity. The consequence scales that I use for transparency- and intervenability-related requirements are shown in Table 19.38.

The threat diagrams given in Figures 19.21, 19.22, 19.23, and 19.24 already contain the estimated likelihoods and consequences for the threat scenarios and unwanted incidents. As the estimation of likelihoods and consequences is not in the focus of my research, I do not pro-

Table 19.30.: *Deviations for the collection of studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies from Student and Study Management System in requirement Evaluate (part 1)*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are not collected	1. The student cannot be authenticated and cannot access the course evaluation system. 2. Evaluation form is not provided to student.	Student cannot evaluate the course.	SAS, SIS
MORE	More studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are collected than intended.	Evaluation form is provided to a student who does not satisfy the evaluation requirements.	Student evaluates course for which he or she did not participated sufficiently often in.	SIS
LESS	Less studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are collected than intended.	Due to a software error, a student's evaluation is not registered.	Students evaluation is not stored.	SAS, SIS
AS WELL AS	In addition to studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies, self-identifying information is collected.	The student enters additional, e.g., self-identifying information in a free text field.	The student's opinionOnCourseAndTeacher can be linked to the respective student, and hence possibly parts of the aggregatedEvaluations	UAS21T, UDS21T, SDS3T, TES0

vide details on this. To calculate consistent likelihoods in the threat diagrams, I obtained the likelihood of an unwanted incident as the maximum of the likelihoods implied by the incoming causes relations. The implied likelihood of a causes relation is obtained by taking the minimum likelihood of the source and the likelihood of the causes relation itself. For example, the likelihood of the unwanted incident The student cannot be authenticated and cannot access the course evaluation system in Figure 19.21 is set to Likely, because the incoming causes relations imply likelihoods of Rare (minimum of Rare and Certain) and Likely (minimum of Likely and Certain).

Table 19.39 lists all privacy risks that can be deduced from the previously introduced threat

Table 19.31.: *Deviations for the collection of studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies from Student and Study Management System in requirement Evaluate (part 2)*

Guide word	**Deviations**	**Possible causes**	**Consequences**	**Harmed Privacy Requirements**
INCORRECT	The collected studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are incorrect.	1. Due to a software error, the student's evaluation is unintendedly modified. 2. The student him- or herself enters incorrect data. 3. The attacker or another student has access to the course evaluation system using the stolen credentials and enters incorrect data.	Student's evaluation is incorrect.	SIS
LATE	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are collected later than intended relative to clock time.	Due to a software error, the evaluation is collected after the evaluation period ended.	Student's evaluation is not contained in the evaluation results.	SAS, SIS
AFTER	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are collected after another subsequent operation.	Due to a software error, the evaluation results are computed before the student was able to submit his or her evaluation.	Student's evaluation is not contained in the evaluation results.	SAS, SIS

diagrams. Each harms relation from an unwanted incident to a privacy requirement represents a privacy risk. Its likelihood is the likelihood of the respective unwanted incidents and its consequence is annotated at the harms relation.

The identified privacy risks are put into the risk matrix shown in Table 19.40. The cells with white background represent acceptable risks, those with light gray background tolerable risks, and those with dark gray background unacceptable risks. The risk matrix shows that the privacy risks concerning the exceptional information requirement TES0 need to be treated, and also two risks concerning the counterstakeholder attacker. The risks concerning malicious students as counterstakeholders are considered as tolerable, as well as, the risks that the evaluation results

Table 19.32.: *Deviations for the storage of studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies at Evaluations in requirement Evaluate*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are not stored	Due to a software error, a student's evaluation is not registered.	Student's evaluation is not stored.	SIS
INCORRECT	The stored studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are incorrect.	Due to a software error, an evaluation is registered for the wrong course, or student.	1. Student's evaluation is not stored. 2. Student's evaluation is incorrect.	SIS
REVERSE	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are deleted.	Due to a software error, registered evaluations are deleted.	Students' evaluations are not stored.	SIS
AFTER	studentID, opinionOnCourseAndTeacher, sufficientNumberOfParticipationsForCourse, and studies are stored after another subsequent operation.	Due to a software error, an evaluation is stored after the evaluation results are computed.	Student's evaluation is not contained in the evaluation results.	SIS

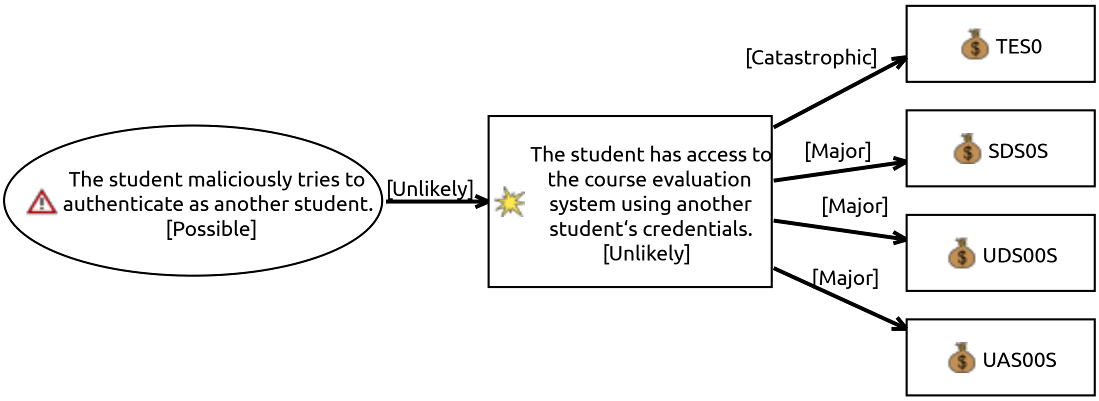

The student maliciously tries to authenticate as another student. [Possible]
[Unlikely]
The student has access to the course evaluation system using another student's credentials. [Unlikely]
[Catastrophic]
[Major]
[Major]
[Major]
TES0
SDS0S
UDS00S
UAS00S

Figure 19.22.: *Threat diagram for the data confidentiality, data unlinkability, and anonymity requirements for the counterstakeholder student*

are not stored. The other privacy risks are considered to be acceptable.

Table 19.33.: *Deviations for the deduction of aggregatedEvaluations by Course Evaluation System in requirement Create Report*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	aggregatedEvaluations are not deduced.	No or too few evaluations are available to create evaluation results.	Evaluation results are not computable.	SAS
MORE	More aggregatedEvaluations are deduced than necessary.	The student's opinionOnCourseAndTeacher can be linked to the respective student.	Evaluation results or parts of them are linkable to the corresponding students.	UAS21T, UDS21T, SDS3T, TES0
INCORRECT	The deduced aggregatedEvaluations are incorrect.	1. Student's evaluation is incorrect. 2. A software error leads to a creation of invalid evaluation results	Evaluation results are incorrect.	SIS
REVERSE	original PD are or can be deduced from aggregatedEvaluations.	The student enters additional, e.g., self-identifying information in a free text field.	Evaluation results or parts of them are linkable to the corresponding students.	UAS21T, UDS21T, SDS3T, TES0
EARLY	aggregatedEvaluations are deduced earlier than intended relative to clock time.	The evaluation results are computed before the evaluation period ended.	The evaluation results are incomplete	SIS

Table 19.34.: *Deviations for the storage of aggregatedEvaluations at Evaluation Reports in requirement Create Report*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	aggregatedEvaluations are not stored	Evaluation results are not computable.	Evaluation results are not stored.	SAS, SIS
INCORRECT	The stored aggregatedEvaluations are incorrect.	1. Students' evaluations are incorrect. 2. A software error leads to an unintended modification of the evaluation results.	Evaluation results are incorrect.	SIS
REVERSE	aggregatedEvaluations are deleted.	Due to a software error, evaluation results are deleted.	Evaluation results are not stored.	SAS, SIS

Table 19.35.: *Deviations for the flow of aggregatedEvaluations to Teacher in requirement Show Results*

Guide word	Deviations	Possible causes	Consequences	Harmed Privacy Requirements
NO	aggregatedEvaluations do not flow to Teacher.	1. Evaluation results are not stored. 2. Teacher cannot be authenticated. 3. Due to a software error, the evaluation results are not provided.	Teacher cannot access the evaluation results.	-
MORE	More aggregatedEvaluations flow to Teacher than intended.	Evaluation results or parts of it are linkable to the corresponding students.	Teacher can link the evaluation results or parts of it to specific students.	UAS21T, UDS21T, SDS3T, TES0
INCORRECT	The aggregatedEvaluations flowing to Teacher are incorrect.	Evaluation results are incorrect.	Teacher accesses incorrect evaluation results.	SIS
OTHER THAN	aggregatedEvaluations flow to Attacker.	The attacker may access the evaluation results by attacking the network or the teacher's browser	The attacker has access to the evaluation results.	SDS3Att, TES0
EARLY	aggregatedEvaluations flow earlier than intended to Teacher relative to clock time.	The teacher requests the evaluation results before the evaluation period ended and the results were computed.	Teacher cannot access the evaluation results.	-

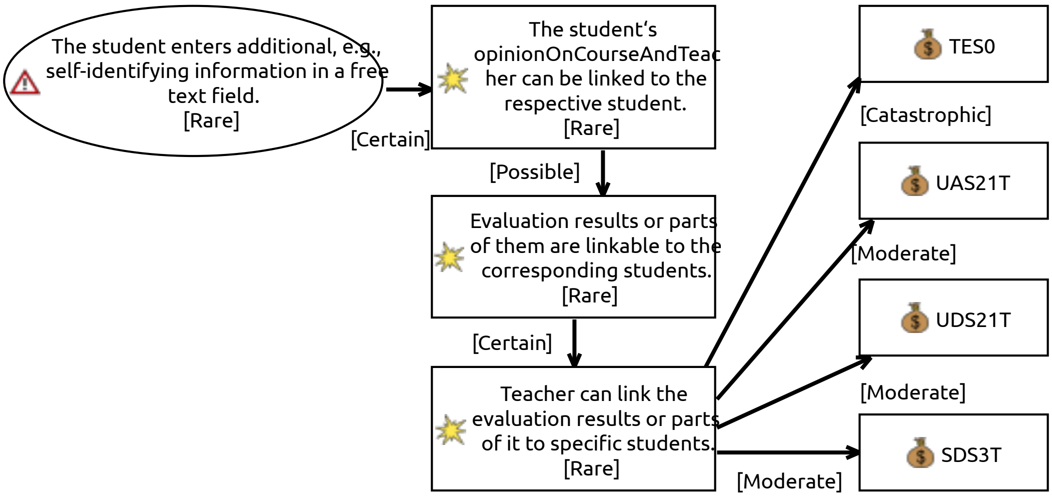

The student enters additional, e.g., self-identifying information in a free text field. [Rare]
[Certain]
The student's opinionOnCourseAndTeacher can be linked to the respective student. [Rare]
[Possible]
Evaluation results or parts of them are linkable to the corresponding students. [Rare]
[Certain]
Teacher can link the evaluation results or parts of it to specific students. [Rare]
TES0
[Catastrophic]
UAS21T
[Moderate]
UDS21T
[Moderate]
SDS3T
[Moderate]

Figure 19.23.: *Threat Diagram for the data confidentiality, data unlinkability, and anonymity requirements for the counterstakeholder teacher*

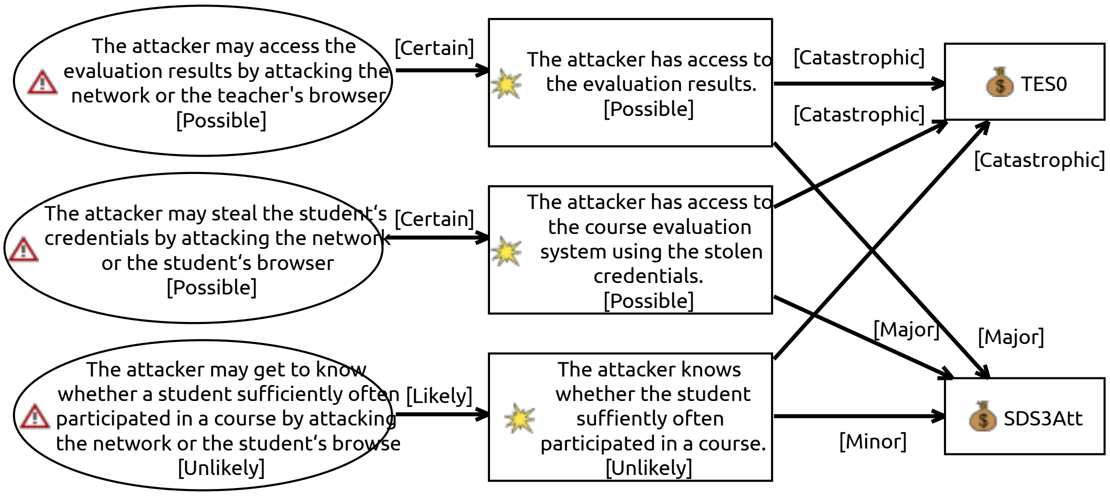

The attacker may access the evaluation results by attacking the network or the teacher's browser [Possible]
[Certain]
The attacker has access to the evaluation results. [Possible]
[Catastrophic]
TES0
[Catastrophic]
[Catastrophic]
The attacker may steal the student's credentials by attacking the network or the student's browser [Possible]
[Certain]
The attacker has access to the course evaluation system using the stolen credentials. [Possible]
[Major]
[Major]
The attacker may get to know whether a student sufficiently often participated in a course by attacking the network or the student's browse [Unlikely]
[Likely]
The attacker knows whether the student suffiently often participated in a course. [Unlikely]
[Minor]
SDS3Att

Figure 19.24.: *Threat Diagram for the data confidentiality, data unlinkability, and anonymity requirements for the counterstakeholder attacker*

Table 19.36.: *Likelihood scale for the course evaluation system*

Likelihood	Definition
Rare	Less than once per ten years
Unlikely	Less than once per two years
Possible	Less than twice per year
Likely	Two to five times per year
Certain	Five times or more per year

Table 19.37.: *Consequence scales for security- and unlinkability-related privacy requirements*

Consequence	Security of participations	Security of evaluations	Security of evaluation results
Insignificant	1-2 participations of a student for the same course are affected	0 evaluations are affected	0 evaluation results are affected
Minor	3-4 participations of a student for the same course are affected	1-2 evaluations are affected	-
Moderate	5-6 participations of a student for the same course are affected	3-4 evaluations are affected	1-2 evaluation results are affected
Major	7-10 participations of a student for the same course are affected	5-6 evaluations are affected	3-4 evaluation results are affected
Catastrophic	>10 participations of a student for the same course are affected	>6 evaluations are affected	>4 evaluation results are affected

Table 19.38.: *Consequence scales for the transparency- and intervenability-related privacy requirements*

Consequence	Transparency	Intervenability
Insignificant	Information is provided in a sufficient manner, but not recognized	Data subject can intervene and is sufficiently informed about the options, but does not recognize this information
Minor	Information is provided, but in an insufficient manner	Data subject can intervene, but is insufficiently informed about the options
Moderate	Information is available only with manual effort	Data subject can intervene, but is not informed about the options
Major	Information is only available on request	Data subject has only partial intervention options
Catastrophic	Information is not accessable and not provided on request	Data subject has no possibility to intervene

Table 19.39.: *Risks identified for the course evaluation system*

Risk	Unwanted incident	Privacy req.
RSAS1	Student cannot evaluate the course	SAS
RSAS2	Participation is not registered	SAS
RSAS3	Evaluation results are not stored	SAS
RSIS1	Student cannot evaluate the course	SIS
RSIS2	Participation is not registered	SIS
RSIS3	Evaluation results are not stored	SIS
RTES01	The student has access to the course evaluation system using another student's credentials	TES0
RSDS01	The student has access to the course evaluation system using another student's credentials	SDS0S
RUDS00S	The student has access to the course evaluation system using another student's credentials	UDS00S
RUAS00S	The student has access to the course evaluation system using another student's credentials	UAS00S
RTES02	Teacher can link the evaluation results or parts of it to specific students	TES0
RSDS3T	Teacher can link the evaluation results or parts of it to specific students	RSDS3T
RUDS21T	Teacher can link the evaluation results or parts of it to specific students	RUDS21T
RUAS21T	Teacher can link the evaluation results or parts of it to specific students	RUAS21T
RTES03	The attacker has access to the evaluation results	TES0
RSDS3Att1	The attacker has access to the evaluation results	SDS3Att
RTES04	The attacker has access to the course evaluation system using the stolen credentials	TES0
RSDS3Att2	The attacker has access to the course evaluation system using the stolen credentials	SDS3Att
RTES05	The attacker knows whether the student sufficiently often participated in a course	TES0
RSDS3Att3	The attacker knows whether the student sufficiently often participated in a course	SDS3Att

19.7. Privacy Measure Integration

Having identified the privacy requirements of the course evaluation system, and having identified and evaluated the risks to them, I select privacy measures and reason why these satisfy the privacy requirements and mitigate the risks to these. That is, I apply the step Privacy Measure Integration introduced in Chapter 17.

In Section 19.7.1, I briefly describe how I prioritized the privacy requirements and risks. I describe the selection and integration of privacy measures, and the specification of satisfaction arguments for different groups of privacy requirements in Sections 19.7.2-19.7.5. Availability and integrity requirements are considered in Section 19.7.2, confidentiality requirements in Section 19.7.3, transparency requirements in Section 19.7.4, and intervenability requirements in Section 19.7.5. I start in Sections 19.7.3-19.7.5 with the selection of privacy measures, followed by their integration, and finally I present the satisfaction arguments for the respective privacy requirements.

19.7.1. Prioritize Privacy Requirements and Risks

As the course evaluation system has a small scope and is not concerned with processing critical personal data, except the students' opinion on the course and teacher, I decided to set the highest acceptable risk level for all privacy requirements that do not concern the personal data opinionOnCourseAndTeacher to tolerable, and for all privacy requirements that are concerned with these personal data to acceptable.

19.7.2. Addressing Availability and Integrity Requirements

The scenarios leading to a violation of the availability and integrity requirements are all out of the scope of the machine and can hardly be handled by technical measures (see Figure 19.21). Additionally, the risks identified for the availability and integrity requirements are evaluated to be acceptable or tolerable. The tolerable risks originate from the case that not enough students have evaluated in a course to create an evaluation report. This restriction exists to ensure that the evaluation results cannot be linked to individual students. Hence, I did not select any privacy measures to address the availability and integrity requirements of the course evaluation system, and consequently, I considered the availability requirement SAS and the integrity requirement SIS to be satisfied.

Table 19.40.: *Risk matrix for the course evaluation system*

	Insignificant	Minor	Moderate	Major	Catastrophic
Rare		RSAS2, RSIS2	RSAS1, RSIS1, RSDS3T, RUDS21T, RUAS21T		RTES02
Unlikely		RSDS3Att3		RSDS01, RUDS00S, RUAS00S	RTES01, RTES05
Possible		RSAS3, RSIS3		RSDS3Att1, RSDS3Att2	RTES03, RTES04
Likely					
Certain					

19.7.3. Addressing Data Confidentiality, Data Unlinkability, and Anonymity Requirements

19.7.3.1. Select Privacy Measures

The functional requirement Authenticate already represents a privacy measure to ensure that only authenticated teachers and students are allowed to access the course evaluation system. As the authentication mechanism of the existing study management system is used, the correctness and effectiveness of the authentication mechanism depends on this other system. I assumed that the existing authentication mechanism is properly designed, because the study management system has to deal with more personal data and has to satisfy stronger privacy and security requirements than the course evaluation system. Hence, I assumed that the authentication mechanism provided by the study management system is sufficiently secure.

The threats concerning the attacks on the students' and teachers' browser are also out of the scope of the course evaluation system. For the sake of simplicity, I assumed that students and teachers are sufficiently aware of these threats and able to protect themselves against these.

The only data unlinkability and anonymity requirements that are not completely covered by the authentication mechanism are the requirements UDS21T and UAS21T. However, these are only considered to be violated when students enter self-identifying personal data in free text fields of the evaluation forms. I expected that students are sufficiently aware of this threat and that they rarely do this (cf. Figure 19.23). If this assumption is considered to be too strong, there are different options to mitigate this threat. For example, text fields could be forbidden in evaluation forms, or the content of the free text fields could be checked manually by a person for self-identifying information that is then removed from the evaluation.

Note that it was investigated in the EU project ABC4Trust (Deibler et al., 2014) to use anonymous-based credentials (see the PET pattern Privacy-ABCs in Section 16.3.1) for a course evaluation system with similar requirements. In that pilot, smart cards were provided to the students with which they could register their participation in a course using a smart card reader in the lecture hall. The number of participations were then stored on the smart card. Using their smart card and a smart card reader the students were able to prove that they sufficiently often participated in a course to evaluate it using an online platform without revealing additional information about them. However, I considered the implementation and integration of attribute-based credentials for the simple course evaluation system considered in this chapter as too expensive (see also the liabilities documented for Privacy-ABCs in Section 16.3.1).

19.7.3.2. Integrate Privacy Measures

The previously mentioned assumptions are added to the problem frame model for the course evaluation system as shown in the domain knowledge diagram in Figure 19.25. The assumption Secure Authentication constrains the Study Management System to perform the authentication in a secure way, and the assumption Security Awareness constrains the biddable domains Student and Teacher to be aware of the security issues concerning the interactions with the course evaluation system. This assumption also includes that students are aware that they should not enter self-identifying information in the free text fields of the evaluation forms.

Note that I also added the above mentioned privacy measures to the respective collection, storage, and flow information requirements (see attribute measures in Figure 7.6 on page 112) to document that students shall be informed about the technical measures integrated in the course evaluation system to protect their personal data.

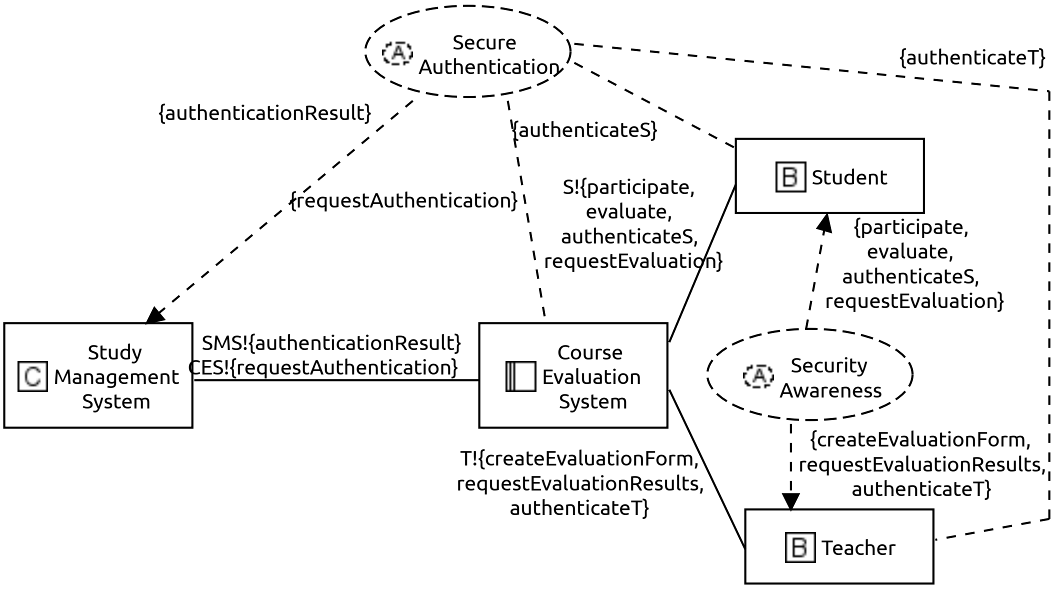

Secure Authentication
{authenticateT}
{authenticationResult}
{authenticateS}
{requestAuthentication}
S!{participate, evaluate, authenticateS, requestEvaluation}
Student
{participate, evaluate, authenticateS, requestEvaluation}
Study Management System
SMS!{authenticationResult}
CES!{requestAuthentication}
Course Evaluation System
Security Awareness
{createEvaluationForm, requestEvaluationResults, authenticateT}
T!{createEvaluationForm, requestEvaluationResults, authenticateT}
Teacher

Figure 19.25.: *Domain knowledge diagram for the assumptions concerning the satisfaction of the data confidentiality, data unlinkability, and anonymity requirements*

19.7.3.3. Document Satisfaction Arguments

With the functional requirement Authenticate, the assumptions Secure Authentication and Security Awareness, I can reason that the data confidentiality requirements SDS3A, SDS1T, SDS3T, SDS0S and SDS3S, the data unlinkability requirements UDS01T, UDS21T and UDS00S, and the anonymity requirements UAS01T, UAS21T and UAS00S are sufficiently satisfied, because only authenticated users get access to the course evaluation system, and students and teachers are aware of security risks and behave accordingly.

19.7.4. Addressing Transparency Requirements

19.7.4.1. Select Privacy Measures

I decided to provide a privacy policy that contains the information about the processing of personal data by the course evaluation system to students. This privacy policy shall be accessible without a prior authentication from the course evaluation system. The privacy policy shall address all collection, storage, and flow information requirements. This also means that the privacy policy shall inform the students about their intervention options and the consequences of these.

I did not decide to select any technical privacy measures to address the exceptional information requirements. These are addressed by non-technical measures, e.g., by informing students about the occurrence of data breaches, system changes, and non-compliance by sending the affected or possibly affected students a notification when this is necessary. Also authorities are informed in these cases.

To comply to the storage information requirements, I needed to refine the functional requirements Request Evaluation and Create Report. This is, because during the data flow analysis, I documented that the participations and evaluations shall only be retained for the time that they are needed for the purpose of the course evaluation. The need to store the participations ends with the successful request of the respective evaluation form by the corresponding student. Hence, I added to functional requirement Request Evaluation the need to delete the no longer needed data from the domain Participations. Furthermore, the need to store a student's evalua-

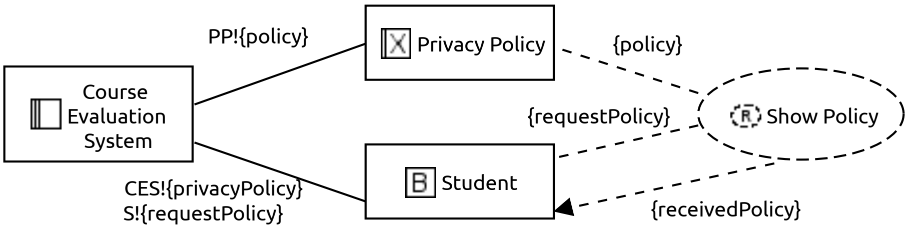


Figure 19.26.: *Problem diagram for the functional requirement Show Policy*

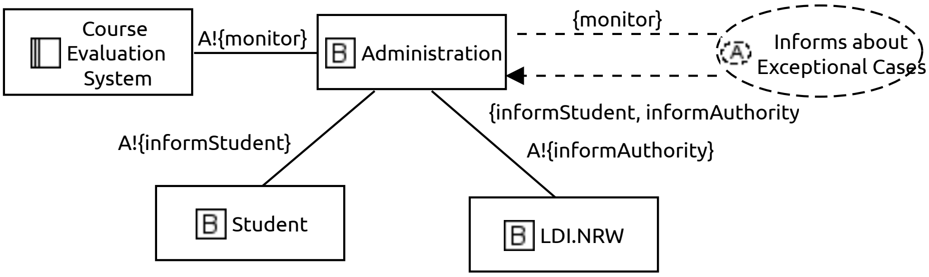


Figure 19.27.: *Domain knowledge diagram for the assumption concerning the satisfaction of the exceptional information requirements*

tion in an identifiable way in the domain Evaluations ends with the creation of the corresponding evaluation report. Hence, I refined the functional requirement Create Report to delete the used data from the designed domain Evaluations after the creation of an evaluation report.

19.7.4.2. Integrate Privacy Measures

Figure 19.26 shows the problem diagram for the additional functional requirement Show Policy that allows students to retrieve the privacy policy of the course evaluation system. The domain knowledge diagram for the assumption addressing the exceptional information requirements is shown in Figure 19.27. For the sake of simplicity, I do not show the updated problem diagrams for the functional requirements Request Evaluation and Create Report.

19.7.4.3. Document Satisfaction Arguments

Due to the added functional requirement Show Policy, the students can retrieve the privacy policy of the course evaluation system. This functionality shall be available without a prior authentication, and the relevant parts of the privacy policy shall also be shown whenever the students provide personal data (see aspect Obtain Consent presented in Section 19.7.5). In this way, all collection, storage, and flow information requirements are satisfied, as the privacy policy shall contain all needed information. The modifications to the functional requirements Request Evaluation and Create Report additionally ensure that the retention period of the data available at the lexical domains Participations and Evaluations complies to the storage information requirements.

The assumption Informs about Exceptional Cases documents that the administration informs students and the LDI.NRW as supervisory authority about the occurrence of exceptional cases during the processing of the students' personal data by monitoring the course evaluation system. In this way, all exceptional information requirements are satisfied.

19.7.5. Addressing Intervenability Requirements

19.7.5.1. Select Privacy Measures

As the processing of the students' personal data shall be based on consent, this consent has to be obtained prior the data are processed. I decided to obtain this consent during each action with which students provide personal data to the course evaluation system. This is, before students submit data to the course evaluation system the relevant part of the privacy policy shall be presented to them and they have to explicitly consent to the described processing of the data they provide.

To address the data subject intervention requirements concerning the right to review the processed personal data, I added a functionality with which the students can retrieve all personal data about them stored by the course evaluation system. These personal data include the recorded participations and evaluations, but not the aggregated evaluations. From the list of processed personal data, the students shall also be able to withdraw the consent to process a specific recorded participation or an evaluation. This option to withdraw the consent to process the respective personal data also allows the students to object to the processing of their personal data. Additionally, students shall be able to edit their evaluations as long as the evaluation period has not yet ended. In this way, the data subject intervention requirements concerning the rights to challenge the accuracy and completeness of the processed personal data are implemented.

As I did not expect frequent requests of supervisory authorities concerning the course evaluation system, I decided not to handle these requests with technical measures. I assumed that the authority intervention requirements can be satisfied by the administration as controller of the course evaluation system.

19.7.5.2. Integrate Privacy Measures

The consent of students to process their personal data shall be obtained whenever they provide personal data. Hence, obtaining the consent of students is a cross-cutting functional requirement formulated by the aspect Obtain Consent. The corresponding aspect diagram is shown in Figure 19.28. The aspect Obtain Consent refers to the event provideData controlled by the biddable join point User. Before this event is further processed, the referred to policy shall be presented to users by the machine join point Data Processor in the form of a consent form (constraint on the phenomenon showConsentForm of the Data Processor). The User can then decide to consentToPolicy or not. When the User gives consent to the processing, the consent shall be stored in the lexical domain Consent, and the Data Processor shall only further process the data of the users if a respective consent was recorded.

The Obtain Consent aspect needs to be integrated into the functional requirements Participate, Evaluate, and Authenticate, because the student provides personal data in the context of these. The join point instantiations to weave the Obtain Consent aspect into the previously mentioned requirements is shown in Table 19.41. For the sake of simplicity, I omit the corresponding sequence diagrams.

The problem diagram for the functional requirement Review that allows students to query the personal data processed of them from the course evaluation system is shown in Figure 19.29. Figure 19.30 shows the problem diagram for the functional requirement Withdraw Consent. This requirement allows students to withdraw the consent to process their personal data, which consequently results in the deletion of the documented consent, and the affected data in the lexical domains Participations and Evaluations. I created for the functional requirement Challenge Accuracy and Completeness a problem diagram similar to the problem diagram shown in Figure 19.30.

The assumption Handle Authority Interventions that documents that the administration can

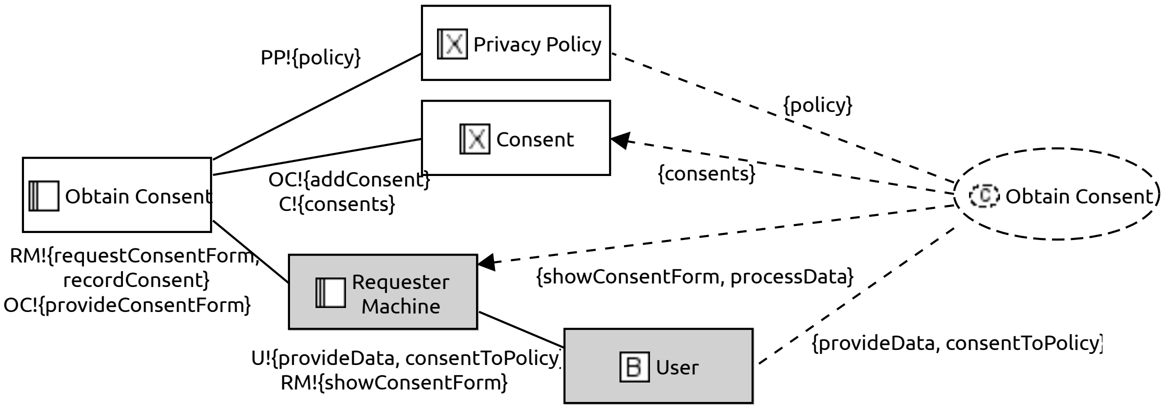


Figure 19.28.: *Aspect diagram for the cross-cutting functional requirement* Obtain Consent

Table 19.41.: *Join point instantiations for the integration of the* Obtain Consent *aspect into the functional requirements* Participate, Evaluate, *and* Authenticate

Join Point	**Instantiation for Participate**	**Instantiation for Evaluate**	**Instantiation for Authenticate**
User	Student	Student	Student
Data Processor	Course Evaluation System	Course Evaluation System	Course Evaluation System
provideData	participate	evaluate	authenticateS
processData	addParticipation, queryEnrollment	addEvaluation	requestAuthentication

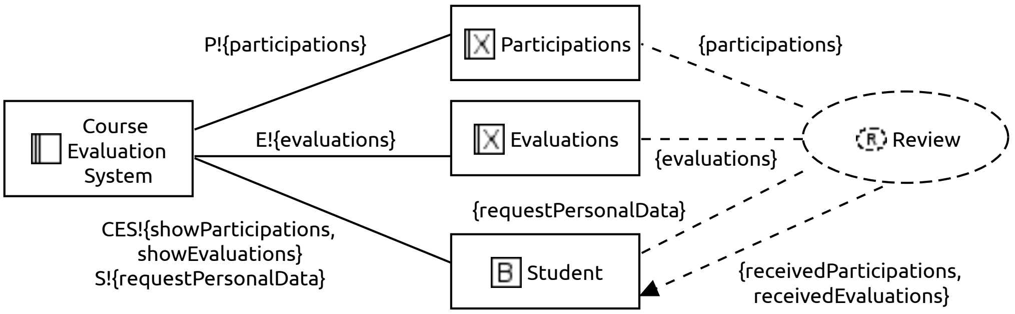


Figure 19.29.: *Problem diagram for the functional requirement* Review

handle intervention requests of authorities is modeled in a domain knowledge diagram similar to the domain knowledge diagram shown in Figure 19.27.

19.7.5.3. Document Satisfaction Arguments

The data subject intervention requirements with the type doNotConsent are realized by integrating the aspect Obtain Consent into the functional requirements Participate, Evaluate, and Authenticate, because the personal data of students are only processed if these give consent to process the personal data.

Functional requirements Review, Withdraw Consent, and Challenge Accuracy and Completeness satisfy the data subject intervention requirements with the types review, withdrawConsent, challengeAccuracy, and challengeCompleteness, respectively, by providing functionalities to students to exercise their data subject rights.

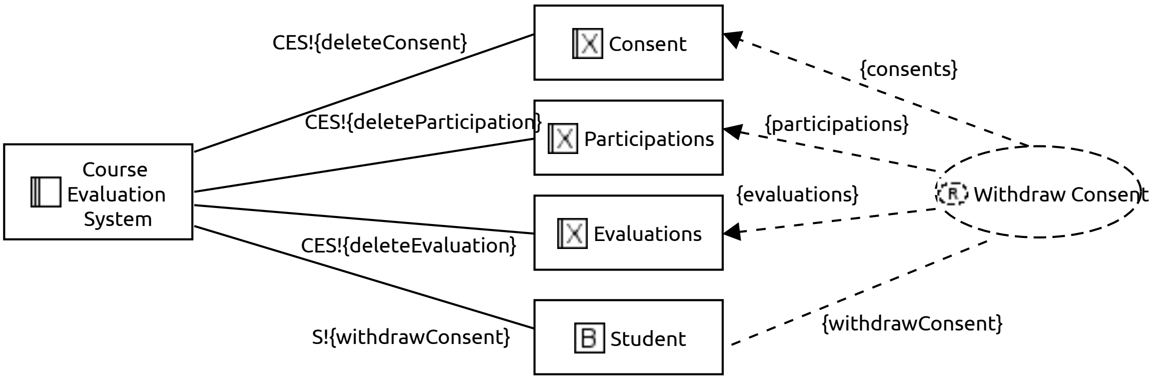


Figure 19.30.: *Problem diagram for the functional requirement* Withdraw Consent

The authority intervention requirements are satisfied by the assumption Handle Authority Interventions that states that the administration is able to handle all intervention requests of authorities.

19.8. Privacy Risk Analysis Part 2

After the integration of the privacy measures, I assess whether the newly introduced privacy measures cause privacy risks and whether the selected privacy measures sufficiently mitigate the previously identified privacy risks.

As newly introduced privacy risks, I identified that due to the privacy measures Withdraw Consent and Challenge Accuracy and Completeness the integrity and availability requirements SIS and SAS may be harmed. However, as the individual students themselves cause these risks, I consider them as acceptable.

The updated threat diagrams for the data confidentiality, data unlinkability, and anonymity requirements for the counterstakeholders student and attacker are given in Figures 19.31 and 19.32, respectively. These document how the privacy measures Authenticate, Secure Authentication, and Security Awareness reduce the likelihood of the contained threat scenarios, and how the privacy measure Inform about Exceptional Cases reduces the consequences on the exceptional information requirement TES0.

The updated risk matrix is shown in Table 19.42. As the risk matrix contains no unacceptable risks, and all provided satisfaction arguments sufficiently reason that the privacy requirements

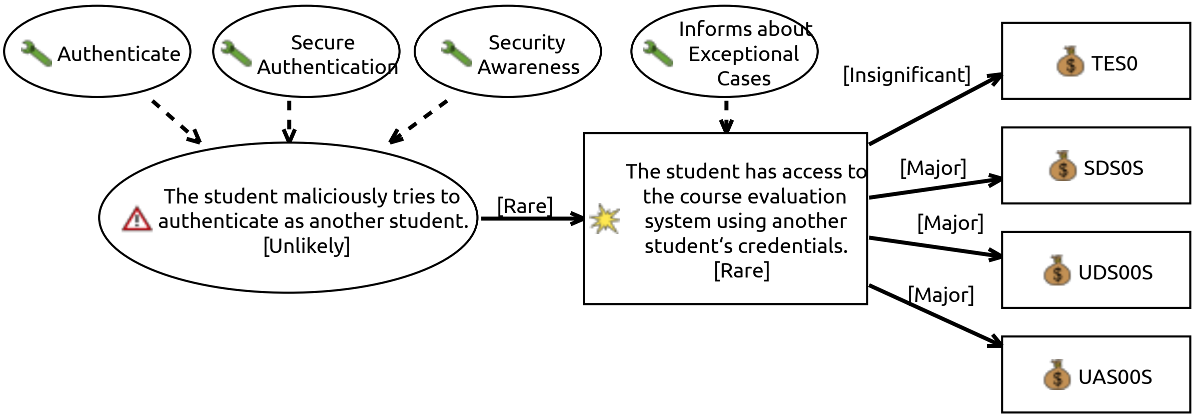

Authenticate
Secure Authentication
Security Awareness
Informs about Exceptional Cases
The student maliciously tries to authenticate as another student. [Unlikely]
[Rare]
The student has access to the course evaluation system using another student's credentials. [Rare]
[Insignificant]
[Major]
[Major]
[Major]
TES0
SDS0S
UDS00S
UAS00S

Figure 19.31.: *Updated threat diagram for the data confidentiality, data unlinkability, and anonymity requirements for the counterstakeholder student*

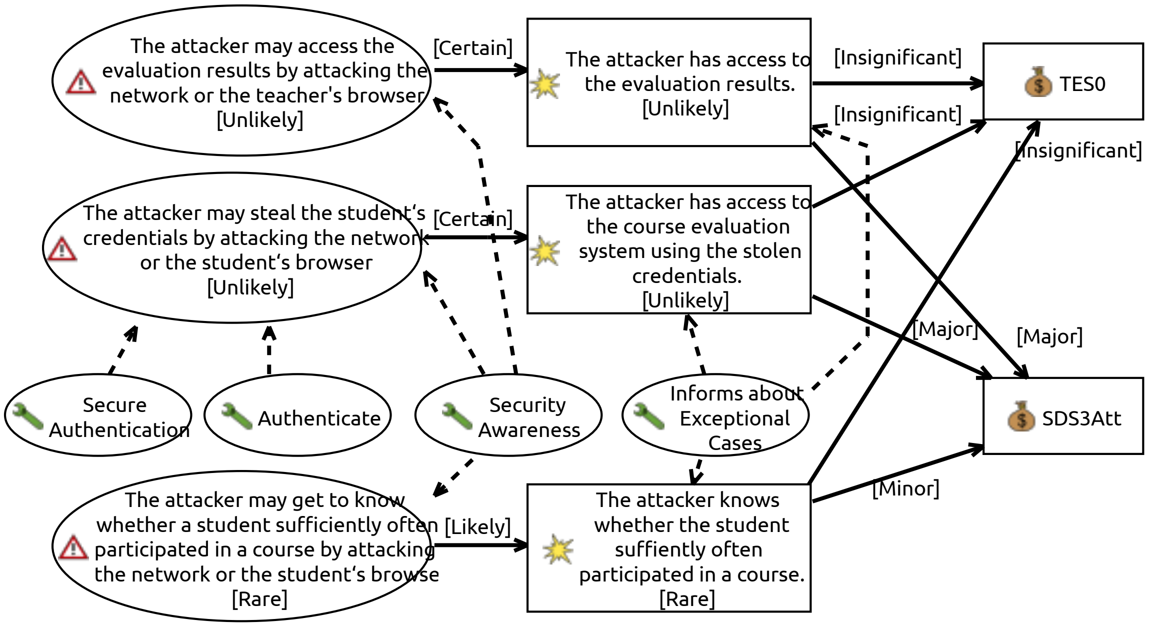

The attacker may access the evaluation results by attacking the network or the teacher's browser [Unlikely]
[Certain]
The attacker has access to the evaluation results. [Unlikely]
[Insignificant]
TES0
[Insignificant]
[Insignificant]
The attacker may steal the student's credentials by attacking the network or the student's browser [Unlikely]
[Certain]
The attacker has access to the course evaluation system using the stolen credentials. [Unlikely]
[Major]
[Major]
Secure Authentication
Authenticate
Security Awareness
Informs about Exceptional Cases
SDS3Att
The attacker may get to know whether a student sufficiently often participated in a course by attacking the network or the student's browse [Rare]
[Likely]
The attacker knows whether the student suffiently often participated in a course. [Rare]
[Minor]

Figure 19.32.: *Updated Threat Diagram for the data confidentiality, data unlinkability, and anonymity requirements for the counterstakeholder attacker*

Table 19.42.: *Updated risk matrix for the course evaluation system*

	Insignificant	Minor	Moderate	Major	Catastrophic
Rare	RTES01, RTES02, RTES05	RSAS2, RSIS2, RSDS3Att3	RSAS1, RSIS1, RSDS3T, RUDS21T, RUAS21T	RSDS01, RUDS00S, RUAS00S	
Unlikely	RTES03, RTES04			RSDS3Att1, RSDS3Att2	
Possible		RSAS3, RSIS3			
Likely					
Certain					

are at least satisfied with a tolerable risk level. Consequently, the privacy analysis with the ProPAn method is finished as all privacy requirements concerning students are sufficiently operationalized and all privacy risks sufficiently mitigated.

19.9. PIA Report Creation

I show in Appendix B the PIA report that I created based on the results obtained by the application of the ProPAn method as shown in this chapter. For the creation of the PIA report, I followed the guidelines provided for the step PIA Report Creation in Chapter 18. To keep the PIA report compact, I left out the details on the analysis of the course evaluation system's indirect environment. Furthermore, I aggregated several privacy requirements and risks. For example, I present only one collection information requirement that aggregates all collection information requirements presented in Section 19.5.

19.10. Conclusion

In this chapter, I applied the ProPAn method on a course evaluation system. This application has shown that the ProPAn method systematically assists conducting a privacy requirements analysis starting with a set of functional requirements. I obtained the privacy relevant domain knowledge and assessed the necessity to perform a detailed privacy analysis using the guidelines and questionnaires provided by the ProPAn method. The ProPAn method also provided guidance to analyze the flows of personal data through the course evaluation system, and to derive the privacy requirements on it from the elicited data flows. During the data flow analysis, I gained a better understanding of the functional requirements, and which data need to be available at which domains for which purposes. For example, I identified that the personal data available at the lexical domains Participations and Evaluations need only to be available there until the corresponding evaluation form is requested and the corresponding evaluation report is created, respectively. This supports the analysis team to follow the principles of privacy-by-design and privacy-by-default and to refine the given functional requirements to adhere to these principles.

Furthermore, the ProPAn method guided me through the identification and evaluation of risks to the system's privacy requirements. Also the selection and integration of privacy measures to implement the privacy requirements and to mitigate the risks to them is supported by the ProPAn method. The application to the course evaluation scenario has shown that non-technical measures, as well as technical measures can be integrated into the problem frame model by adding domain knowledge and problem diagrams to it, and by creating aspect diagrams and corresponding weaving relations for technical privacy measures that contain cross-cutting functional requirements.

The ProPAn model has shown to document all relevant information needed to be documented during a privacy requirements analysis. This is especially visible from the created PIA report for the course evaluation system shown in Appendix B. Additionally, the tool support of the ProPAn method made it easy to apply its steps and allows to create different views on the ProPAn model. For example, Tables 19.10, 19.11, 19.12, 19.13, 19.16, and 19.17, and all figures shown in this chapter are derived from the ProPAn model by the ProPAn tool.

Chapter 20

Conclusion

This chapter concludes my thesis. I provide a summary of my thesis' content in Section 20.1. In Section 20.2, I answer the research questions which I aimed to address within my thesis. These research questions were introduced before in Section 1.2. In Chapter 3, I proposed criteria to compare privacy requirements engineering methods and applied them to the state of the art privacy requirements engineering methods that I identified through a literature review. To compare my proposed privacy requirements engineering method ProPAn with the other state of the art methods, I evaluate ProPAn concerning the previously mentioned criteria, and I compare ProPAn to the state of the art requirements engineering methods in Section 20.3. Finally, I give directions for future research in the context of my thesis in Section 20.4.

20.1. Summary

I summarize my thesis by following its structure. The foundations of my thesis are given in Part I. In Section 20.1.1, I provide an overview of the content of this part. I introduced in Section 1.2 the following central research question underlying my research.

> **How can requirements engineers be supported to consider privacy as a software quality during the requirements analysis starting with a given set of functional requirements?**

My three main contributions to answer this research question are

1. my privacy requirements taxonomy introduced in Part II, which provides a collection of privacy requirements that refine the data protection goals proposed by Hansen et al. (2015),

2. the privacy requirements engineering method ProPAn introduced in Part III, which is a computer-aided method that allows to systematically analyze the privacy protection needs of a software-based system, and

3. the AORE4PF framework introduced in Part IV, which introduces aspect-oriented concepts to Jackson's problem frames approach and that I use in the ProPAn method to integrate privacy measures addressing the privacy protection needs of the system into it.

I summarize the chapters of Parts II-IV in Sections 20.1.2-20.1.4, respectively. In the last part of my dissertation (Part V), I evaluate the ProPAn method by showing how it can be used to assist privacy impact assessments and by applying it to a second case study. The chapters of this part are summarized in Section 20.1.5.

20.1.1. Foundations

In Chapter 1, I motivated my thesis in Section 1.1, I introduced the research questions that I aimed to answer in my thesis in Section 1.2, I summarized the contributions of my thesis in Section 1.3 and related them to the research questions. An overview of my dissertation was given in Section 1.4, and the list of publications on which my thesis is based was given in Section 1.5.

After the introduction to my thesis, I presented the background of my thesis in Chapter 2. In this chapter, I introduced basic requirements engineering terminology in Section 2.1, the problem frames approach in Section 2.2, the tool support for the problem frames approach (called UDEPF) on which I based the tool support of the ProPAn method in Section 2.3, and the definition of privacy used in this thesis in Section 2.4. The UDEPF tool allows to create context, problem, and domain knowledge diagrams, and to store them in a so called problem frame model.

To assess the existing privacy requirements engineering methods and potential research gaps in the field, I performed a literature review presented in Chapter 3. To compare the identified methods, I additionally proposed comparison criteria and a high-level privacy requirements engineering method in Chapter 3. The high-level privacy requirements engineering method consists of the core steps, artifacts, and roles that I identified to exist in privacy requirements engineering methods.

In Chapter 4, I introduced an electronic health system that is concerned with the management of electronic health records (EHRs) of patients. These EHRs shall be accessible by doctors, the accounting and billing of patients shall be possible based on the EHRs, mobile devices of patients shall be used to record vital signs of them in the EHRs and to inform patients about alarms, instructions and appointments, and the EHRs shall be provided in an anonymized form to researchers for clinical research purposes. I used the electronic health system as a running example in Chapters 9-13, 17, and 18 to illustrate the application of the steps of the ProPAn method presented in the respective chapter.

20.1.2. Privacy Requirements Taxonomy

Before I introduced my privacy requirements taxonomy refining the data protection goals proposed by Hansen et al. (2015) in Chapter 7, I refined the protection goals transparency and intervenability in Chapters 5 and 6 to more fine grained privacy requirements. For this refinement, I used the international standard ISO 29100 (ISO/IEC, 2011) and the EU General Data Protection Regulation (GDPR) (European Commission, 2016) as sources for transparency and intervenability requirements. My complete taxonomy of privacy requirements, including a formalization of these in the form of an EMF metamodel and semantics for the privacy requirements in the form of text templates, was presented in Chapter 7. My privacy requirements taxonomy bridges the gap between privacy protection needs emerging from or defined by legislation, standards, best practices, and data subjects' needs on one side, and more technical privacy requirements needed to formulate the privacy properties a software-based system shall have on the other side.

20.1.3. ProPAn

I provided an overview of the ProPAn method's steps, and the used and produced artifacts and their relations in Chapter 8. The ProPAn method starts with the set of functional requirements that shall be satisfied when the machine is integrated into the system. These functional requirements are expected to be given as problem diagrams in a problem frame model.

Chapter 9 introduced the first step of the ProPAn method that is concerned with eliciting and modeling privacy relevant knowledge about the (indirect) environment of the system-to-be.

ProPAn's privacy threshold analysis was presented in Chapter 10. The aim of the threshold analysis is to decide whether a detailed privacy analysis is necessary or not. For this, the personal data that might be processed by the system-to-be are derived from the functional requirements and the elicited domain knowledge, and the potential data flows due to the functional requirements and the domain knowledge are visualized as a data flow graph.

If a detailed privacy analysis was identified to be necessary, the step Data Flow Analysis of the ProPAn method needs to be performed. This step was introduced in Chapter 11. During the data flow analysis, the analysis team elicits systematically which personal data actually flow between which domains, in which amount they flow, for which instances of the target domains the personal data shall be accessible, how long these data are available at the target domains, and to which other personal data available at the target domain these shall be linkable.

In Chapter 12, I described how the privacy requirements of my taxonomy can automatically be derived from the information elicited and documented during the data flow analysis. Furthermore, I explained how the automatically generated privacy requirements may be adjusted, and how the adjusted privacy requirements can be validated. The validation checks whether an adjusted privacy requirement is itself consistent, whether there are inconsistencies between the privacy requirements, and whether the privacy requirements are consistent to the information elicited and modeled during the privacy threshold analysis and data flow analysis.

Having identified the privacy requirements for the system-to-be, it has to be assessed under which circumstances these could be violated. I introduced in Chapter 13 the privacy risk analysis step of the ProPAn method. During this step, deviations of the functional requirements of the system-to-be are assessed to identify situations that might violate the privacy requirements. The locally identified situations and scenarios leading to these are then modeled in a global threat model. This model is used to consistently evaluate the likelihood and consequences of the situations leading to violations of the privacy requirements. This evaluation leads to the privacy risks existing in the system-to-be and that need to be addressed by technical or non-technical privacy measures if these risks are unacceptable.

20.1.4. AORE4PF

Many technical measures that address software qualities can elegantly be represented as cross-cutting functional requirements, i.e., these technical measures often need to be integrated into different functional requirements to achieve the desired software quality. Hence, I propose the usage of aspect-oriented concepts to model cross-cutting functional requirements and to integrate them into the functional requirements they cross-cut.

In Chapter 14, I proposed an extension of Jackson's problem frames approach to support aspect-oriented concepts, called AORE4PF. Based on this extension, I introduced four aspect frames that are patterns supporting requirements engineers to model often occurring cross-cutting functional requirements as aspect diagrams in Chapter 15. The aspect frames are similar to Jackson's problem frames.

To support a privacy analysis team to select technical privacy measures, I proposed to document privacy enhancing technologies (PETs) using a pattern format that I presented in Chapter 16. The instantiation of this pattern format results in PET patterns. I showed two PET patterns as examples in Chapter 16.

I presented ProPAn's step for the selection and integration of privacy measures in Chapter 17. In this step, I proposed to integrate non-technical privacy measures as assumptions, and technical requirements as (cross-cutting) functional requirements into the given problem frame model to implement the identified privacy requirements and to mitigate the privacy risks to these privacy requirements.

20.1.5. Evaluation

In Chapter 18, I showed how the ProPAn method and the artifacts produced during its execution can be used to assist conducting a privacy impact assessment (PIA) and creating a PIA report, respectively. This chapter showed that the ProPAn method can be used to perform parts of a PIA and consequently to create parts of a PIA report.

I applied the ProPAn method on a course evaluation scenario to show its applicability on a second case study. This application was presented in Chapter 19.

The current chapter (Chapter 20), concludes my thesis and gives directions for future research.

20.2. Answers to my Research Questions

RQ 1 What kinds of privacy requirements exist?

Answer to RQ 1 I created a privacy requirements taxonomy that refines the data protection goals proposed by Hansen et al. (2015) in Chapter 7. My taxonomy is presented as an EMF metamodel that can be used to instantiate concrete privacy requirements for a system-to-be. Additionally, I provide text templates that define the semantics of the proposed privacy requirements. The transparency and intervenability requirements of my taxonomy are derived from the ISO 29100 (ISO/IEC, 2011) and the GDPR (European Commission, 2016) (see Chapters 5 and 6) and hence, also reflect the privacy principles and privacy legislation provided in these sources. I further specify how the privacy requirements taxonomy is related to privacy principles by providing a mapping between them. This mapping states which privacy requirements need to be instantiated to address a specific privacy principle. Additionally, I discussed the relation of my privacy requirements taxonomy to other privacy conceptual models used in the literature (cf. Chapter 3).

Hence, I provide a set of privacy requirements that can be used to specify the needed privacy properties of a software-based system. Additionally, I provide guidance for the use of my requirements taxonomy if privacy protection needs of another privacy conceptual model are used by mapping the other privacy conceptual models' elements to my privacy requirements taxonomy.

RQ 2 Which knowledge in addition to a software's functional requirements is needed for a privacy analysis of these?

Answer to RQ 2 This research question was mainly answered in Chapter 9. In Chapter 9, I proposed a systematic method to identify and model potential data subjects and counterstakeholders in the indirect environment of the system, interfaces and relations between domains in the environment that are out of the scope of the machine, and connection domains refining interfaces in the system that might introduce privacy issues. The identification of these is supported by questionnaires.

Hence, the knowledge that is needed in addition to a software's functional requirements, is the domain knowledge concerning data subjects and counterstakeholders in the indirect environment of the system that otherwise would not have been considered, interfaces in the environment that are out of the scope of the machine, but may introduce privacy issues, and connection domains that refine interfaces of the system and might be (directly or indirectly) the source of privacy violations.

RQ 3 How can requirements engineers systematically identify the personal data the software system shall process and the data subjects of these data?

Answer to RQ 3 In Chapter 10, I addressed this research question as the first step of the privacy threshold analysis. The potential data subjects are the biddable domains documented in the problem frame model that contains the functional requirements and the domain knowledge elicited during the privacy context elicitation step. To identify the personal data of these potential data subjects that might be processed by the system, I propose to consider the phenomena of the corresponding biddable domains that a statement in the problem frame model refers to as candidates. I also described in Chapter 10 how the relation between the personal data and the corresponding data subject shall be documented in the ProPAn model.

RQ 4 How to support requirements engineers to understand how the identified personal data are processed by the software system?

Answer to RQ 4 This research question is partly answered by the privacy threshold analysis presented in Chapter 10. During the privacy threshold analysis, the information flows due to the functional requirements and the elicited privacy relevant domain knowledge are over-approximated to automatically generate a data flow graph for each data subject. These data flow graphs illustrate how the data subjects' personal data flow through the system. Consequently, it can be derived which personal data are potentially collected, stored, and provided to others by the machine.

When the analysis team decides during the privacy threshold analysis that a detailed privacy analysis is necessary, then the data flow analysis step of the ProPAn method is performed. This step was introduced in Chapter 11. During the data flow analysis, the analysis team determines the actual personal data flows between the domains of the system due to the functional requirements and elicited domain knowledge, which were over-approximated before. The ProPAn method also requires that during the elicitation of these data flows, information about the amount in which the personal data flow, for which instances of the target domains the personal data shall be accessible, how long these data are available at the target domains, and to which other personal data available at the target domain these shall be linkable. From the elicited data flows, a data flow graph can be generated for each data subject. From these data flow graphs, it can be derived which personal data are collected, stored, and provided to others by the machine.

Hence, the understanding of how personal data are processed by the system-to-be can be supported by a systematic assessment of the data flows implied by the functional requirements and domain knowledge of the system, and the generation of data flow graphs visualizing the elicited data flows for each data subject.

RQ 5 How to systematically derive a software's privacy requirements?

Answer to RQ 5 In Chapter 12, I presented a method to automatically generate privacy requirements that are instances of my privacy requirements taxonomy based on the elicited personal data flows. Furthermore, I explained how these generated privacy requirements may be adjusted, e.g., weakened, strengthened, or completed. Finally, I provided validation conditions that can automatically check the adjusted privacy requirements, their relations to each other, and their relation to the elicited personal data flows for consistency.

RQ 6 How to identify and assess the risks of threats to a software's privacy requirements?

Answer to RQ 6 In Chapter 13, I proposed a method for the identification of a system's privacy threats and the assessment of the risks implied by those. Using this method, deviations of the system's functional requirements are assessed to identify situations that might violate the documented privacy requirements. The locally identified situations and scenarios leading to these situations are then modeled in a global threat model. This model is used to consistently evaluate the likelihood and consequences of the situations leading to violations of the privacy requirements. This evaluation reveals the privacy risks existing in the system-to-be and whether these are acceptable, tolerable, or unacceptable.

RQ 7 How can requirements engineers be supported to treat privacy risks and to implement privacy requirements?

Answer to RQ 7 I proposed a systematic method to select and integrate privacy measures that treat privacy risks and implement privacy requirements in Chapter 17. Non-technical privacy measures are added as assumptions to the problem frame model, and technical privacy measures as (cross-cutting) functional requirements. To reason that the selected privacy measures sufficiently treat the privacy risks and implement the privacy requirements, I proposed to document a satisfaction argument for each privacy requirement that refers to the privacy risks violating the privacy requirement, and the selected privacy measures to address it.

To define cross-cutting functional requirements in the problem frames approach, I extended it with aspect-oriented concepts in Chapter 14. This extension, called AORE4PF, allows to specify privacy measures separately from the system-to-be, and hence, supports the reuse of the so specified privacy measures for the analysis of other systems. To assist requirements engineers to specify cross-cutting functional requirements, I proposed aspect frames that are patterns for classes of cross-cutting functional requirements in Chapter 15.

To support the selection of privacy measures, I proposed to document privacy measures, especially PETs, as so called PET patterns. I introduced the pattern format for PET patterns and two instances of it in Chapter 16. My PET pattern format follows the general guidelines that are suggested by the pattern community (Harrison, 2003; Wellhausen and Fießer, 2011). This structured presentation allows to easily assess whether a privacy measure fits to the context of the system that is analyzed, whether the privacy measure addresses the needed privacy requirements, how it positively or negatively influences privacy requirements and other non-functional requirements, and how the privacy measure has to be integrated into the system-to-be.

Hence, the treatment of privacy risks and the implementation of privacy requirements can be supported by providing a catalog of PET patterns that contains knowledge about PETs in a structured and reusable form and a systematic method to select and integrate privacy measures into a given system.

RQ 8 Can artifacts of a privacy requirements analysis be used to create the documentation of a privacy impact assessment?

Answer to RQ 8 In Chapter 18, I presented which steps of the ProPAn method can be used to implement steps of the PIA method introduced in ISO 29134 (ISO/IEC, 2017a). Moreover, I discussed which artifacts produced during the ProPAn method and documented in the ProPAn model can be used to create a PIA report or may be included in it. As a more general answer to this research question, I showed in Chapter 18 which steps of the high-level privacy requirements engineering process (proposed in Chapter 3) may support steps of the PIA method introduced in ISO 29134.

20.3. Comparison with the State of the Art

In Chapter 3, I presented criteria to compare privacy requirements engineering methods. Using these criteria, I compared 40 privacy requirements engineering methods that I identified in a literature review that is also presented in Chapter 3. To compare my ProPAn method with the other privacy requirements engineering methods, I instantiated the criteria for the ProPAn method, as I did for the other methods before. The result of the instantiation is shown in Tables 20.1 and 20.2. I also gave a condensed overview of the 40 privacy requirements engineering methods in Table 3.20 on page 59 and Table 3.21 on page 60. The corresponding condensed view on the properties of the ProPAn method is given in Table 20.3.

In the following Sections 20.3.1-20.3.7, I discuss the ProPAn method concerning the different comparison criteria.

Table 20.1.: *The ProPAn method in the context of the high-level privacy requirements engineering method (part 1)*

Criteria	ProPAn
General Criteria	
PCM	Privacy principles, privacy regulations, privacy protection goals
PNF	GDPR
Req. Not.	Problem diagrams
Mod. Lang.	Problem diagrams
Extend requirements specification and system model	
Outcome	Description of the system's indirect environment as domain knowledge diagrams, personal data and properties of these, availability of personal data at the domains of the system, data flow graph (DFG) visualizing the flows of personal data through the system
D & M	Domain knowledge diagrams, personal data diagrams, available data diagrams, and data flow graphs
Technique	Stepwise method supported by questionnaires, patterns, modeling guidelines, and a metamodel
Tool	UDEPF allows to document the domain knowledge diagrams, ProPAn tool supports the elicitation, generation and documentation of personal data diagrams, available data diagrams, and data flow graphs
Elicit privacy requirements	
Outcome	Privacy requirements of ProPAn's privacy requirements taxonomy are documented
D & M	ProPAn metamodel allowing a structured documentation of the privacy requirements
Technique	Automatic generation of privacy requirements from elicited data flows, guidelines to adjust the privacy requirements, automatic validation of adjusted privacy requirements
Tool	ProPAn tool can automatically generate privacy requirements candidates, validate adjusted privacy requirements, and store these in a ProPAn model
Elicit privacy risks	
Outcome	Threat diagrams containing threat scenarios and unwanted incidents that lead to violations of the privacy requirements, privacy risks derived from the threat diagrams
D & M	Threat diagrams, risk matrix
Technique	Threats are derived from deviations of the system's functional requirements, the local threats are then collected in a global threat model (represented by threat diagrams), risks are derived from the threat model
Tool	ProPAn tool allows to create and store threat diagrams, likelihood and consequence scales, and the risk matrix.

Table 20.2.: *The ProPAn method in the context of the high-level privacy requirements engineering method (part 2)*

Criteria	ProPAn
Refine privacy requirements	
Outcome	Privacy measures and a satisfaction argument for each privacy requirement that reasons why the measures suffice to satisfy the privacy requirement
D & M	Domain knowledge diagrams for non-technical privacy measures, problem and aspect diagrams for technical privacy measures, satisfaction arguments
Technique	PET patterns, aspect-oriented requirements engineering for cross-cutting privacy measures, aspect frames, informal reasoning for satisfaction arguments
Tool	UDEPF tool allows to model domain knowledge, problem, aspect and weaving diagrams, ProPAn tool allows to document the satisfaction arguments
Treat privacy risks	
Outcome	Privacy measures mitigating privacy risks, a satisfaction argument for each privacy requirement that reasons why the measures suffice to satisfy the privacy requirement under the light of the identified privacy risks, updated threat diagrams and risk matrix
D & M	Domain knowledge diagrams for non-technical privacy measures, problem and aspect diagrams for technical privacy measures, satisfaction arguments, threat diagrams, risk matrix
Technique	PET patterns, aspect-oriented requirements engineering for cross-cutting privacy measures, aspect frames, informal reasoning for satisfaction arguments, updating threat model with measures, re-evaluating privacy risks
Tool	UDEPF tool allows to model domain knowledge, problem, aspect and weaving diagrams, ProPAn tool allows to document the satisfaction arguments, threat diagrams, and the risk matrix

Table 20.3.: *Condensed overview of the ProPAn method*

Privacy Method	PCM	Req. Not. & Mod. Lang.	Techniques	Strat.	Op.	Ext.	Meth.	Tool
ProPAn	PG, PT, PR	Problem diagrams	Questionnaires, Patterns, Guidelines	C	+	+	+	+

PG: Protection goals, PT: Privacy threats, PR: Privacy regulations, C: Combined, +: Existing

20.3.1. Privacy Conceptual Model (PCM)

Table 20.1 shows that the ProPAn method uses as privacy conceptual models (PCMs) privacy principles, privacy regulations, and primarily privacy protection goals. In this way the ProPAn method bridges the gap between less technical requirements formulated in privacy principles and regulations, and the more technical privacy protection goals that I refined to a privacy requirements taxonomy. This supports requirements engineers to translate the privacy protection needs formulated in privacy principles and regulations to technical measures that can be integrated into a system.

20.3.2. Privacy Normative Framework (PNF)

As privacy normative framework (PNF), the ProPAn method is based on the GDPR, especially concerning the generation and validation of transparency and integrity requirements. In this way, the ProPAn method aims to support the development of software compliant to this regulation. However, I expect that the ProPAn method is also applicable in cases where other or additional privacy regulations have to be considered.

20.3.3. Notations and Languages (Req. Not. & Mod. Lang.)

Problem diagrams are used as both, requirements notation (Req. Not.) and modeling language (Mod. Lang.). This is, because problem diagrams present a combined view on the domains of the system and their interfaces, and the functional requirements of the system and their relations to the system's domains. Furthermore, additional modeling artifacts, describing both static and dynamic views on the system-to-be, are created during the ProPAn method. For example, sequence diagrams are created providing a behavioral view on how the domains of the system interact with each other to satisfy the functional requirements, and data flow graphs are generated that visualize how the personal data of data subjects are expected to flow through the system.

20.3.4. Operationalization Strategy (Strat.)

The ProPAn method supports both, the refinement-based and the prevention-based operationalization of privacy. First, the privacy requirements of the system-to-be are identified in the ProPAn method. Then risks leading to their violation are identified and evaluated. During the selection and integration of privacy measures, both the implementation of the privacy requirements and the mitigation of the privacy risks is considered. This is done by the specification of a satisfaction argument for each privacy requirement that reasons why the requirement is sufficiently refined and why the identified privacy risks are sufficiently prevented.

20.3.5. Detail of Methodology and Methodological Support (Meth. and Techniques)

All five tasks of the high-level privacy requirements engineering method are covered by the ProPAn method and supported by techniques. For each step of the ProPAn method, I specified a stepwise method with defined procedure, inputs, and outputs. To support the procedure and the creation of the needed outputs, I provide questionnaires, elicitation templates, patterns, metamodels, and modeling guidelines. Furthermore, several steps are automated by the ProPAn tool (cf. Section 20.3.7).

No other state of the art privacy requirements engineering method provides an as detailed methodology and as much methodology support as the ProPAn method does.

20.3.6. Extend requirements specification and system model (Ext.)

The ProPAn method needs as input only the functional requirements of the system that shall be analyzed. These functional requirements need to be presented as problem diagrams in a problem frame model. During the first three steps of the ProPAn method, this requirements specification and the system model contained in the problem diagrams is extended.

During the Privacy Context Elicitation (see Chapter 9), the scope of the system described by the functional requirements is widened to identify possible indirect data subjects and counterstakeholders, to identify interfaces between domains of the machines environment, and to identify connection domains that refine interfaces already documented in the problem diagrams.

If the identified additional elements are possibly relevant for the further privacy analysis, they are added to the problem frame model in the form of domain knowledge diagrams. In this way, the system description is extended with privacy-relevant context information.

During the Privacy Threshold Analysis (see Chapter 10), the personal data processed by the system are identified based on the biddable domains documented in the problem frame model and the statements (functional requirements and domain knowledge) referring to phenomena of these biddable domains. Furthermore, the flow of personal data through this step is over-approximated based on the documented statements and visualized as a data flow graph. From this data flow graph, it can also be over-approximated which personal data are collected, stored, and provided to other domains by the machine.

During the Data Flow Analysis (see Chapter 11), the actually intended flow of personal data through the system is elicited and documented. Together with the information that personal data flow to a domain, it is documented in which amount the personal data shall be available at the domain, which instances of the domain shall be able to access the data, in which amount the personal data flow, how long the personal data are retained at the domain, and whether personal data are linkable to each other at the domain. From this information, data flow graphs can automatically be generated that visualize the flow of a data subject's personal data through the system. Furthermore, it can be derived which personal data are collected, stored, and provided to other domains by the machine from these graphs.

The ProPAn method provides more support for the extension of a given requirements specification and system model than any other privacy requirements engineering method. The other methods mostly consider the information elicited in the previously mentioned steps as an input, e.g., the data flow graphs and the personal data that are processed by the system, or do not consider this information at all, e.g., the indirect environment of the system is out of the scope of most methods. Some of the outputs produced during the previously mentioned steps, e.g., the data flow graphs, might be used as an input for privacy requirements engineering methods in cases where the needed input is not available and has to be created.

20.3.7. Tool Support

The aim of the ProPAn method, as computer-aided privacy requirements engineering method, is to provide as much tool support for its steps as possible. For each step of the ProPAn method, I provide tool support. This tool support consists of the UDEPF tool for the creation of context, problem, domain knowledge, and aspect diagrams, and the ProPAn tool to assist the analysis team to perform the ProPAn method's steps, including the creation of the needed artifacts. To document the artifacts and to make them machine-readable, I created metamodels that specify the relations between the artifacts and their attributes. The different metamodels contain traceability links that document and allow to later reconstruct why specific artifacts were created and they are related to each other.

The ProPAn tool not only allows the creation of the needed artifacts, it also automates and guides several steps of the ProPAn method. This automation is possible because of the machine-readable nature of the artifacts produced using the ProPAn tool. For example, the ProPAn tool guides the analysis team through the execution of the data flow analysis. Furthermore, the ProPAn tool can automatically generate data flow graphs, privacy requirements, and derive which personal data are collected, stored, and provided to others by the machine. The ProPAn tool can also automatically validate adjusted privacy requirements for their consistency, the consistency between the privacy requirements, and the consistency between the privacy requirements and the documented information concerning the flow of personal data through the system.

With the UDEPF and ProPAn tools, the ProPAn method provides more tool support than any other privacy requirements engineering method, as the ProPAn tool covers all steps of the method and goes beyond the creation of needed artifacts by providing automatic and semi-automatic functionalities.

20.4. Future Work

In this section, I present possible future work in the context of my thesis. An industrial application of the ProPAn method is discussed in Section 20.4.1. The extension of the ProPAn method to the subsequent tasks of the software development process is elaborated in Section 20.4.2. I discuss the applicability of the ProPAn method in an agile software development process in Section 20.4.3. In Section 20.4.4, I consider the combination of ProPAn with requirements engineering methods for other software qualities.

20.4.1. Industrial Application of the ProPAn Method

The ProPAn method was yet only developed and applied in the scientific context. In future work, the ProPAn method should be applied in an industrial setting. Only in a realistic industrial setting with practical engineers, the applicability of the ProPAn method can be evaluated.

As a result of the application of the ProPAn method by engineers, I expect feedback concerning the usability and overhead of the ProPAn method, especially in comparison to the benefits it brings to the engineers. These results can be used to make the ProPAn method more light-weight in the sense that the effort to perform the method and the benefit it provides are balanced.

It can furthermore be investigated which other artifacts can be used as inputs to the ProPAn method if such additional inputs are found to be available at the beginning of a privacy requirements analysis. Similarly, it can be investigated which additional outputs may be produced by the ProPAn method, e.g., by producing additional artifacts, extending existing artifacts, and by providing other views on the existing artifacts. Additionally, the use of parts of the ProPAn method and its artifacts might be possible instead of applying the whole method.

As the ProPAn tool is yet a research prototype, it can be further extended and improved, especially concerning its usability.

To support the selection and integration of privacy measures, a catalog of PET patterns could be created. This catalog could be made available online, similar to other privacy pattern catalogs[1]. It could further be investigated how these catalogs can be combined or merged.

It is planned that some of the above mentioned points are investigated as part of the EU Horizon 2020 innovation action PDP4E[2].

20.4.2. Extension to Subsequent Tasks of the Software Development Life Cycle

The ProPAn method only covers the requirements engineering phase of the software development life cycle. Hence, it could be investigated how the results can be used to derive system and software architectures that further refine and implement the selected privacy measures. Additionally, it could be investigated how it can be tested whether or even be verified that the developed system satisfies its privacy requirements. For this, metrics need to be specified that allow to measure and formalize to which degree the system satisfies its privacy requirements. A first step to address this issue are the likelihood and consequence scales that are specified in the risk analysis step of the ProPAn method (see Chapter 13). However, these scales may refer to phenomena that are not directly observable and measurable in (automatic) test cases or can hardly be formalized.

[1] For example, `https://privacypatterns.eu` (accessed on 30 August 2018)

[2] `https://www.pdp4e-project.eu/` (accessed on 30 August 2018)

20.4.3. Usage of ProPAn in an Agile Context

Agile software development (Beck, 2000) aims at solving issues of traditional software development processes and their performance. The main principles of agile software development are documented in the agile manifesto[3]. Agile software development aims at a fast delivery of working software, which is contrary to an extensive requirements analysis as performed by the problem frames approach and the ProPAn method.

To reach a fast delivery of working software, the only consistent documentation of the software development process are the requirements in the form of user stories, and the source code of the software and its test cases. That is, models may be created by the agile development team, but these models are not maintained and not kept consistent through the further development. However, under the light of the GDPR, a documentation concerning the personal data processing of the system-to-be has to be created if the developed software shall to comply to it.

To address this documentation needs, a respective documentation needs either to be created after the software development, or as part of it. The first option may be risky, because late changes in the developed software were reported to be expensive (Buchan et al., 2009). For the second option, documentation and privacy analysis tasks would need to be added to the agile development process. In future research it could be assessed which privacy (requirements) engineering tasks can be performed at which points in the agile development process.

A more detailed discussion about the challenges and opportunities when (privacy) threat modeling and agile software development shall be combined is provided by Galvez and Gürses (2018).

20.4.4. Combination of ProPAn with Requirements Engineering Methods for Other Software Qualities

Some method steps and artifacts of the ProPAn method are not only relevant for a privacy analysis, but may also be valuable for assessing the security, safety, or even other software qualities of a system.

Together with Azadeh Alebrahim and Maritta Heisel, I generalized the context elicitation step of the ProPAn method and explained how such a step can be developed for other software qualities (Alebrahim et al., 2014).

The ProCOR method (Wirtz et al., 2018) is a security threat analysis method. This method contains a step to elicit how data flows through the system. This step is similar to ProPAn's data flow analysis. Hence, it could be assessed how these methods could be combined and complement each other.

In future work, it may also be investigated whether ProPAn's automatic privacy requirements generation and validation may be adapted for the generation of requirements related to other software qualities.

With PRIOP, I proposed a method to identify privacy threats from deviations of the system's functional requirements in Chapter 13. This method is an adaption of HAZOP (IEC, 2001), which is originally a method to identify hazards, i.e., threats to the software quality safety. Hence, the consideration of deviations of the system's functional requirements may also be used to identify threats to further software qualities.

Similar to the PET patterns proposed in Chapter 16, it could be investigated how patterns for techniques addressing other software qualities could be structured and which properties of these are import to be documented. The pattern format for PET patterns already proposes to consider the impact of privacy measures on other software qualities as general forces, benefits and liabilities. This list of software qualities may also be extended in future work.

[3] `http://agilemanifesto.org/` (accessed on 30 August 2018)

Part VI.

Appendix

Appendix A

OCL Expressions to Validate Privacy Requirements

In this chapter, I provide the overview of all validation conditions for privacy requirements that I introduced in Section 12.4 and I provide a formalization of these as OCL invariants. Table A.1 provides an overview of all validation conditions and also references for each validation conditions the listing in which it is formalized.

Table A.1.: *Overview of all validation conditions for privacy requirements*

No.	Requirement	Condition	Type	OCL
VP1	Privacy requirement	The data referenced by a privacy requirement shall be personal data of the mentioned data subject.	Error	Listing A.1
VP2	Privacy requirement	All necessary attributes of a privacy requirement have to be set.	Error	meta-model
VSC1	Confidentiality requirement	A confidentiality requirement with availability none shall not have a repudiation type different from none.	Error	Listing A.2
VSC2	Confidentiality requirement	For each instance of a subclass of confidentiality requirement and each combination of (linkability,) data subject, personal data, and counterstakeholder, there shall be at most one instance for each availability degree.	Error	Listing A.3
VSC3	Confidentiality requirement	For each confidentiality requirement with availability none, there shall not be another confidentiality requirement of the same type concerning the same personal data, data subject, and counterstakeholder.	Error	Listing A.4
VSC4	Confidentiality requirement	For each confidentiality requirement with availability all and repudiation nonRepudiation, there shall not be another confidentiality requirement of the same type concerning the same personal data, data subject, and counterstakeholder.	Error	Listing A.5

VSC5	Confidentiality requirement	For all combinations of data subject ds, counterstakeholder c, and availability a, check whether each personal data object of ds is referenced by at least one of the confidentiality requirements for ds, c, and a.	Warning	Listing A.6
VSD1	Data confidentiality requirement	For each data confidentiality requirement that specifies that to certain counterstakeholders personal data shall be available, there shall be undetectability requirements that specify that the counterstakeholders are allowed to know about the existence of the personal data.	Error	Listing A.7
VSD2	Data confidentiality requirement	For each undetectability and data confidentiality requirement, the personal data referenced by these shall not be available at the referenced counterstakeholders.	Error	Listing A.8
VSA1	Availability requirement	For each availability requirement, there shall be data confidentiality requirements that permit the access for data subjects to the personal data listed by the availability requirement.	Error	Listing A.9
VSA2	Availability requirement	All personal data of a data subject that are processed by the machine have to be contained in an availability requirement of the data subject.	Error	Listing A.10
VSA3	Availability requirement	All personal data contained in an availability requirement have to be processed by the machine.	Error	Listing A.11
VSI1	Integrity requirement	All personal data of a data subject that are processed by the machine have to be contained in an integrity requirement of the data subject.	Error	Listing A.12
VSI2	Integrity requirement	All personal data contained in an integrity requirement have to be processed by the machine.	Error	Listing A.13
VU1	Unlinkability requirement	An unlinkability requirement shall not have the availability degree none.	Error	Listing A.14
VU2	Unlinkability requirement	An unlinkability requirement that does not allow linkage between personal data shall not specify a repudiation type.	Error	Listing A.14
VU3	Unlinkability requirement	For each unlinkability requirement there shall be data confidentiality requirements that specify that the personal data may be available to the counterstakeholders.	Error	Listing A.16
VUD1	Data unlinkability requirement	The personal data referenced by a data unlinkability requirement shall be equal to the set of data referenced by the associated links.	Error	Listing A.17

VUD2	Data unlinkability requirement	For each data unlinkability requirement, all pairs of personal data referenced by the requirement shall be linkable with a linkability weaker than or equal to the linkability of the data unlinkability requirement for all listed counterstakeholders.	Error	Listing A.18
VUD3	Data unlinkability requirement	For all combinations of data subject ds and counterstakeholder c, and availability a, check whether each link between personal data of ds that are available to a counterstakeholder is referenced by at least one data unlinkability requirements for ds, c, and a.	Warning	Listing A.19
VUD4	Data unlinkability requirement	For each data unlinkability requirement, all pairs of personal data referenced by the requirement should be linkable with a linkability stronger than or equal to the linkability of the data unlinkability requirement for all listed counterstakeholder.	Warning	Listing A.20
VUA1	Anonymity requirement	The linkability specified by an anonymity requirement must not be weaker than the weakest linkability specified by a related to relation for the data subject and personal data of the anonymity requirement.	Error	Listing A.21
VUA2	Anonymity requirement	For every anonymity requirement, all personal data referenced by it have to be linkable to the data subject with a linkability weaker than or equal to the linkability mentioned in the anonymity requirement for all the listed counterstakeholders.	Error	Listing A.22
VUA3	Anonymity requirement	For each anonymity requirement, all personal data referenced by it should be linkable to the data subject with a linkability stronger than or equal to the linkability mentioned in the anonymity requirement for all listed counterstakeholder.	Warning	Listing A.23
VUP1	Pseudonymity requirement	A pseudonymity requirement shall not have the availability degree none.	Error	Listing A.24
VUP2	Pseudonymity requirement	For each pseudonymity requirement there shall not be an anonymity requirement for the same data subject and availability degree that (partially) shares personal data and counterstakeholders.	Error	Listing A.25
VTP1	Processing information requirement	For each processing information requirement whose grounds include consent, there shall be data subject intervention requirements associated to the processing information requirement with the intervention types doNotConsent and withDrawConsent.	Error	Listing A.26

VTP2	Processing information requirement	All personal data referenced by a collection or storage information requirement have to be available at a designed domain.	Error	Listing A.27
VTP3	Processing information requirement	If a processing information requirement does not include consent as processing ground, it has to list contract, regulation, vital interest, public interest, controller interest, or legal claims as processing ground.	Error	Listing A.28
VTP4	Processing information requirement	If a processing information requirement has as control options data subject intervention requirements with type doNotConsent and withdrawConsent, then its processing grounds have to include consent.	Warning	Listing A.29
VTP5	Processing information requirement	For each storage and flow information requirement about personal data that the controller can uniquely link to the data subject, the requirement has to have a data subject intervention requirement with type access as control option.	Error	Listing A.30
VTP6	Processing information requirement	For each storage and flow information requirement about personal data that the controller can uniquely link to the data subject, the requirement has to have data subject intervention requirements with types challengeAccuracy and challengeCompleteness as control options.	Error	Listing A.31
VTP7	Processing information requirement	For each processing information requirement about personal data that the controller can uniquely link to the data subject and whose grounds include neither regulation, public interest, nor legal claims, the requirement has to have a data subject intervention requirement with type object and effect erasure as control option.	Error	Listing A.32
VTP8	Processing information requirement	For each processing information requirement about personal data that the controller can uniquely link to the data subject, the requirement has to have a data subject intervention requirement with type object and effect noProcessing or restrictedProcessing as control option.	Error	Listing A.33
VTC1	Collection information requirement	There shall be at most one collection information requirement for each combination of data subject, personal data, and purpose.	Error	Listing A.34
VTC2	Collection information requirement	For all personal data flows from a given domain to a designed domain, a corresponding collection information requirement shall exist.	Error	Listing A.35

VTC3	Collection information requirement	For each collection information requirement about personal data that the controller can uniquely link to the data subject and whose grounds do not include public interest, the requirement has to have a data subject intervention requirement with type requestDataCopy as control option.	Error	Listing A.36
VTF1	Flow information requirement	There shall be at most one flow information requirement for each combination of data subject, personal data, purpose, and target.	Error	Listing A.37
VTF2	Flow information requirement	All personal data referenced by a flow information requirement have to be available at the target of the personal data flow.	Error	Listing A.38
VTF3	Flow information requirement	For all personal data flows from a designed domain to a given domain, or between two given domains due to a functional requirement, a corresponding flow information requirement shall exist.	Error	Listing A.39
VTS1	Storage information requirement	There shall be at most one storage information requirement for each combination of data subject, personal data, and purpose.	Error	Listing A.40
VTS2	Storage information requirement	For all personal data available at a designed domain, a corresponding storage information requirement shall exist.	Error	Listing A.41
VTE1	Exceptional information requirement	There shall be at most one exceptional information requirement for each combination of data subject, personal data, case, and authority.	Error	Listing A.42
VTE2	Exceptional information requirement	For each personal data of a data subject, an exceptional information requirement with case dataBreach has to exist for the data subject and the personal data.	Error	Listing A.43
VTE3	Exceptional information requirement	For each personal data of a data subject, an exceptional information requirement with case authorityRequest has to exist for the data subject and has to include the personal data.	Error	Listing A.44
VTE4	Exceptional information requirement	For each exceptional information requirement with case authorityRequest, its authority intervention requirements have to include an authority intervention requirement with type obtainAccess.	Error	Listing A.44
VTE5	Exceptional information requirement	For each personal data of a data subject, an exceptional information requirement with case nonCompliance has to exist for the data subject and has to include the personal data.	Error	Listing A.44

VTE6	Exceptional information requirement	For each exceptional information requirement with case nonCompliance, its authority intervention requirements have to include authority intervention requirements with the types suspendDataFlows, orderBanOfProcessing, orderErasure, and orderRectification.	Error	Listing A.44
VID1	Data subject intervention requirement	For each data subject intervention requirement, the combinations of intervention types and intervention effects have to comply to Table 6.4 on page 92.	Error	Listing A.45
VID2	Data subject intervention requirement	Each data subject intervention requirement with intervention type review, challengeAccuracy, and challengeCompleteness has to be a control option of a flow or storage information requirement.	Error	Listing A.46
VID3	Data subject intervention requirement	Each data subject intervention requirement that is a control option of a collection information requirement has to have a time that is smaller than or equal to beforeCollection.	Error	Listing A.47
VID4	Data subject intervention requirement	Each data subject intervention requirement that is a control option of a flow information requirement has to have a time that is smaller than or equal to beforeTransmission.	Error	Listing A.47
VID5	Data subject intervention requirement	Each data subject intervention requirement that is a control option of a storage information requirement has to have a time that is smaller than or equal to atRecording.	Error	Listing A.47
VID6	Data subject intervention requirement	Each data subject intervention requirement with type review has to be a control option of a storage or flow information requirement.	Error	Listing A.48
VID7	Data subject intervention requirement	Each data subject intervention requirement with type review has to have the time anyTime.	Error	Listing A.48
VID8	Data subject intervention requirement	Each data subject intervention requirement with type challengeAccuracy or challengeCompleteness has to be a control option of a storage or flow information requirement.	Error	Listing A.49
VID9	Data subject intervention requirement	Each data subject intervention requirement with type challengeAccuracy or challengeCompleteness has to have the time anyTime.	Error	Listing A.49
VID10	Data subject intervention requirement	Each data subject intervention requirement with type withdrawConsent, object, or requestDataCopy has to have the time anyTime.	Error	Listing A.50

VID11	Data subject intervention requirement	For each data subject intervention requirement with type withdrawConsent that is a control option of a processing information requirement whose grounds include neither contract, regulation, vital interest, public interest, controller interest, nor legal claims, the effects have to include erasure.	Error	Listing A.51
VIA1	Authority intervention requirement	For each authority intervention requirement, the combinations of intervention types and intervention effects have to comply to Table 6.3 on page 92.	Error	Listing A.52

Listing A.1: *VP1 expressed as OCL invariant for the class PrivacyRequirement*

```
context PrivacyRequirement
inv VP1: self.personalData→forAll(pd|
           self.dataSubject.relatedtos→exists(rt|rt.personalData = pd))
```

Listing A.2: *OCL invariant VSC1 for the class ConfidentialityRequirement*

```
context ConfidentialityRequirement
inv VSC1: self.availability = AvailabilityDegree::none implies
        self.repudiation = Repudiation::none
```

Listing A.3: *OCL invariant VSC2 for the class ConfidentialityRequirement*

```
context ConfidentialityRequirement
inv VSC2: not ConfidentialityRequirement.allInstances()→exists(cr|
            cr <> self and cr.oclType() = self.oclType and
            if self.oclIsKindOf(UnlinkabilityRequirement) then
              cr.linkability = self.linkability
            else
              true
            endif and
            cr.dataSubject = self.dataSubject and
            cr.availability = self.availability and
            cr.personalData→intersection(cr.personalData)→notEmpty() and
            cr.counterstakeholders→intersection(
              cr.counterstakeholders)→notEmpty())
```

Listing A.4: *OCL invariant VSC3 for the class DataConfidentialityRequirement*

```
context ConfidentialityRequirement
inv VSC3: self.availability = AvailabilityDegree::none implies
            not ConfidentialityRequirement.allInstances()→exists(cr|
              cr <> self and cr.oclType() = self.oclType() and
              cr.dataSubject = self.dataSubject and
              cr.personalData→intersection(cr.personalData)→notEmpty() and
              cr.counterstakeholders→intersection(
                cr.counterstakeholders)→notEmpty())
```

Listing A.5: *OCL invariant VSC4 for the class DataConfidentialityRequirement*

```
context ConfidentialityRequirement
inv VSC4: (self.availability = AvailabilityDegree::all and
            self.repudiation = Repudiation::nonRepudiation) implies
            not ConfidentialityRequirement.allInstances()→exists(cr|
          cr <> self and cr.oclType() = self.oclType() and
          cr.dataSubject = self.dataSubject and
          cr.personalData→intersection(cr.personalData)→notEmpty() and
          cr.counterstakeholders→intersection(
            cr.counterstakeholders)→notEmpty())
```

Listing A.6: *OCL specification of the validation condition VSC5 on the class* ConfidentialiltyRequirement

```
context ConfidentialiltyRequirement
inv VSC5:
Person.allInstances()→forAll(ds, cs|
  ds.relatedtos.personalData→forAll(p|
    ds.privacyrequirements→exists(pr|
      pr.oclIsKindOf(ConfidentialiltyRequirement) and
      pr.personalData→includes(p) and
      pr.counterstakeholders→includes(cs))))
```

Listing A.7: *OCL invariant VU3 for the class* DataConfidentialityRequirement

```
context DataConfidentialityRequirement
inv VSD1: self.availability <> AvailabilityDegree::none implies
            self.personalData→forAll(pd|
              self.counterstakeholders→forAll(cs|
                UndetectabilityRequirement.allInstances()→exists(ur|
                ur.dataSubject = self.dataSubject and
                ur.personalData→includes(pd) and
                ur.counterstakeholders→includes(cs) and
                (ur.availability = self.availability or
                  ur.availability = AvailabilityDegree::all))))
```

Listing A.8: *OCL specification of the validation condition VSD2 on the class* ConfidentialityRequirement

```
context ConfidentialityRequirement
inv VSD2: (self.oclIsTypeOf(UndetectabilityRequirement) or
    self.oclIsTypeOf(DataConfidentialityRequirement)) implies
  self.personalData→forAll(p|
    self.counterstakeholders.pdavailableats→select(pd|
      pd.availability = self.availability).personalData→excludes(p))
```

Listing A.9: *OCL invariant VSA1 for the class* AvailabilityRequirement *constraining the consistency of confidentiality requirements to availability requirements*

```
context AvailabilityRequirement
inv VSA1: DataConfidentialityRequirement.allInstances()
            →select(dcr|dcr.dataSubject = self.dataSubject and
              dcr.counterstakeholders→includes(self.dataSubject) and
              (availability = AvailabilityDegree::individual or
              availability = AvailabilityDegree::all))
            .personalData→includesAll(self.personalData)
```

Listing A.10: *OCL specification of the validation condition VSA2 on the class* AvailabilityRequirement

```
context AvailabilityRequirement
inv VSA2:
self.dataSubject.processedPersonalData→forAll(p|
  self.dataSubject.privacyrequirements→select(pr|
    pr.oclIsTypeOf(AvailabilityRequirement))→exists(ar|
        ar.personalData→includes(p)))
```

Listing A.11: *OCL specification of the validation condition VSA3 on the class* AvailabilityRequirement

```
context AvailabilityRequirement
inv VSA3:
self.dataSubject.processedPersonalData→forAll(d|
  self.dataSubject.privacyrequirements→select(pr|
    pr.oclIsTypeOf(AvailabilityRequirement))→exists(ar|
        ar.personalData→includes(d)))
```

Listing A.12: *OCL specification of the validation condition VSI1 on the class* IntegrityRequirement

```
context IntegrityRequirement
inv VSI1:
self.dataSubject.processedPersonalData→forAll(p|
  self.dataSubject.privacyrequirements→select(pr|
    pr.oclIsTypeOf(IntegrityRequirement))→exists(ir|
        ir.personalData→includes(p)))
```

Listing A.13: *OCL specification of the validation condition VSI2 on the class* IntegrityRequirement

```
context IntegrityRequirement
inv VSI2:
self.dataSubject.processedPersonalData→forAll(d|
  self.dataSubject.privacyrequirements→select(pr|
    pr.oclIsTypeOf(IntegrityRequirement))→exists(ir|
        ir.personalData→includes(d)))
```

Listing A.14: *OCL invariant VU1 for the class* UnlinkabilityRequirement

```
context UnlinkabilityRequirement
inv VU1: self.availability <> AvailabilityDegree::none
```

Listing A.15: *OCL invariant VU2 for the class* UnlinkabilityRequirement

```
context UnlinkabilityRequirement
inv VU2: self.linkability = Linkability::anonymous implies
        self.repudiation = Repudiation::none
```

Listing A.16: *OCL invariant VU3 for the class* UnlinkabilityRequirement

```
context UnlinkabilityRequirement
inv VU3: self.personalData→forAll(pd|
              self.counterstakeholders→forAll(cs|
                DataConfidentialityRequirement.allInstances()→exists(dcr|
                dcr.dataSubject = self.dataSubject and
                dcr.personalData→includes(pd) and
                dcr.counterstakeholders→includes(cs) and
                (dcr.availability = self.availability or
                  dcr.availability = AvailabilityDegree::all))))
```

Listing A.17: *OCL invariant VUD1 for the class* DataUnlinkabilityRequirement *constraining the consistency of attributes* links *and* personalData

```
context DataUnlinkabilityRequirement
inv VUD1: self.personalData = self.links.data→asSet()
```

Listing A.18: *OCL specification of the validation condition VUD2 on the class* DataUnlinkabilityRequirement

```
context DataUnlinkabilityRequirement
inv VUD2:
  self.links→forAll(l|
    self.counterstakeholders.linkableClosureMax(self.dataSubject)→forAll(t|
      l.data = t.link implies self.linkability.min(t.l) = t.l))
```

Listing A.19: *OCL specification of the validation condition VUD3 on the class DataUnlinkabilityRequirement*

```
context DataUnlinkabilityRequirement
inv VUD3:
Person.allInstances()→forAll(ds,cs|
  cs.pdavailableats.personalData→forAll(p1,p2|
    ds.privacyrequirements→exists(pr|
      pr.oclIsKindOf(DataUnlinkabilityRequirement) and
      pr.links→exists(l|l.data = Set{p1, p2}) and
      pr.counterstakeholders→includes(cs))))
```

Listing A.20: *OCL specification of the validation condition VUD4 on the class DataUnlinkabilityRequirement*

```
context DataUnlinkabilityRequirement
inv VUD4:
  self.links→forAll(l|
    self.counterstakeholders.linkableClosureMax(self.dataSubject)→forAll(t|
      l.data = t.link implies self.linkability.max(t.l) = t.l))
```

Listing A.21: *OCL invariant VUA1 for the class AnonymityRequirement*

```
context AnonymityRequirement
inv VUA1: self.personalData→forAll(pd|
            RelatedTo.allInstances()→select(rt|
              rt.dataSubject = self.dataSubject and
                rt.personalData = pd)→forAll(rt|
                  self.linkability.value <= rt.linkability.value))
```

Listing A.22: *OCL specification of the validation condition VUA2 on the class AnonymityRequirement*

```
context AnonymityRequirement
inv VUA2:
  self.personalData→forAll(p|
    self.counterstakeholders.relatedClosureMax(self.dataSubject)→forAll(t|
      t.d = p implies self.linkability.min(t.l) = t.l))
```

Listing A.23: *OCL specification of the validation condition VUA3 on the class AnonymityRequirement*

```
context AnonymityRequirement
inv VUA3:
  self.personalData→forAll(p|
    self.counterstakeholders.relatedClosureMax(self.dataSubject)→forAll(t|
      t.d = p implies self.linkability.max(t.l) = t.l))
```

Listing A.24: *OCL invariant VUP1 for the class PseudonymityRequirement*

```
context PseudonymityRequirement
inv VUP1: self.availability <> AvailabilityDegree::none
```

Listing A.25: *OCL invariant VUP2 for the class PseudonymityRequirement*

```
context PseudonymityRequirement
inv VUP2: self.personalData→forAll(pd|
             self.counterstakeholders→forAll(cs|
                not AnonymityRequirement.allInstances()→exists(ar|
                   ar.dataSubject = self.dataSubject and
                   ar.personalData→includes(pd) and
                   ar.counterstakeholders→includes(cs) and
                   ar.availability = self.availability)))
```

Listing A.26: *OCL invariant VTP1 for the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP1: self.grounds→includes(ProcessingGrounds::consent) implies
              (self.controlOptions.type→includes(
                 DataSubjectIntervention::doNotConsent) and
               self.controlOptions.type→includes(
                 DataSubjectIntervention::withDrawConsent))
```

Listing A.27: *OCL specification of the validation condition VTP2 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP2:
(self.oclIsTypeOf(CollectionInformationRequirement) or
    self.oclIsTypeOf(StorageInformationRequirement)) implies
self.personalData.pdavailableats→exists(pd|pd.domain.domain.designed)
```

Listing A.28: *OCL specification of the validation condition VTP3 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP3:
self.grounds→excludes(ProcessingGrounds::consent) implies
  (self.grounds→includes(ProcessingGrounds::contract) or
   self.grounds→includes(ProcessingGrounds::regulation) or
   self.grounds→includes(ProcessingGrounds::vitalInterest) or
   self.grounds→includes(ProcessingGrounds::publicInterest) or
   self.grounds→includes(ProcessingGrounds::controllerInterest) or
   self.grounds→includes(ProcessingGrounds::legalClaims))
```

Listing A.29: *OCL specification of the validation condition VTP4 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP4:
self.controlOptions→exists(ir1, ir2|
   ir1.type→includes(DataSubjectIntervention::doNotConsent) and
   ir2.type→includes(DataSubjectIntervention::withdrawConsent)) implies
self.grounds→includes(ProcessingGrounds::consent)
```

Listing A.30: *OCL specification of the validation condition VTP5 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP5:
(self.identifiableData and (self.oclIsTypeOf(StorageInformationRequirement) or
   self.oclIsTypeOf(FlowInformationRequirement))) implies
self.controlOptions→exists(ir|
   ir.type→includes(DataSubjectIntervention::access))
```

Listing A.31: *OCL specification of the validation condition VTP6 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP6:
(self.identifiableData and (self.oclIsTypeOf(StorageInformationRequirement) or
  self.oclIsTypeOf(FlowInformationRequirement))) implies
self.controlOptions→exists(ir1, ir2|
  ir1.type→includes(DataSubjectIntervention::challengeAccuracy) and
  ir2.type→includes(DataSubjectIntervention::challengeCompleteness))
```

Listing A.32: *OCL specification of the validation condition VTP7 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP7:
(self.identifiableData and
  self.grounds→intersection(Set{ProcessingGrounds::regulation,
    ProcessingGrounds::publicInterest,
        ProcessingGrounds::legalClaims}).isEmpty()) implies
self.controlOptions→exists(ir|
  ir.type→includes(DataSubjectIntervention::object) and
  ir.effect→includes(InterventionEffect::erasure))
```

Listing A.33: *OCL specification of the validation condition VTP8 on the class ProcessingInformationRequirement*

```
context ProcessingInformationRequirement
inv VTP8:
self.identifiableData implies
self.controlOptions→exists(ir|
  ir.type→includes(DataSubjectIntervention::object) and
  (ir.effect→includes(InterventionEffect::noProcessing) or
    ir.effect→includes(InterventionEffect::restrictedProcessing)))
```

Listing A.34: *OCL invariant VTC1 for the class CollectionInformationRequirement*

```
context CollectionInformationRequirement
inv VTC1: self.personalData→forAll(pd|
            self.purpose→forAll(p|
              not CollectionInformationRequirement.allInstances()→exists(cir|
                cir <> self and cir.dataSubject = self.dataSubject and
                cir.personalData→includes(pd) and
                cir.purpose→includes(p))))
```

Listing A.35: *OCL specification of the validation condition VTC2 on the class CollectionInformationRequirement*

```
context CollectionInformationRequirement
inv VTC2:
self.dataSubject.finalDfg.collectionEdges.data→forAll(d|
  self.dataSubject.privacyrequirements→select(pr|
    pr.oclIsTypeOf(CollectionInformationRequirement))→exists(tc|
      tc.dataSubject = self and
      tc.personalData→includes(d)))
```

Listing A.36: *OCL specification of the validation condition VTC3 on the class CollectionInformationRequirement*

```
context CollectionInformationRequirement
inv VTC3:
(self.identifiableData and
    self.grounds→excludes(ProcessingGrounds::publicInterest)) implies
self.controlOptions→exists(ir|
  ir.type→includes(DataSubjectIntervention::requestDataCopy))
```

Listing A.37: *OCL invariant VTC1 for the class* FlowInformationRequirement

```
context FlowInformationRequirement
inv VTF1: self.personalData→forAll(pd|
           self.purpose→forAll(p|
             not FlowInformationReqirement.allInstances()→exists(cir|
               cir <> self and cir.dataSubject = self.dataSubject and
               cir.personalData→includes(pd) and
               cir.purpose→includes(p) and
               cir.target = self.target)))
```

Listing A.38: *OCL specification of the validation condition VTF2 on the class* FlowInformationRequirement

```
context FlowInformationRequirement
inv VTF2:
self.personalData.pdavailableats→exists(pd|pd.domain = self.target)
```

Listing A.39: *OCL specification of the validation condition VTF3 on the class* FlowInformationRequirement

```
context FlowInformationRequirement
inv VTF3:
self.dataSubject.finalDfg.flowEdges→forAll(fe|fe.data→forAll(d|
  d.pdavailableats→select(pd|pd.domain = fe.target).availability→forAll(ad|
    self.dataSubject.privacyrequirements→select(pr|
      pr.oclIsTypeOf(FlowInformationRequirement))→exists(tf|
        tf.dataSubject = self and
        tf.personalData→includes(d) and
        tf.target = fe.target and
        tf.availability = ad))))
```

Listing A.40: *OCL invariant VTS1 for the class* StorageInformationRequirement

```
context StorageInformationRequirement
inv VTS1: self.personalData→forAll(pd|
           self.purpose→forAll(p|
             not StorageInformationReqirement.allInstances()→exists(sir|
               sir <> self and sir.dataSubject = self.dataSubject and
               sir.personalData→includes(pd) and
               sir.purpose→includes(p))))
```

Listing A.41: *OCL specification of the validation condition VTS2 on the class* StorageInformationRequirement

```
context StorageInformationRequirement
inv VTS2:
self.dataSubject.relatedtos.personalData.pdavailableats→select(pd|
  pd.domain.domain.designed).personalData→forAll(d|
    self.dataSubject.privacyrequirements→select(pr|
      pr.oclIsTypeOf(StorageInformationRequirement))→exists(ts|
        ts.dataSubject = self and
        ts.personalData→includes(d)))
```

Listing A.42: *OCL invariant VTE1 for the class* ExceptionalInformationRequirement

```
context ExceptionalInformationRequirement
inv VTE1: self.personalData→forAll(pd|
           self.authorities→forAll(a|
             not ExceptionalInformationReqirement.allInstances()→exists(eir|
               eir <> self and eir.dataSubject = self.dataSubject and
```

```
                eir.personalData→includes(pd) and
                eir.case = self.case and eir.authorities→includes(a))))
```

Listing A.43: *OCL specification of the validation condition VTE2 on the class* Person

```
context Person
inv VTE2:
self.processedPersonalData→forAll(p|
  self.privacyrequirements→exists(pr|
    pr.oclIsTypeOf(ExceptionalInformationRequirement) and
    pr.personalData→includes(p) and
    pr.case = ExceptionalCase::dataBreach))
```

Listing A.44: *OCL specification of the validation condition VTE3-VTE6*

```
context Person
inv VTE3:
self.processedPersonalData→forAll(p|
  self.privacyrequirements→exists(pr|
    pr.oclIsTypeOf(ExceptionalInformationRequirement) and
    pr.personalData→includes(p) and
    pr.case = ExceptionalCase::authorityRequest))
inv VTE5:
self.processedPersonalData→forAll(p|
  self.privacyrequirements→exists(pr|
    pr.oclIsTypeOf(ExceptionalInformationRequirement) and
    pr.personalData→includes(p) and
    pr.case = ExceptionalCase::nonCompliance))

context ExceptionalInformationRequirement
inv VTE4:
self.case = ExceptionalCase::authorityRequest implies
self.authorityinterventionrequirements→exists(air|
  air.type→includes(AuthorityIntervention::obtainAccess))
inv VTE6:
self.case = ExceptionalCase::nonCompliance implies
self.authorityinterventionrequirements.type→includesAll(Set{
  AuthorityIntervention::suspendDataFlows,
  AuthorityIntervention::orderBanOfProcessing,
  AuthorityIntervention::orderErasure,
  AuthorityIntervention::orderRectification})
```

Listing A.45: *OCL invariant VID1 for the class* DataSubjectInterventionRequirement

```
context DataSubjectInterventionRequirement
inv VID1:
(self.type = DataSubjectIntervention::doNotConsent implies
  self.effect = InterventionEffect::noProcessing) and
(self.type = DataSubjectIntervention::withDrawConsent implies
  (self.effect = InterventionEffect::noProcessing or
    self.effect = InterventionEffect::restrictedProcessing or
    self.effect = InterventionEffect::erasure)) and
(self.type = DataSubjectIntervention::review implies
  self.effect = InterventionEffect::access) and
(self.type = DataSubjectIntervention::challengeAccuracy implies
  (self.effect = InterventionEffect::correction or
    self.effect = InterventionEffect::amendment or
    self.effect = InterventionEffect::erasure)) and
(self.type = DataSubjectIntervention::challengeCompleteness implies
  (self.effect = InterventionEffect::amendment or
    self.effect = InterventionEffect::erasure)) and
```

```
(self.type = DataSubjectIntervention::object implies
  (self.effect = InterventionEffect::noProcessing or
    self.effect = InterventionEffect::restrictedProcessing or
    self.effect = InterventionEffect::erasure)) and
(self.type = DataSubjectIntervention::requestDataCopy implies
  self.effect = InterventionEffect::dataCopy)
```

Listing A.46: *OCL invariant VID2 for the class DataSubjectInterventionRequirement*

```
context DataSubjectInterventionRequirement
inv VID2: (self.type = DataSubjectIntervention::review or
            self.type = DataSubjectIntervention::challengeAccuracy or
            self.type = DataSubjectIntervention::challengeCompleteness) implies
              self.controlOptions→forAll(pir |
                pir.oclIsTypeOf(FlowInformationRequirement) or
                  pir.oclIsTypeOf(StorageInformationRequirement))
```

Listing A.47: *OCL specification of the validation condition VID3-VID5 on the class DataSubjectInterventionRequirement*

```
context DataSubjectInterventionRequirement
inv VID3:
self.processinginformationrequirement.oclIsTypeOf(
  CollectionInformationRequirement) implies
self.time→exists(t|t.min(ActionTime::beforeCollection) = t)
inv VID4:
self.processinginformationrequirement.oclIsTypeOf(
  FlowInformationRequirement) implies
self.time→exists(t|t.min(ActionTime::beforeTransmission) = t)
inv VID5:
self.processinginformationrequirement.oclIsTypeOf(
  StorageInformationRequirement) implies
self.time→exists(t|t.min(ActionTime::atRecording) = t)
```

Listing A.48: *OCL specification of the validation condition VID6 and VID7 on the class DataSubjectInterventionRequirement*

```
context DataSubjectInterventionRequirement
inv VID6:
self.type→includes(DataSubjectIntervention::review) implies
(self.processinginformationrequirement.oclIsTypeOf(
  CollectionInformationRequirement) or
 self.processinginformationrequirement.oclIsTypeOf(
   StorageInformationRequirement))
inv VID7:
self.type→includes(DataSubjectIntervention::review) implies
self.time→includes(ActionTime::anyTime)
```

Listing A.49: *OCL specification of the validation condition VID8 and VID9 on the class DataSubjectInterventionRequirement*

```
context DataSubjectInterventionRequirement
inv VID8:
(self.type→includes(DataSubjectIntervention::challengeAccuracy) or
  self.type→includes(DataSubjectIntervention::challengeCompleteness)) implies
(self.processinginformationrequirement.oclIsTypeOf(
  FlowInformationRequirement) or
self.processinginformationrequirement.oclIsTypeOf(
  StorageInformationRequirement))
inv VID9:
(self.type→includes(DataSubjectIntervention::challengeAccuracy) or
  self.type→includes(DataSubjectIntervention::challengeCompleteness)) implies
self.time→includes(ActionTime::anyTime)
```

Listing A.50: *OCL specification of the validation condition VID10 on the class DataSubjectInterventionRequirement*

```
context DataSubjectInterventionRequirement
inv VID10:
(self.type→includes(DataSubjectIntervention::withdrawConsent) or
  self.type→includes(DataSubjectIntervention::object) or
  self.type→includes(DataSubjectIntervention::requestDataCopy)) implies
self.time→includes(ActionTime::anyTime)
```

Listing A.51: *OCL specification of the validation condition VID11 on the class DataSubjectInterventionRequirement*

```
context DataSubjectInterventionRequirement
inv VID11:
(self.type→includes(DataSubjectIntervention::withdrawConsent) and
  self.processinginformationrequirement.grounds→intersection(Set{
    ProcessingGrounds::contract, ProcessingGrounds::regulation,
    ProcessingGrounds::vitalInterest, ProcessingGrounds::publicInterest,
    ProcessingGrounds::controllerInterest,
        ProcessingGrounds::legalClaims}).isEmpty()) implies
self.effect→includes(InterventionEffect::erasure)
```

Listing A.52: *OCL invariant VIA1 for the class AuthorityInterventionRequirement*

```
context AuthorityInterventionRequirement
inv VIA1:
(self.type = AuthorityIntervention::suspendDataFlows implies
  self.effect = InterventionEffect::suspendedDataFlows) and
(self.type = AuthorityIntervention::orderBanOfProcessing implies
  (self.effect = InterventionEffect::noProcessing or
    self.effect = InterventionEffect::restrictedProcessing)) and
(self.type = AuthorityIntervention::orderErasure implies
  self.effect = InterventionEffect::erasure) and
(self.type = AuthorityIntervention::orderRectification implies
  (self.effect = InterventionEffect::correction or
    self.effect = InterventionEffect::amendment)) and
(self.type = AuthorityIntervention::obtainAccess implies
  self.effect = InterventionEffect::access)
```

Appendix B

Privacy Impact Assessment Report for the Course Evaluation System

This chapter presents the privacy impact assessment (PIA) report for the course evaluation system that I considered as second case study to illustrate the application of the ProPAn method. The following PIA report is based on the results presented in Chapter 19 and created following the guidelines given in Chapter 18. The PIA report is structured as suggested in ISO 29134 (ISO/IEC, 2017a).

B.1. Introduction

At the University Duisburg-Essen a course evaluation system shall be developed that allows teachers to receive feedback for the courses they have given. For this, they can create their own evaluation forms for each course they give. At the end of the course, the students that regularly participated in the course are allowed to evaluate it once. Hence, the students have to be able to register their participation in the course. Such a registration of participation shall only be possible for those students that actually participate in the course and that are officially enrolled in the course.

The administration of the university already runs a study management system that manages the students of the university, their contact information, the studies and courses of the university, the studies and courses the students are enrolled in, and the students' trials and grades for the courses they have taken. Additionally, the study management system manages the teachers of the universities, their university contact information, and the courses they give. The students and teachers shall be able to authenticate themselves in the course evaluation system using their credentials for the study management system. Using the study management system, the course evaluation system shall also be able to determine whether a student is enrolled in a course, and whether a teacher gives a specific course.

B.2. Scope of PIA

B.2.1. Process Under Evaluation

B.2.1.1. System Requirement Information

The course evaluation system has the following functional requirements.

Create Forms Teachers shall be able to create evaluation forms for their courses.

Participate Students enrolled in a course shall be able to register their participation in a course.

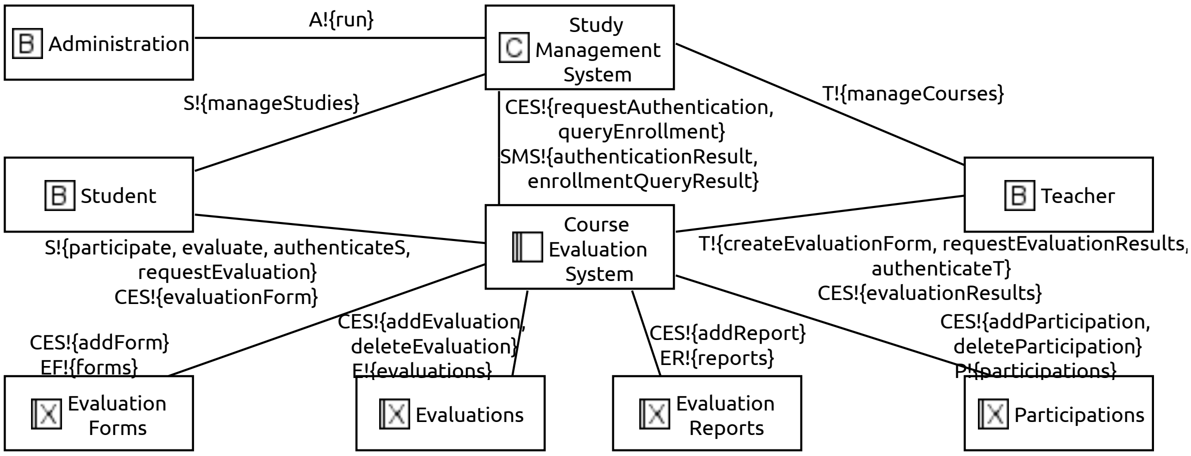


Figure B.1.: *Context diagram for the course evaluation system*

Request Evaluation During the evaluation period, the students that regularly participated in a course shall be able to request the evaluation form for the course, unless they have not already evaluated the course. If the request is successful, all participations for the course shall be deleted.

Evaluate A student can fill out a received evaluation form and submit it to evaluate a course.

Create Report After the evaluation period, the course evaluation system shall create an evaluation report for each course that does not allow to link evaluations to single students. After the creation of an evaluation report, all evaluations for the course shall be deleted.

Show Results Teachers shall be able to access the evaluation reports of their courses.

Authenticate Students and teachers shall authenticate themselves to the course evaluation system using the credentials they also use for the study management system.

The context diagram of the course evaluation system is shown in Figure 19.1. The machine to be built is the Course Evaluation System and it manages the Evaluation Forms that Teachers can create for their courses, the Evaluations performed by Students, the final Evaluation Reports of the courses, and the registered Participations of the students in the courses. Teachers can create evaluation forms and request evaluation results from the Course Evaluation System. A Student can register his or her participation in a course and evaluate courses using the Course Evaluation System. To perform the above mentioned actions, Students and Teachers have to be authenticated. This is done using the existing Study Management System that is run by the Administration of the university. The Students and Teachers use the Study Management System to manage their studies and the courses they give, respectively. Furthermore, the Course Evaluation System can request from the Study Management System whether a Student is officially enrolled in the course he or she wants to participate in to later evaluate it. The Study Management System can also be asked whether a teacher gives a specific course for which he or she wants to create an evaluation form, or wants to access the evaluation results.

Students and teachers shall be able to access the course evaluation system over the internet with a web browser. That is, the interfaces between students and the course evaluation system, and between teachers and the course evaluation system shall be realized by a website.

Table B.1.: *Personal data stored in the lexical domain Participations*

Personal data	Purpose of storage	Retention	Linkable to
participation date and course	To check whether the students sufficiently often participated in a course to evaluate it.	Until the evaluation form is successfully requested.	student ID
student ID	To link a participation date and course to the respective student.	see above	participation date and course

Table B.2.: *Personal data stored in the lexical domain Evaluations*

Personal data	Purpose of storage	Retention	Linkable to
opinion on course and teacher	Represents the feedback that shall be provided to teachers in an aggregated form.	Until the evaluation report for the course was created.	student ID
student ID	To link the other available personal data to the respective student.	see above	the other available personal data of the respective student
studies	To provide the teachers an overview of the studies of the evaluating students.	see above	student ID

Table B.3.: *Personal data stored in the lexical domain EvaluationReports*

Personal data	Purpose of storage	Retention	Linkable to
aggregated evaluations	The feedback that shall be provided to teachers.	Until it is necessary to delete the data.	

B.2.1.2. System Design Information

The personal data of students that shall be stored at the different lexical domains of the course evaluation system are listed in Tables B.1-B.3. The properties of these personal data of students are listed in Table B.4.

A data flow graph illustrating the flow of the students' personal data through the course evaluation system is shown in Figure B.2.

B.2.1.3. Operational Plans and Procedures Information

To protect the privacy of students, the following privacy measures are considered.

Secure Authentication Students and teachers shall authenticate themselves to the course evaluation system using the credentials they also use for the study management system. The authentication mechanism used by the study management is secure and robust.

Security Awareness Students shall be informed about the risks concerning the use of the course evaluation system and how they can protect themselves against these risks.

Table B.4.: *Properties of the processed personal data of students*

Personal data	Sensitive?	How collected?	Linkability to student
aggregated evaluations	no	Derived from studies and opinion on course and teacher of the students who evaluated the respective course	Linkable to a group of 6-50 students
opinion on course and teacher	yes	Directly collected from students	Linkable to a group of 6-50 students
participation date and course	no	Directly collected from students	Linkable to a group of 6-50 students
student ID	no	Reused from the study management system	Linkable to a single student
studies	no	Reused from the study management system	Linkable to a group of 6-50 students

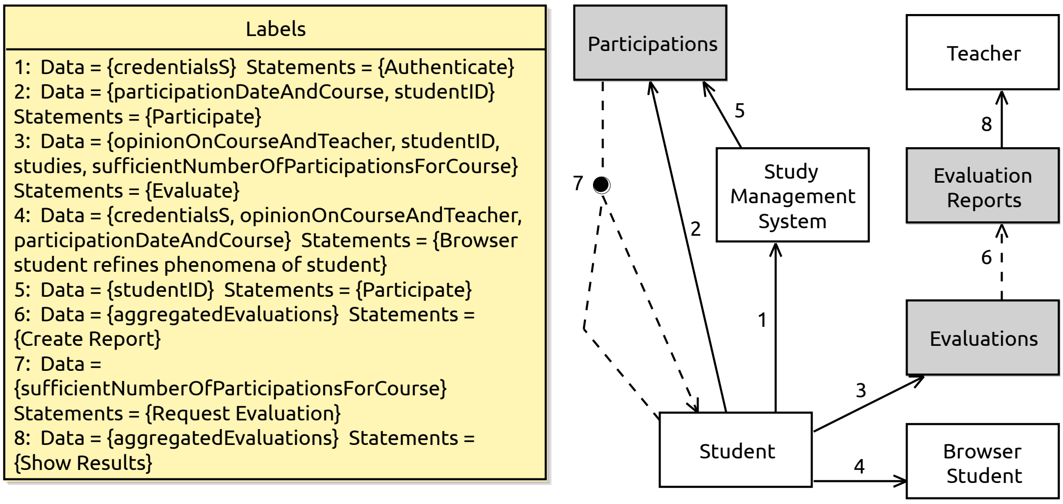

Labels
1: Data = {credentialsS} Statements = {Authenticate}
2: Data = {participationDateAndCourse, studentID} Statements = {Participate}
3: Data = {opinionOnCourseAndTeacher, studentID, studies, sufficientNumberOfParticipationsForCourse} Statements = {Evaluate}
4: Data = {credentialsS, opinionOnCourseAndTeacher, participationDateAndCourse} Statements = {Browser student refines phenomena of student}
5: Data = {studentID} Statements = {Participate}
6: Data = {aggregatedEvaluations} Statements = {Create Report}
7: Data = {sufficientNumberOfParticipationsForCourse} Statements = {Request Evaluation}
8: Data = {aggregatedEvaluations} Statements = {Show Results}
Participations
Teacher
5
8
7
Study Management System
Evaluation Reports
2
6
1
Evaluations
3
Student
4
Browser Student

Figure B.2.: *Data flow graph for the data subject student*

Show Policy The privacy policy of the course evaluation system shall be publicly accessible without the need for a prior authentication. The privacy policy shall inform students about which data of them are collected, for which purpose, how long they are retained, and to whom these data are provided. Additionally, students shall be informed about their rights to intervene in the processing of their personal data.

Informs about Exceptional Cases The administration of the university (in the role of the data controller) shall inform students and the responsible data protection authority LDI.NRW about exceptional cases that occur during the processing of the students' personal data.

Obtain Consent Before data are collected from a student, he or she has to consent to the collection and processing of the data. Together with the consent form, the relevant part of the privacy policy shall be provided to the student. More precisely, the student has to consent to the processing of his or her personal data prior the collection of the personal data due to the functional requirements Participate, Evaluate, and Authenticate.

Table B.5.: *Likelihood scale for the course evaluation system*

Likelihood	Definition
Rare	Less than once per ten years
Unlikely	Less than once per two years
Possible	Less than twice per year
Likely	Two to five times per year
Certain	Five times or more per year

Table B.6.: *Consequence scales for security- and unlinkability-related privacy requirements*

Consequence	Security of participations	Security of evaluations	Security of evaluation results
Insignificant	1-2 participations of a student for the same course are affected	0 evaluations are affected	0 evaluation results are affected
Minor	3-4 participations of a student for the same course are affected	1-2 evaluations are affected	-
Moderate	5-6 participations of a student for the same course are affected	3-4 evaluations are affected	1-2 evaluation results are affected
Major	7-10 participations of a student for the same course are affected	5-6 evaluations are affected	3-4 evaluation results are affected
Catastrophic	>10 participations of a student for the same course are affected	>6 evaluations are affected	>4 evaluation results are affected

Review Students shall be able to review the personal data stored by the course evaluation system. Hence, students can request the list of their personal data stored by the course evaluation system.

Withdraw Consent Students shall be able to withdraw the consent to process all or specific personal data stored of them. These personal data shall consequently be deleted.

Challenge Accuracy and Completeness Students shall be able to challenge the accuracy and completeness of their created evaluations. This is realized by giving students the opportunity to edit their created evaluations until the evaluation period ended. To ensure the validity of the students' participations, the students are not allowed to edit these.

Handle Authority Interventions The administration shall be able to handle the intervention requests of the supervisory authority LDI.NRW.

B.2.2. Risk Criteria

The risk criteria for the course evaluation system are given by the likelihood scale presented in Table B.5, the consequence scales presented in Tables B.6 and B.7, and the risk matrix presented in Table B.8. The cells with white background in the risk matrix represent acceptable risks, those with light gray background tolerable risks, and those with dark gray background unacceptable risks.

Table B.7.: *Consequence scales for the transparency- and intervenability-related privacy requirements*

Consequence	Transparency	Intervenability
Insignificant	Information is provided in a sufficient manner, but not recognized	Data subject can intervene and is sufficiently informed about the options, but does not recognize this information
Minor	Information is provided, but in an insufficient manner	Data subject can intervene, but is insufficiently informed about the options
Moderate	Information is available only with manual effort	Data subject can intervene, but is not informed about the options
Major	Information is only available on request	Data subject has only partial intervention options
Catastrophic	Information is not accessable and not provided on request	Data subject has no possibility to intervene

Table B.8.: *Risk matrix for the course evaluation system*

	Insignificant	Minor	Moderate	Major	Catastrophic
Rare	□	□	□	▒	▒
Unlikely	□	□	▒	▒	■
Possible	□	▒	▒	■	■
Likely	▒	▒	■	■	■
Certain	▒	■	■	■	■

B.3. Privacy Requirements

The following privacy requirements have been identified for the course evaluation system.

Integrity Random faults of the system and teachers, students, or attackers shall not be able to negatively influence the consistency and correctness of the personal data aggregated evaluations, opinion on course and teacher, participation date and course, student ID, and studies of students.

Availability Random faults of the system and teachers, students, or attackers shall not be able to hinder the corresponding student to access his or her personal data opinion on course and teacher, participation date and course, student ID, and studies.

Data Confidentiality Attacker The personal data aggregated evaluations, opinion on course and teacher, participation date and course, student ID, and studies of students shall be kept confidential from attackers.

Data Confidentiality Teacher The personal data opinion on course and teacher of students shall be kept confidential from teachers. The personal data aggregated evaluations shall only be accessible to those teachers who gave the corresponding course.

Data Confidentiality Student The personal data opinion on course and teacher, participation date and course, student ID, and studies of students shall only be available to those students to whom these data belong to.

Anonymity Teacher Teachers shall only be able to link the personal data aggregated evaluations of students to those students who regularly participated in the course (a group of 6-50 students).

Collection Information Students shall be informed that their personal data participation date and course, student ID, opinion on course and teacher, and studies are mandatorily collected by the system-to-be that is run by the administration of the University Duisburg-Essen. The applied collection methods to obtain the personal data from the data subject are direct, and reused. The data subject's possibilities to control the collection of his or her data are Intervention Consent and Intervention Withdraw Consent. The personal data are collected during Participate and Evaluate for the purpose of Request Evaluation, Evaluate, and Create Report. The processing grounds on consent. The controller has selected the protection mechanisms Secure Authentication to protect the personal data. The information has to be presented in German and English before the data are collected.

Storage Information Students shall be informed that their personal data participation date and course, student ID, opinion on course and teacher, studies, and aggregated evaluations are mandatorily stored by the system-to-be that is run by the administration of the University Duisburg-Essen. The personal data participation date and course, student ID, opinion on course and teacher, and studies are retained by the system-to-be only for the action to be performed for creating evaluation reports. The personal data aggregated evaluations are retained by the system-to-be until it is necessary to delete them. The data subject's possibilities to control the storage of his or her data are Intervention Consent, Intervention Withdraw Consent, Intervention Review, and Intervention Challenge Accuracy and Completeness. The personal data are stored during Participate, Evaluate, and Create Report for the purpose of Request Evaluation, Evaluate, Create Report, and Show Results. The processing grounds on consent. The controller has selected the protection mechanisms Secure Authentication to protect the personal data. The information has to be presented in German and English before the data are collected.

Flow Information Teacher Students shall be informed that their personal data aggregated evaluations mandatorily flow to authorized teachers due to the system-to-be that is run by the administration of the University Duisburg-Essen. The authorized teachers are located in Germany and they are employees of the University Duisburg-Essen. The data subject's possibilities to control the flow of his or her data are Intervention Consent and Intervention Withdraw Consent. The personal data flow during Show Results. The processing grounds on consent. The controller has selected the protection mechanisms Secure Authentication to protect the personal data. The information has to be presented in German and English before the data are collected.

Flow Information Study Management System Students shall be informed that their personal data credentials mandatorily flow to the study management system due to the system-to-be that is run by the administration of the University Duisburg-Essen. The study management system is located in Germany and run by the administration of the University Duisburg-Essen. The data subject's possibilities to control the flow of his or her data are Intervention Consent and Intervention Withdraw Consent. The personal data flow during Authenticate for the purpose of Participate, Request Evaluation Form, Evaluate, Review, Withdraw Consent,and Challenge Accuracy and Completeness. The processing grounds on consent. The controller has selected the protection mechanisms Secure Authentication to protect the personal data. The information has to be presented in German and English before the data are collected.

Exceptional Information Students and the LDI.NRW shall be informed in the case of a data breach, system change, non compliance, or an authority request regarding or affecting the personal data aggregated evaluations, opinion on course and teacher, participation date and course, student ID, and studies of students. The LDI.NRW may exercise in this situation their rights described by Intervention Authority.

Intervention Consent Before the collection of personal data, students shall be able to not consent to the processing described in Collection Information, Storage Information, Flow Information Teacher, and Flow Information Study Management System. The intervention shall result in no processing of the respective data. This may result in the consequence that the student is not able to evaluate the course.

Intervention Withdraw Consent Any time, students shall be able to withdraw their consent to the processing described in Collection Information, and Storage Information. The intervention shall result in the deletion of the respective personal data. This may result in the consequence that the student is not able to evaluate the course.

Intervention Review Any time, students shall be able to review the personal data whose processing is described in Storage Information. The intervention shall result in access to the personal data.

Intervention Challenge Accuracy and Completeness Any time, students shall be able to challenge the accuracy and completeness of the personal data whose processing is described in Storage Information. The intervention shall result in a correction and amendment of the personal data.

Intervention Authority The LDI.NRW shall be able to suspend data flows, order a ban of processing, erasure, or rectification, or request to obtain access in the cases described in Exceptional Information. The intervention may result in suspended data flows, no or restricted processing of the personal data, erasure, amendment, correction, and access to the personal data.

Table B.9.: *Risks identified for the course evaluation system*

Risk	Description	Likelihood	Harms	Consequence
R1	Students cannot evaluate a course because they are not able to authenticate themselves leading to fewer evaluations recorded by the system.	Rare	Integrity and Availability	Moderate
R2	Students cannot register a participation for a course because they are not able to authenticate themselves leading to fewer participations recorded by the system.	Rare	Integrity and Availability	Moderate
R3	No or too few evaluations are stored to create an evaluation report leading to its unavailability.	Possible	Integrity and Availability	Minor
R4	A student maliciously gets access to the evaluation system by stealing a student's or teacher's credentials leading to the situation that he or she gets access to other students' personal data.	Rare	Data Confidentiality Student	Major
R5	A student enters self-identifying information into an evaluation form leading to the situation that the teacher can link a part of an evaluation report to the respective student, allowing the teacher also to get to know the student's personal data opinion on course and teacher.	Rare	Anonymity Teacher and Data Confidentiality Teacher	Moderate
R6	An attacker maliciously gets access to the evaluation system by stealing a student's or teacher's credentials leading to the situation that he or she gets access to a students' personal data.	Unlikely	Data Confidentiality Attacker	Major
R7	An attacker eavesdrops a student's or teacher's web browser or network leading to a disclosure of personal data of students.	Unlikely	Data Confidentiality Attacker	Major
R8	A data breach occurs about which students and the LDI.NRW have to be informed	Unlikely	Exceptional Information	Insignificant

B.4. Risk Assessment

The risks that have been identified for the course evaluation system are listed in Table B.9. The severity of these risks is given by the risk matrix shown in Table B.10.

Table B.10.: *Severity of the identified risks for the course evaluation system*

	Insignificant	Minor	Moderate	Major	Catastrophic
Rare			R1, R2, R5	R4	
Unlikely	R8			R6, R7	
Possible		R3			
Likely					
Certain					

Table B.11.: *Risk treatment plan for the course evaluation system*

Risk	Mitigating Privacy Measures
R1	-
R2	-
R3	-
R4	The authentication mechanism shall be secure and hard to by-pass. Furthermore, students shall be aware of how they can protect themselves against cyber security attacks.
R5	Students shall be told not to enter self-identifying information into an evaluation form.
R6	The authentication mechanism shall be secure and hard to by-pass. Furthermore, students and teachers shall be aware of how they can protect themselves against cyber security attacks.
R7	Students and teachers shall be aware of how they can protect themselves against cyber security attacks.
R8	The administration shall monitor the course evaluation and notify students and the LDI.NRW when exceptional cases occur.

B.5. Risk Treatment Plan

The privacy measures to treat the risks are given in Table B.11. Note that the previously presented likelihoods and consequences were already estimated considering the measures listed in Table B.11.

B.6. Conclusion and Decision

The impact of the course evaluation system on the privacy of students is considered to be acceptable. The risk evaluation has shown that no unacceptable risks have been identified. The data subject right to be informed about the processing of one's personal data and the rights to intervene in the processing of one's personal data are realized by respective functionalities. Furthermore, the course evaluation system only process personal data of a student if he or she consents to the processing. The processed personal data are only retained for the time these necessarily need to be stored, and after this time, the personal data are deleted.

Consequently, the performed PIA shows that the course evaluation system does not imply severe privacy impacts on students.

Bibliography

Alessandro Acquisti, Idris Adjerid, and Laura Brandimarte. Gone in 15 seconds: The limits of privacy transparency and control. *IEEE Security and Privacy*, 11(4):72–74, jul 2013. ISSN 1540-7993. doi: 10.1109/msp.2013.86. (Cited on page 94.)

Philip E. Agre. The architecture of identity: Embedding privacy in market institutions. *Information, Communication and Society*, pages 1–25, 1999. (Cited on page 25.)

Amir Shayan Ahmadian and Jan Jürjens. Supporting model-based privacy analysis by exploiting privacy level agreements. In *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 360–365, Dec 2016. doi: 10.1109/cloudcom.2016.0063. (Cited on pages 48, 49, 61, 169, 202, 264 and 337.)

Amir Shayan Ahmadian, Daniel Strüber, Volker Riediger, and Jan Jürjens. *Model-Based Privacy Analysis in Industrial Ecosystems*, pages 215–231. Springer International Publishing, Cham, 2017. ISBN 978-3-319-61482-3. doi: 10.1007/978-3-319-61482-3_13. URL `https://doi.org/10.1007/978-3-319-61482-3_13`. (Cited on page 49.)

Susana Alcalde Bagüés, Jelena Mitic, Andreas Zeidler, Marta Tejada, IgnacioR. Matias, and Carlos Fernandez Valdivielso. Obligations: Building a bridge between personal and enterprise privacy in pervasive computing. In *Trust, Privacy and Security in Digital Business*, LNCS 5185, pages 173–184. Springer, 2008. ISBN 978-3-540-85734-1. doi: 10.1007/978-3-540-85735-8_17. (Cited on page 83.)

Azadeh Alebrahim, Isabelle Côté, Maritta Heisel, Christine Choppy, and Denis Hatebur. Designing architectures from problem descriptions by interactive model transformation. In *Proceedings 27th Symposium on Applied Computing*, pages 1256–1258. ACM, 2012a. URL `http://dl.acm.org/`. (Cited on page 279.)

Azadeh Alebrahim, Thein Than Tun, Yijun Yu, Maritta Heisel, and Bashar Nuseibeh. An aspect-oriented approach to relating security requirements and access control. In *Proceedings of the CAiSE Forum*, volume 855 of *CEUR Workshop Proceedings*, pages 15–22. CEUR-WS.org, 2012b. URL `http://ceur-ws.org/`. (Cited on page 304.)

Azadeh Alebrahim, Maritta Heisel, and Rene Meis. A structured approach for eliciting, modeling, and using quality-related domain knowledge. In *Proceedings of the 14th International Conference on Computational Science and Its Applications (ICCSA)*, LNCS 8583, pages 370–386. Springer, 2014. doi: 10.1007/978-3-319-09156-3_27. URL `http://dx.doi.org/10.1007/978-3-319-09156-3_27`. (Cited on page 422.)

Ian Alexander. Misuse cases: use cases with hostile intent. *IEEE Software*, 20(1):58–66, Jan 2003. ISSN 0740-7459. doi: 10.1109/ms.2003.1159030. (Cited on page 48.)

Ian F. Alexander. A taxonomy of stakeholders: Human roles in system development. *IJTHI*, 1 (1):23–59, 2005. doi: 10.4018/jthi.2005010102. (Cited on page 149.)

Ian F. Alexander and Suzanne Robertson. Understanding project sociology by modeling stakeholders. *IEEE Software*, 21(1):23–27, 2004. URL `http://doi.ieeecomputersociety.org/10.1109/MS.2004.1259199`. (Cited on pages 135 and 149.)

Dalal Alrajeh, Jeff Kramer, Alessandra Russo, and Sebastián Uchitel. Learning operational requirements from goal models. In *31st International Conference on Software Engineering, ICSE 2009, May 16-24, 2009, Vancouver, Canada, Proceedings*, pages 265–275. IEEE, 2009. doi: 10.1109/icse.2009.5070527. URL `https://doi.org/10.1109/ICSE.2009.5070527`. (Cited on page 278.)

Thibaud Antignac, Riccardo Scandariato, and Gerardo Schneider. *A Privacy-Aware Conceptual Model for Handling Personal Data*, pages 942–957. Springer International Publishing, Cham, 2016. ISBN 978-3-319-47166-2. doi: 10.1007/978-3-319-47166-2_65. URL `https://doi.org/10.1007/978-3-319-47166-2_65`. (Cited on pages 37, 38, 169, 202, 203 and 242.)

Annie I. Antón. Goal-based requirements analysis. In *Proceedings of the Second International Conference on Requirements Engineering*, pages 136–144, Apr 1996. doi: 10.1109/icre.1996.491438. (Cited on pages 13 and 45.)

Annie I. Antón and Julia B. Earp. A requirements taxonomy for reducing web site privacy vulnerabilities. *Requirements Engineering*, 9(3):169–185, 2004. (Cited on pages 51, 83, 94 and 242.)

Annie I. Antón and Qingfeng He. A framework for modeling privacy requirements in role engineering. In *Pre-Proceedings of the 9th International Workshop on Requirements Engineering - Foundation for Software Quality*, pages 115–124. REFSQ, 2003. URL `http://crinfo.univ-paris1.fr/REFSQ/03/papers/REFSQ03-PreProceedings.pdf`. (Cited on pages 45 and 337.)

Annie I. Antón, Julia B. Earp, and A. Reese. Analyzing website privacy requirements using a privacy goal taxonomy. In *IEEE Int. Conf. on Requirements Engineering*, pages 23–31, 2002. URL `http://doi.ieeecomputersociety.org/10.1109/ICRE.2002.1048502`. (Cited on pages 83 and 94.)

Annie I. Antón, Julia B. Earp, M. W. Vail, N. Jain, C. M. Gheen, and J. M. Frink. HIPAA's effect on web site privacy policies. *Security Privacy, IEEE*, 5(1):45–52, Jan 2007. ISSN 1540-7993. doi: 10.1109/msp.2007.7. (Cited on page 83.)

Nikolaos Argyropoulos, Christos Kalloniatis, Haralambos Mouratidis, and Andrew Fish. Incorporating privacy patterns into semi-automatic business process derivation. In *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, pages 1–12, June 2016. doi: 10.1109/rcis.2016.7549305. (Cited on pages 45, 46 and 337.)

Alessandro Armando, Silvio Ranise, Riccardo Traverso, and Konrad Wrona. Smt-based enforcement and analysis of nato content-based protection and release policies. In *Proceedings of the 2016 ACM International Workshop on Attribute Based Access Control*, ABAC '16, pages 35–46, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4079-3. doi: 10.1145/2875491.2875493. (Cited on page 45.)

Charles Arthur. iphone keeps record of everywhere you go. Online: `https://www.theguardian.com/technology/2011/apr/20/iphone-tracking-prompts-privacy-fears`, 2011. accessed on 25. April 2017. (Cited on page 3.)

Article 29 Data Protection Working Party. 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. Technical Report WP 248 rev.01, Article 29 Data Protection Working Party, 2017. URL `http://ec.europa.eu/newsroom/article29/item-detail.cfm?item_id=611236`. (Cited on pages 343, 344, 346 and 348.)

Article 29 Data Protection Working Party. Guidelines on transparency under regulation 2016/679. Technical Report WP 260, Article 29 Data Protection Working Party, 2018. URL `http://ec.europa.eu/newsroom/article29/item-detail.cfm?item_id=622227`. (Cited on page 74.)

Micheal Barbaro and Tom Zeller. A face is exposed for aol searcher no. 4417749. *The New York Times*, August 2006. URL `http://www.nytimes.com/2006/08/09/technology/09aol.html`. accessed on 30 June 2017. (Cited on page 25.)

Ken Barker, Mina Askari, Mishtu Banerjee, Kambiz Ghazinour, Brenan Mackas, Maryam Majedi, Sampson Pun, and Adepele Williams. *A Data Privacy Taxonomy*, pages 42–54. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-642-02843-4. doi: 10.1007/978-3-642-02843-4_7. URL `https://doi.org/10.1007/978-3-642-02843-4_7`. (Cited on pages 40, 49, 56, 120 and 123.)

Adam Barth, Anupam Datta, John C. Mitchell, and Helen Nissenbaum. Privacy and contextual integrity: Framework and applications. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, SP '06, pages 184–198, Washington, DC, USA, 2006. IEEE Computer Society. ISBN 0-7695-2574-1. doi: 10.1109/sp.2006.32. URL `https://doi.org/10.1109/SP.2006.32`. (Cited on page 40.)

Romina Barth. Entwicklung einer modell-basierten bibliothek von privacy-threats. Bachelor thesis, Universität Duisburg-Essen, 2016. (Cited on page 255.)

Cesare Bartolini, Robert Muthuri, and Cristiana Santos. *Using Ontologies to Model Data Protection Requirements in Workflows*, pages 233–248. Springer International Publishing, Cham, 2017. ISBN 978-3-319-50953-2. doi: 10.1007/978-3-319-50953-2_17. URL `https://doi.org/10.1007/978-3-319-50953-2_17`. (Cited on pages 40 and 61.)

Tânia Basso, Regina Moraes, Mario Jino, and Marco Vieira. Requirements, design and evaluation of a privacy reference architecture for web applications and services. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 1425–1432. ACM, 2015. (Cited on page 94.)

Kent Beck. *Extreme Programming Explained: Embrace Change.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000. ISBN 0-201-61641-6. (Cited on pages 13 and 422.)

Kristian Beckers, Isabelle Côté, Stephan Faßbender, Maritta Heisel, and Stefan Hofbauer. A pattern-based method for establishing a cloud-specific information security management system. *Requirements Engineering*, pages 1–53, 2013a. URL `http://www.springerlink.com/`. (Cited on pages 133 and 137.)

Kristian Beckers, Stephan Faßbender, Maritta Heisel, and Federica Paci. Combining goal-oriented and problem-oriented requirements engineering methods. In *Proceedings of the International Cross Domain Conference and Workshop (CD-ARES 2013)*, LNCS 8127, pages 178–194. Springer, 2013b. URL `http://www.springerlink.com/`. (Cited on pages 62 and 278.)

Kristian Beckers, Stephan Faßbender, Stefanos Gritzalis, Maritta Heisel, Christos Kalloniatis, and Rene Meis. Privacy-aware cloud deployment scenario selection. In *Trust, Privacy, and Security in Digital Business*, LNCS 8647, pages 94–105. Springer, 2014a. doi: 10.1007/978-3-319-09770-1_9. URL `http://dx.doi.org/10.1007/978-3-319-09770-1_9`. (Cited on page 133.)

Kristian Beckers, Stephan Faßbender, Maritta Heisel, and Rene Meis. A problem-based approach for computer aided privacy threat identification. In D. Ikonomou and Bart Preneel, editors, *Privacy Technologies and Policy*, volume 8319 of *LNCS*, pages 1–16. Springer, 2014b. doi: 10.1007/978-3-642-54069-1_1. URL http://dx.doi.org/10.1007/978-3-642-54069-1_1. (Cited on page 99.)

Victoria Bellotti and Abigail Sellen. Design for privacy in ubiquitous computing environments. In *Proceedings of the Third Conference on European Conference on Computer-Supported Cooperative Work*, ECSCW'93, pages 77–92, Norwell, MA, USA, 1993. Kluwer Academic Publishers. ISBN 0-7923-2447-1. URL http://dl.acm.org/citation.cfm?id=1241934.1241940. (Cited on pages 46, 47, 51, 123, 242 and 337.)

Christoph Bier. How usage control and provenance tracking get together - a data protection perspective. In *IEEE Security and Privacy Workshops (SPW)*, pages 13–17, May 2013. doi: 10.1109/spw.2013.24. (Cited on pages 82 and 94.)

Conrad Bock, Steve Cook, Pete Rivett, Tom Rutt, Ed Seidewitz, Bran Selic, and Doug Tolbert. *OMG Unified Modeling Language (OMG UML)*, March 2015. http://www.omg.org/spec/UML/2.5/. (Cited on pages 13 and 270.)

Frederik Zuiderveen Borgesius. Informed consent: We can do better to defend privacy. *IEEE Security & Privacy*, 13(2):103–107, 2015. (Cited on page 94.)

Travis Breaux. Privacy requirements in an age of increased sharing. *Software, IEEE*, 31(5): 24–27, Sept 2014. doi: 10.1109/ms.2014.118. (Cited on pages 82 and 94.)

Travis Breaux and David Gordon. What engineers should know about us security and privacy law. *Security Privacy, IEEE*, 11(3):72–76, May 2013. ISSN 1540-7993. doi: 10.1109/msp.2013.74. (Cited on page 83.)

Travis D. Breaux and Ashwini Rao. Formal analysis of privacy requirements specifications for multi-tier applications. In *2013 21st IEEE International Requirements Engineering Conference (RE)*, pages 14–23, July 2013. doi: 10.1109/re.2013.6636701. (Cited on page 41.)

Travis D. Breaux, Hanan Hibshi, and Ashwini Rao. Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements. *Requirements Engineering*, 19(3):281–307, Sep 2014. ISSN 1432-010X. doi: 10.1007/s00766-013-0190-7. URL https://doi.org/10.1007/s00766-013-0190-7. (Cited on page 41.)

Travis D. Breaux, Daniel Smullen, and Hanan Hibshi. Detecting repurposing and over-collection in multi-party privacy requirements specifications. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pages 166–175, Aug 2015. doi: 10.1109/re.2015.7320419. (Cited on pages 41 and 264.)

Sean Brooks, Micheal Garcia, Naomi Lefkovitz, Suzanne Lightman, and Ellen Nadeau. An introduction to privacy engineering and risk management in federal systems. Technical Report NISTIR 8062, National Institute of Standards and Technology (NIST) U.S. Department of Commerce, January 2017. (Cited on pages 27 and 28.)

Jim Buchan, Christian Harsana Ekadharmawan, and Stephen G. MacDonell. Insights into domain knowledge sharing in software development practice in smes. In *16th Asia-Pacific Software Engineering Conference, APSEC 2009, 1-3 December 2009, Batu Ferringhi, Penang, Malaysia*, pages 93–100. IEEE Computer Society, 2009. doi: 10.1109/apsec.2009.47. URL https://doi.org/10.1109/APSEC.2009.47. (Cited on page 422.)

Jan Camenisch, Ioannis Krontiris, Anja Lehmann, Gregory Neven, Christian Paquin, Kai Rannenberg, and Harald Zwingelberg. D2.1 architecture for attribute-based credential technologies – version 1. Technical report, ABC4Trust, 2011. `https://abc4trust.eu/download/ABC4Trust-D2.1-Architecture-V1.2.pdf` (accessed on 21 March 2017). (Cited on pages 310, 312, 316, 317 and 318.)

Xavier Caron, Rachelle Bosua, Sean B. Maynard, and Atif Ahmad. The internet of things (iot) and its impact on individual privacy: An australian perspective. *Computer Law & Security Review*, 32(1):4 – 15, 2016. ISSN 0267-3649. doi: 10.1016/j.clsr.2015.12.001. URL `http://www.sciencedirect.com/science/article/pii/S0267364915001661`. (Cited on page 94.)

Marco Casassa Mont. Dealing with privacy obligations: Important aspects and technical approaches. In *Trust and Privacy in Digital Business*, LNCS 3184, pages 120–131. Springer, 2004. ISBN 978-3-540-22919-3. doi: 10.1007/978-3-540-30079-3_13. (Cited on page 83.)

Ann Cavoukian. Privacy by design – the 7 foundational principles. Online: `https://www.ipc.on.ca/wp-content/uploads/Resources/7foundationalprinciples.pdf` (Accessed 13. April 2017), January 2011. (Cited on pages 4 and 27.)

David Chaum. Achieving electronic privacy. *Scientific American*, 267(2):96–101, 1992. doi: 10.1038/scientificamerican0892-96. (Cited on page 24.)

Christine Choppy and Maritta Heisel. Une approache à base de "patrons" pour la spécification et le développement de systèmes d'information. In *Proceedings Approches Formelles dans l'Assistance au Développement de Logiciels - AFADL'2004*, pages 61–76, 2004. (Cited on page 19.)

Christine Choppy, Denis Hatebur, and Maritta Heisel. Systematic architectural design based on problem patterns. In P. Avgeriou, J. Grundy, J. G. Hall, P. Lago, and I. Mistrik, editors, *Relating Software Requirements and Architectures*, chapter 9, pages 133–159. Springer, 2011. URL `https://link.springer.com/`. (Cited on page 279.)

Lawrence Chung and Julio Cesar Sampaio do Prado Leite. *On Non-Functional Requirements in Software Engineering*, pages 363–379. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-642-02463-4. doi: 10.1007/978-3-642-02463-4_19. (Cited on page 15.)

Siobhán Clarke and Robert J. Walker. Composition patterns: An approach to designing reusable aspects. In *Proceedings of the 23rd International Conference on Software Engineering, ICSE 2001, 12-19 May 2001, Toronto, Ontario, Canada*, pages 5–14. IEEE Computer Society, 2001. doi: 10.1109/icse.2001.919076. (Cited on page 304.)

Dai Clegg and Richard Barker. *Case Method Fast-Track: A Rad Approach.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1994. ISBN 020162432X. (Cited on page 339.)

Cloud Security Alliance. Privacy level agreement [v2]: A compliance tool for providing cloud services in the european union. Technical report, Cloud Security Alliance, 2009. URL `https://cloudsecurityalliance.org/download/privacy-level-agreement-version-2`. (accessed on 20 October 2017). (Cited on page 48.)

Michael Colesky and Sepideh Ghanavati. Privacy shielding by design – a strategies case for near-compliance. In *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*, pages 271–275, Sept 2016. doi: 10.1109/rew.2016.051. (Cited on page 45.)

Michael Colesky, Jaap-Henk Hoepman, and Christiaan Hillen. A critical analysis of privacy design strategies. In *2016 IEEE Security and Privacy Workshops (SPW)*, pages 33–40, May 2016. doi: 10.1109/spw.2016.23. (Cited on pages 44, 45, 61 and 337.)

Isabelle Côté, Denis Hatebur, Maritta Heisel, Holger Schmidt, and Ina Wentzlaff. A systematic account of problem frames. In *Proceedings of the European Conference on Pattern Languages of Programs (EuroPLoP)*, pages 749–767. Universitätsverlag Konstanz, 2008. URL `http://www.uvk.de/`. (Cited on page 19.)

Isabelle Côté, Denis Hatebur, Maritta Heisel, and Holger Schmidt. UML4PF – a tool for problem-oriented requirements analysis. In *Proceedings of the International Conference on Requirements Engineering (RE)*, pages 349–350. IEEE Computer Society, 2011. URL `https://www.ieee.org`. (Cited on page 20.)

Lorrie Faith Cranor. Necessary but not sufficient: Standardized mechanisms for privacy notice and choice. *JTHTL*, 10(2):273–308, 2012. URL `http://www.jthtl.org/content/articles/V10I2/JTHTLv10i2_Cranor.PDF`. (Cited on page 94.)

Alberto Crespo, Nicolás Notario, Carmela Troncoso, Daniel Le Métayer, Inga Kroener, David Wright, José M. del Álamo, and Yod-Samuel Martín. Pripare privacy- and security-by-design methodology handbook. Technical Report V1.00, PRIPARE Project, 2015. URL `http://pripareproject.eu/wp-content/uploads/2013/11/PRIPARE-Methodology-Handbook-Final-Feb-24-2016.pdf`. (Cited on pages 53, 54, 150, 169, 202, 264, 337 and 355.)

Javier Cubo, Jorge Cuellar, Steffen Fries, J. Antonio Martin, Francisco Moyano, Gerardo Fernandez, Aljosa Pasic, Rodrigo Roman, Ruben Torres Dieguez, and Isabel Vinagre. Deliverable d11.2 – selection and documentation of the two major application case studies. Technical report, Network of Excellence on Engineering Secure Future Internet Software Services and Systems (NESSoS), 2011. URL `http://www.nessos-project.eu/media/deliverables/y1/NESSoS-D11.2.pdf`. (Cited on page 65.)

Christian Dänekas, Christian Neureiter, Sebastian Rohjans, Mathias Uslar, and Dominik Engel. *Towards a Model-Driven-Architecture Process for Smart Grid Projects*, pages 47–58. Springer International Publishing, Cham, 2014. ISBN 978-3-319-04313-5. doi: 10.1007/978-3-319-04313-5_5. URL `https://doi.org/10.1007/978-3-319-04313-5_5`. (Cited on page 40.)

Sourya Joyee De and Daniel Le Métayer. *PRIAM: A Privacy Risk Analysis Methodology*, pages 221–229. Springer International Publishing, Cham, 2016. ISBN 978-3-319-47072-6. doi: 10.1007/978-3-319-47072-6_15. (Cited on pages 40, 150, 169, 202, 203 and 264.)

Martin Degeling, Christopher Lentzsch, Alexander Nolte, Thomas Herrmann, and Kai-Uwe Loser. Privacy by socio-technical design: A collaborative approach for privacy friendly system design. In *2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC)*, pages 502–505, Nov 2016. doi: 10.1109/cic.2016.077. (Cited on pages 47, 48, 242 and 337.)

Daniel Deibler, Malte Engeler, Ioannis Krontiris, Anja Lehmann, Vasiliki Liagkou, Apostolos Pyrgelis, Eva Schlehahn, Yannis Stamatiou, Welderufael Tesfay, and Harald Zwingelberg. D7.3 evaluation of the student pilot. Technical report, ABC4Trust, 2014. `https://abc4trust.eu/download/Deliverable%20D7.3.pdf` (accessed on 31 July 2018). (Cited on page 403.)

Mina Deng, Kim Wuyts, Riccardo Scandariato, Bart Preneel, and Wouter Joosen. A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. *Requirements Engineering*, 16(1):3–32, March 2011. doi: 10.1007/s00766-010-0115-7. (Cited on pages 48, 54, 58, 82, 94, 101, 121, 122, 203, 242, 264 and 337.)

Michelle Dennedy, Jonathan Fox, and Thomas Finneran. *The Privacy Engineer's Manifesto*. Apress, 1 edition, 2014. ISBN 978-1-4302-6355-5. doi: 10.1007/978-1-4302-6356-2. (Cited on pages 42 and 337.)

Deutscher Bundestag. Gesetz über Personalausweise und den elektronischen Identitätsnachweis sowie zur Änderung weiterer Vorschriften. *Bundesgesetzblatt*, I(33), 2009. (Cited on pages 317 and 318.)

Vasiliki Diamantopoulou, Nikolaos Argyropoulos, Christos Kalloniatis, and Stefanos Gritzalis. Supporting the design of privacy-aware business processes via privacy process patterns. In *2017 11th International Conference on Research Challenges in Information Science (RCIS)*, pages 187–198, May 2017. doi: 10.1109/rcis.2017.7956536. (Cited on pages 45 and 337.)

Michele Drgon, Gail Magnuson, and John Sabo. *Privacy Management Reference Model and Methodology (PMRM)*. OASIS Committee Specification 02, version 1.0 edition, 2016. URL http://docs.oasis-open.org/pmrm/PMRM/v1.0/cs02/PMRM-v1.0-cs02.html. (Cited on pages 58, 150, 169, 170, 202, 337 and 355.)

Olha Drozd. Privacy pattern catalogue: A tool for integrating privacy principles of ISO/IEC 29100 into the software development process. In *Privacy and Identity Management. Time for a Revolution? - 10th IFIP WG 9.2, 9.5, 9.6/11.7, 11.4, 11.6/SIG 9.2.2 International Summer School 2015, Revised Selected Papers*, volume 476 of *IFIP Advances in Information and Communication Technology*, pages 129–140. Springer, 2016. doi: 10.1007/978-3-319-41763-9_9. URL https://doi.org/10.1007/978-3-319-41763-9_9. (Cited on pages 325 and 331.)

Joshua Eaton and Ben Piven. Timeline of edward snowden's revelations. Online: http://america.aljazeera.com/articles/multimedia/timeline-edward-snowden-revelations.html, 2014. accessed on 25. April 2017. (Cited on page 3.)

European Commission. Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation), January 2012. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52012PC0011. (Cited on pages 74, 77 and 83.)

European Commission. REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), April 2016. http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R0679 (accessed on 21 March 2017). (Cited on pages 6, 25, 26, 34, 40, 45, 49, 58, 85, 86, 88, 94, 123, 131, 154, 155, 207, 221, 224, 232, 246, 307, 316, 318, 331, 343, 366, 412 and 414.)

European Parliament. Directive 95/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, October 1995. http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:31995L0046. (Cited on pages 44, 45 and 48.)

Stephan Faßbender, Maritta Heisel, and Rene Meis. Aspect-oriented requirements engineering with problem frames. In *ICSOFT-PT 2014 - Proc. of the 9th Int. Conf. on Software Paradigm Trends*, pages 145–156. SciTePress, 2014. doi: 10.5220/0005001801450156. URL `http://dx.doi.org/10.5220/0005001801450156`. (Cited on pages 269 and 270.)

Stephan Faßbender, Maritta Heisel, and Rene Meis. A problem-, quality-, and aspect-oriented requirements engineering method. In *Software Technologies - 9th International Joint Conference, ICSOFT 2014, Vienna, Austria, August 29-31, 2014, Revised Selected Papers*, volume 555 of *Communications in Computer and Information Science*, pages 291–310. Springer, 2015. doi: 10.1007/978-3-319-25579-8_17. URL `http://dx.doi.org/10.1007/978-3-319-25579-8_17`. (Cited on pages 269 and 270.)

Joan Feigenbaum, Michael J. Freedman, Tomas Sander, and Adam Shostack. Privacy engineering for digital rights management systems. In *Security and Privacy in Digital Rights Management*, LNCS 2320, pages 76–105. Springer, 2002. ISBN 978-3-540-43677-5. doi: 10.1007/3-540-47870-1_6. (Cited on pages 82 and 94.)

Denis Feth, Andreas Maier, and Svenja Polst. *A User-Centered Model for Usable Security and Privacy*, pages 74–89. Springer International Publishing, Cham, 2017. ISBN 978-3-319-58460-7. doi: 10.1007/978-3-319-58460-7_6. URL `https://doi.org/10.1007/978-3-319-58460-7_6`. (Cited on pages 55, 121, 122 and 337.)

Hervais Simo Fhom and Kpatcha M. Bayarou. Towards a holistic privacy engineering approach for smart grid systems. In *IEEE 10th Int. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 234–241, Nov 2011. doi: 10.1109/trustcom.2011.32. (Cited on pages 82 and 94.)

Simone Fischer-Hübner. *IT-Security and Privacy – Design and Use of Privacy-Enhancing Security Mechanisms*, volume 1958 of *LNCS*. Springer, 2001. ISBN 978-3-540-42142-9. doi: 10.1007/3-540-45150-1. (Cited on page 26.)

Gina Fisk, Calvin Ardi, Neale Pickett, John Heidemann, Mike Fisk, and Christos Papadopoulos. Privacy principles for sharing cyber security data. In *2015 IEEE Security and Privacy Workshops*, pages 193–197, May 2015. doi: 10.1109/spw.2015.23. (Cited on pages 45 and 337.)

Ariel Fuxman, Marco Pistore, John Mylopoulos, and Paolo Traverso. Model checking early requirements specifications in tropos. In *Proceedings Fifth IEEE International Symposium on Requirements Engineering*, pages 174–181, 2001. doi: 10.1109/isre.2001.948557. (Cited on page 13.)

Rafa Galvez and Seda Gürses. The odyssey: Modeling privacy threats in a brave new world. In *2018 IEEE European Symposium on Security and Privacy Workshops, EuroS&P Workshops 2018, London, United Kingdom, April 23-27, 2018*, pages 87–94. IEEE, 2018. doi: 10.1109/eurospw.2018.00018. URL `https://doi.org/10.1109/EuroSPW.2018.00018`. (Cited on page 422.)

Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995. ISBN 0-201-63361-2. (Cited on page 304.)

Jinglin Gao. Integration von privacy enhancing techologies (pets) in problem frames. Master thesis, Universität Duisburg-Essen, 2017. (Cited on page 324.)

Alessandro Garcia, Cláudio Sant'Anna, Eduardo Figueiredo, Uirá Kulesza, Carlos Lucena, and Arndt von Staa. *Modularizing Design Patterns with Aspects: A Quantitative Study*, pages 36–74. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-32974-9. doi: 10.1007/11687061_2. (Cited on page 304.)

Samuel Gibbs. Hackers can hijack wi-fi hello barbie to spy on your children. Online: `https://www.theguardian.com/technology/2015/nov/26/hackers-can-hijack-wi-fi-hello-barbie-to-spy-on-your-children`, 2015. accessed on 25. April 2017. (Cited on page 3.)

John C. Grundy. Aspect-oriented requirements engineering for component-based software systems. In *Proceedings of the IEEE International Symposium on Requirements Engineering*, pages 84–91, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0188-5. URL `http://dl.acm.org/citation.cfm?id=647646.731259`. (Cited on page 278.)

GSMA. MOBILE PRIVACY: Consumer research insights and considerations for policymakers. `http://www.gsma.com/publicpolicy/wp-content/uploads/2014/02/MOBILE_PRIVACY_Consumer_research_insights_and_considerations_for_policymakers-Final.pdf` (accessed on 20 June 2016), February 2014. (Cited on page 85.)

Paolo Guarda and Nicola Zannone. Towards the development of privacy-aware systems. *Inf. Softw. Technol.*, 51(2):337–350, feb 2009. ISSN 0950-5849. doi: 10.1016/j.infsof.2008.04.004. (Cited on page 94.)

Paolo Guarda, Silvio Ranise, and Hari Siswantoro. Security analysis and legal compliance checking for the design of privacy-friendly information systems. In *Proceedings of the 22Nd ACM on Symposium on Access Control Models and Technologies*, SACMAT '17 Abstracts, pages 247–254, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4702-0. doi: 10.1145/3078861.3078879. (Cited on pages 45, 61 and 337.)

Seda Gürses. *Multilateral Privacy Requirements Analysis in Online Social Network Services.* PhD thesis, Arenberg School, Katholieke Universiteit Leuven, 2010. (Cited on pages 25, 26 and 265.)

Seda Gürses and José M. del Álamo. Privacy engineering: Shaping an emerging field of research and practice. *IEEE Security & Privacy*, 14(2):40–46, 2016. ISSN 1540-7993. doi: 10.1109/MSP.2016.37. (Cited on page 5.)

Seda Gürses, Jens H. Jahnke, Christina Obry, Adeniyi Onabajo, Thomas Santen, and Morgan Price. Eliciting confidentiality requirements in practice. In *Proceedings of the 2005 conference of the Centre for Advanced Studies on Collaborative research*, CASCON '05, pages 101–116. IBM Press, 2005. (Cited on page 26.)

Seda Gürses, Carmela Troncoso, and Claudia Diaz. Engineering privacy by design. Technical report, Katholieke Universiteit Leuven, 2011. URL `https://www.esat.kuleuven.be/cosic/publications/article-1542.pdf`. (Cited on pages 55, 123, 264 and 337.)

Irit Hadar, Tomer Hasson, Oshrat Ayalon, Eran Toch, Michael Birnhack, Sofia Sherman, and Arod Balissa. Privacy by designers: software developers' privacy mindset. *Empirical Software Engineering*, Apr 2017. ISSN 1573-7616. doi: 10.1007/s10664-017-9517-1. (Cited on page 4.)

Munawar Hafiz. A pattern language for developing privacy enhancing technologies. *Software: Practice and Experience*, 43(7):769–787, 2013. ISSN 1097-024X. doi: 10.1002/spe.1131. (Cited on page 325.)

Jon Hall and Lucia Rapanotti. Problem frames for socio-technical systems. In Panayiotis Zaphiris and Chee Siang Ang, editors, *Human Computer Interaction: Concepts, Methodologies, Tools, and Applications*, pages 713–731. Information Science Reference, Hershey, PA, 2009. URL `http://oro.open.ac.uk/24513/`. (Cited on page 19.)

Marit Hansen. Top 10 mistakes in system design from a privacy perspective and privacy protection goals. In *Privacy and Identity Management for Life*, IFIP AICT 375, pages 14–31. Springer, 2012. ISBN 978-3-642-31667-8. doi: 10.1007/978-3-642-31668-5_2. (Cited on pages 73, 82, 85, 94, 97 and 105.)

Marit Hansen, Meiko Jensen, and Martin Rost. Protection goals for privacy engineering. In *2015 IEEE Security and Privacy Workshops*, pages 159–166, May 2015. doi: 10.1109/spw.2015.13. (Cited on pages 28, 30, 55, 61, 97, 123, 411, 412 and 414.)

Neil Harrison. Advanced pattern writing. In *Proceedings of the 8th European Conference on Pattern Languages of Programms (EuroPLoP '2003), Irsee, Germany, June 25-29, 2003.*, pages 809–828. UVK - Universitaetsverlag Konstanz, 2003. (Cited on pages 281, 308 and 416.)

Neil B. Harrison. Advanced pattern writing – patterns for experienced pattern authors. In Dragos Manolescu, Markus Voelter, and James Noble, editors, *Pattern Languages of Program Design 5*. Addison-Wesley, 2006. (Cited on page 324.)

Denis Hatebur, Maritta Heisel, and Holger Schmidt. A formal metamodel for problem frames. Technical report, Springer, 2008. URL `https://link.springer.com/`. (Cited on page 20.)

Hans Hedbom. A survey on transparency tools for enhancing privacy. In *The Future of Identity in the Information Society*, IFIP AICT 298, pages 67–82. Springer, 2009. ISBN 978-3-642-03314-8. doi: 10.1007/978-3-642-03315-5_5. (Cited on pages 82 and 94.)

Jaap-Henk Hoepman. Privacy design strategies - (extended abstract). In *ICT Systems Security and Privacy Protection - 29th IFIP TC 11 International Conference, SEC*, IFIP AICT 428, pages 446–459. Springer, 2014. doi: 10.1007/978-3-642-55415-5_38. (Cited on pages 38, 44, 61, 82 and 94.)

Hubert F. Hofmann and Franz Lehner. Requirements engineering as a success factor in software projects. *IEEE Software*, 18(4):58–66, Jul 2001. ISSN 0740-7459. doi: 10.1109/ms.2001.936219. (Cited on page 269.)

Jason I. Hong, Jennifer D. Ng, Scott Lederer, and James A. Landay. Privacy risk models for designing privacy-sensitive ubiquitous computing systems. In David Benyon, Paul Moody, Dan Gruen, and Irene McAra-McWilliam, editors, *Proceedings of the Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, Cambridge, MA, USA, August 1-4, 2004*, pages 91–100. ACM, 2004. doi: 10.1145/1013115.1013129. (Cited on pages 51, 123, 264 and 337.)

Micheal Howard and Steve Lipner. *The Security Development Lifecycle*. Microsoft Press, Redmond, WA, USA, 2006. ISBN 0735622140. (Cited on pages 48 and 58.)

Vincent C. Hu, David Ferraiolo, Rick Kuhn, Adam Schnitzer, Kenneth Sandlin, Robert Miller, and Karen Scarfone. Guide to attribute based access control (abac) definition and considerations. Technical report, NIST, 2014. (Cited on page 49.)

Troy Hunt. Data from connected cloudpets teddy bears leaked and ransomed, exposing kids' voice messages. Online: `https://www.troyhunt.com/`

data-from-connected-cloudpets-teddy-bears-leaked-and-ransomed-exposing-kids_voice-messages/, 2017. accessed on 25. April 2017. (Cited on page 3.)

IBM. Rational doors. Online: http://www-03.ibm.com/software/products/en/ratidoor, 2017. accessed on 3. May 2017. (Cited on page 13.)

IEC. Hazard and operability studies (hazop studies). IEC 61882, International Electrotechnical Commission (IEC), 2001. (Cited on pages 246, 247, 250, 265 and 422.)

IEC. Fault tree analysis (fta) (iec 61025:2006). Technical report, International Electrotechnical Commission, 2006. (Cited on pages 261 and 262.)

Shareeful Islam, Haralambos Mouratidis, and Stefan Wagner. *Towards a Framework to Elicit and Manage Security and Privacy Requirements from Laws and Regulations*, pages 255–261. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-14192-8. doi: 10.1007/978-3-642-14192-8_23. URL https://doi.org/10.1007/978-3-642-14192-8_23. (Cited on pages 51, 242, 264 and 337.)

ISO and IEC. Common Criteria for Information Technology Security Evaluation – Part 2 Security functional components. ISO/IEC 15408, International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), 2009. (Cited on page 99.)

ISO/IEC. ISO/IEC 29100:2011 Information technology – Security techniques – Privacy Framework. Technical report, International Organization for Standardization and International Electrotechnical Commission, 2011. (Cited on pages 25, 26, 27, 37, 44, 54, 74, 75, 83, 85, 86, 94, 123, 154, 155, 307, 325, 331, 412 and 414.)

ISO/IEC. *ISO/IEC 24744 Software engineering – Metamodel for development methodologies*, second edition edition, 2014. (Cited on pages 33 and 34.)

ISO/IEC. Information technology — security techniques — information security management systems — overview and vocabulary. Technical report, International Organization for Standardization and International Electrotechnical Commission, 2016. (Cited on pages 28 and 100.)

ISO/IEC. ISO/IEC 29134:2017 Information technology – Security techniques – Guidelines for privacy impact assessment. Technical report, International Organization for Standardization and International Electrotechnical Commission, 2017a. (Cited on pages 343, 344, 348, 355, 416 and 441.)

ISO/IEC. ISO/IEC 29151:2017 Information technology – Security techniques – Code of practice for personally identifiable information protection. Technical report, International Organization for Standardization and International Electrotechnical Commission, 2017b. (Cited on page 331.)

Michael Jackson. *Problem Frames: Analysing & Structuring Software Development Problems.* Addison-Wesley Professional, 2000. ISBN 9780201596274. (Cited on pages xix, 8, 10, 13, 15, 16, 17, 18, 19, 20, 21, 62, 270, 273, 279, 281, 324, 337, 411 and 413.)

Ivar Jacobson and Pan-Wei Ng. *Aspect-Oriented Software Development with Use Cases.* Addison-Wesley Professional, 2004. ISBN 0321268881. (Cited on page 278.)

Samireh Jalali and Claes Wohlin. Systematic literature studies: Database searches vs. backward snowballing. In *Proc. of the ACM-IEEE Int. Symp. on Empirical Software Engineering and Measurement*, ESEM 2012, pages 29–38. ACM, 2012. ISBN 978-1-4503-1056-7. doi: 10.1145/2372251.2372257. (Cited on pages 32, 81 and 93.)

Carlos Jensen, Joseph Tullio, Colin Potts, and Elizabeth D. Mynatt. Strap: A structured analysis framework for privacy. Technical report, Gergia Institute of Technology, 2005. URL `http://hdl.handle.net/1853/4450`. (Cited on pages 51, 264 and 337.)

Nan Jiang. Analysis of privacy threats identified during privacy and operability studies using problem frame models. Master thesis, Universität Duisburg-Essen, 2018. (Cited on page 255.)

Ming Jin, Ruoxi Jia, Zhaoyi Kang, Ioannis C. Konstantakopoulos, and Costas J. Spanos. Presencesense: Zero-training algorithm for individual presence detection based on power monitoring. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, BuildSys '14, pages 1–10, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3144-9. doi: 10.1145/2674061.2674073. (Cited on page 4.)

Richard Jones and Dalal Tahri. EU law requirements to provide information to website visitors. *Computer Law & Security Review*, 26(6):613 – 620, 2010. ISSN 0267-3649. doi: 10.1016/j.clsr.2010.09.006. URL `http://www.sciencedirect.com/science/article/pii/S0267364910001457`. (Cited on page 83.)

Jan Jürjens. *Secure Systems Development with UML*. Springer-Verlag, Berlin, Heidelberg, 2010. ISBN 3642056350, 9783642056352. (Cited on page 48.)

Dawn N. Jutla, Peter Bodorik, and Sohail Ali. Engineering privacy for big data apps with the unified modeling language. In *IEEE International Congress on Big Data, BigData Congress 2013, June 27 2013-July 2, 2013*, pages 38–45. IEEE, 2013. doi: 10.1109/bigdata.congress.2013.15. URL `https://doi.org/10.1109/BigData.Congress.2013.15`. (Cited on pages 39, 121 and 122.)

Christos Kalloniatis. Designing privacy-aware systems in the cloud. In *Trust, Privacy and Security in Digital Business - 12th International Conference, TrustBus 2015*, volume 9264 of *LNCS*, pages 113–123. Springer, 2015. doi: 10.1007/978-3-319-22906-5_9. (Cited on page 94.)

Christos Kalloniatis, Evangelia Kavakli, and Stefanos Gritzalis. Addressing privacy requirements in system design: the pris method. *Requirements Engineering*, 13(3):241–255, Sep 2008. ISSN 1432-010X. doi: 10.1007/s00766-008-0067-3. (Cited on pages 45, 121, 122, 123, 242 and 337.)

Christos Kalloniatis, Evangelia Kavakli, and Efstathios Kontellis. Pris tool: A case tool for privacy-oriented requirements engineering. In *MCIS 2009 Proceedings*. AIS Electronic Library (AISeL), 2009. URL `https://aisel.aisnet.org/mcis2009/71`. (Cited on page 45.)

Christos Kalloniatis, Haralambos Mouratidis, Manousakis Vassilis, Shareeful Islam, Stefanos Gritzalis, and Evangelia Kavakli. Towards the design of secure and privacy-oriented information systems in the cloud: Identifying the major concepts. *Computer Standards & Interfaces*, 36(4):759 – 775, 2014. ISSN 0920-5489. doi: 10.1016/j.csi.2013.12.010. URL `http://www.sciencedirect.com/science/article/pii/S0920548913001840`. (Cited on pages 82 and 94.)

Patrick Gage Kelley, Joanna Bresee, Lorrie Faith Cranor, and Robert W. Reeder. A "nutrition label" for privacy. In *Proc. of the 5th Symp. on Usable Privacy and Security*, SOUPS '09, pages 4:1–4:12. ACM, 2009. ISBN 978-1-60558-736-3. doi: 10.1145/1572532.1572538. (Cited on page 83.)

Patrick Gage Kelley, Lucian Cesca, Joanna Bresee, and Lorrie Faith Cranor. Standardizing privacy notices: An online study of the nutrition label approach. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, CHI '10, pages 1573–1582. ACM, 2010. ISBN 978-1-60558-929-9. doi: 10.1145/1753326.1753561. (Cited on page 83.)

Fabian Knirsch, Dominik Engel, Christian Neureiter, Marc Frincu, and Viktor Prasanna. Model-driven privacy assessment in the smart grid. In *2015 International Conference on Information Systems Security and Privacy (ICISSP)*, pages 1–9, Feb 2015. (Cited on pages 40, 203 and 264.)

S. Komanduri, R. Shay, B. Ur G. Norcie, and Lorrie Faith Cranor. Adchoices? compliance with online behavioral advertising notice and choice requirements. Technical report, CyLab – Carnegie Mellon University, 2011. `https://www.cylab.cmu.edu/files/pdfs/tech_reports/CMUCyLab11005.pdf` (accessed on 20 June 2016). (Cited on page 94.)

Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Patrick Schweitzer. *Foundations of Attack–Defense Trees*, pages 80–95. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-19751-2. doi: 10.1007/978-3-642-19751-2_6. URL `https://doi.org/10.1007/978-3-642-19751-2_6`. (Cited on page 40.)

Martin Kost and Johann-Christoph Freytag. Privacy analysis using ontologies. In *Proceedings of the Second ACM Conference on Data and Application Security and Privacy*, CODASPY '12, pages 205–216, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1091-8. doi: 10.1145/2133601.2133627. (Cited on page 39.)

Martin Kost, Johann-Christoph Freytag, Frank Kargl, and Antonio Kung. Privacy verification using ontologies. In *2011 Sixth International Conference on Availability, Reliability and Security*, pages 627–632, Aug 2011. doi: 10.1109/ares.2011.97. (Cited on pages 38 and 39.)

Kat Krol and Sören Preibusch. Effortless privacy negotiations. *IEEE Security & Privacy*, 13 (3):88–91, 2015. (Cited on page 94.)

P. Kruchten. *The Rational Unified Process: An Introduction.* The Addison-Wesley object technology series. Addison-Wesley, 2004. ISBN 9780321197702. URL `https://books.google.de/books?id=RYCMx6o47pMC`. (Cited on page 55.)

Antonio Kung, Johann-Christoph Freytag, and Frank Kargl. Privacy-by-design in its applications. In *IEEE Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6, June 2011. doi: 10.1109/wowmom.2011.5986166. (Cited on pages 43, 82 and 337.)

Marc Langheinrich. Privacy by design — principles of privacy-aware ubiquitous systems. In *Ubiquitous Computing (Ubicomp)*, LNCS 2201, pages 273–291. Springer, 2001. ISBN 978-3-540-42614-1. doi: 10.1007/3-540-45427-6_23. (Cited on pages 82 and 94.)

Maria Lencastre, João Araújo, Ana Moreira, and Jaelson Castro. Analyzing crosscutting in the problem frames approach. In *Proc. of the 2006 int. workshop on Advances and applications of problem frames (IWAAPF)*, pages 59–64. ACM, 2006. (Cited on page 278.)

Maria Lencastre, Ana Moreira, João Araújo, and Jaelson Castro. Aspects composition in problem frames. In *Proc. of the 2008 16th IEEE Int. Requirements Engineering Conf.*, pages 343–344. IEEE Computer Society, 2008. (Cited on pages 278 and 304.)

Christopher Lentzsch, Kai-Uwe Loser, Martin Degeling, and Alexander Nolte. *Integrating a Practice Perspective to Privacy by Design*, pages 691–702. Springer International Publishing, Cham, 2017. ISBN 978-3-319-58460-7. doi: 10.1007/978-3-319-58460-7_47. URL `https://doi.org/10.1007/978-3-319-58460-7_47`. (Cited on pages 47 and 337.)

William Lidwell, Kritina Holden, and Jill Butler. *Universal Principles of Design.* Rockport Publishers, 2010. (Cited on page 55.)

Lin Liu and Zhi Jin. Integrating goals and problem frames in requirements analysis. In *14th IEEE International Requirements Engineering Conference (RE'06)*, pages 349–350, Sept 2006. doi: 10.1109/re.2006.34. (Cited on page 62.)

Lin Liu, Eric Yu, and John Mylopoulos. Security and privacy requirements analysis within a social setting. In *Proceedings of the 11th IEEE International Conference on Requirements Engineering*, RE '03, pages 151–161, Washington, DC, USA, 2003. IEEE Computer Society. ISBN 0-7695-1980-6. URL `http://dl.acm.org/citation.cfm?id=942807.943910`. (Cited on pages 48, 150, 264 and 337.)

Luanna Lopes Lobato, Eduardo B. Fernandez, and Sérgio Donizetti Zorzo. Patterns to support the development of privacy policies. In *Int. Conf. on Availability, Reliability and Security (ARES)*, pages 744–749, March 2009. doi: 10.1109/ares.2009.114. (Cited on pages 83, 94 and 325.)

Mass Soldal Lund, Bjørnar Solhaug, and Ketil Stølen. *Model-Driven Risk Analysis. The CORAS Approach.* Springer, 2010. (Cited on pages 257, 258, 259, 262 and 353.)

Ashwin Machanavajjhala, Johannes Gehrke, Daniel Kifer, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering, ICDE*, page 24. IEEE Computer Society, 2006. doi: 10.1109/icde.2006.1. URL `https://doi.org/10.1109/ICDE.2006.1`. (Cited on pages 319 and 322.)

Eleni-Laskarina Makri and Costas Lambrinoudakis. Privacy principles: Towards a common privacy audit methodology. In *Trust, Privacy and Security in Digital Business - 12th International Conference, TrustBus 2015*, volume 9264 of *LNCS*, pages 219–234. Springer, 2015. (Cited on page 94.)

Yod-Samuel Martín and José M. del Álamo. A metamodel for privacy engineering methods. *CEUR Workshop Proceedings*, 1873:41–48, 2017. ISSN 16130073. (Cited on pages 33, 34 and 64.)

Betsy Masiello. Deconstructing the privacy experience. *Security Privacy, IEEE*, 7(4):68–70, July 2009. ISSN 1540-7993. doi: 10.1109/msp.2009.88. (Cited on pages 82 and 94.)

Erika McCallister and Karen Scarfone. Guide to protecting the confidentiality of personally identifiable information (PII). Special Publication 800-122, National Institute of Standards and Technology (NIST), 2010. URL `https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-122.pdf`. (Cited on page 151.)

Nancy R. Mead and Ted Stehney. Security quality requirements engineering (square) methodology. In *Proceedings of the 2005 Workshop on Software Engineering for Secure Systems&Mdash;Building Trustworthy Applications*, SESS '05, pages 1–7, New York, NY, USA, 2005. ACM. ISBN 1-59593-114-7. doi: 10.1145/1082983.1083214. (Cited on page 41.)

Nancy R. Mead, Seiya Miyazaki, and Justin Zhan. Integrating privacy requirements considerations into a security requirements engineering method and tool. *IJIPSI*, 1(1):106–126, 2011. doi: 10.1504/ijipsi.2011.043733. URL `https://doi.org/10.1504/IJIPSI.2011.043733`. (Cited on pages 41, 61, 242, 243 and 264.)

Rene Meis. Problem-Based Consideration of Privacy-Relevant Domain Knowledge. In *Privacy and Identity Management for Emerging Services and Technologies*, volume 421 of *IFIP Advances in Information and Communication Technology.* Springer, 2014. doi: 10.1007/

978-3-642-55137-6_12. URL http://dx.doi.org/10.1007/978-3-642-55137-6_12. (Cited on pages 19, 133 and 147.)

Rene Meis and Maritta Heisel. Systematic identification of information flows from requirements to support privacy impact assessments. In *ICSOFT-PT 2015 - Proc. of the 10th Int. Conf. on Software Paradigm Trends*, pages 43–52. SciTePress, 2015. doi: 10.5220/0005518500430052. URL http://dx.doi.org/10.5220/0005518500430052. (Cited on pages 151, 171 and 343.)

Rene Meis and Maritta Heisel. Supporting privacy impact assessments using problem-based privacy analysis. In *Software Technologies - 10th International Joint Conference, ICSOFT 2015, Revised Selected Papers*, volume 586 of *Communications in Computer and Information Science*, pages 79–98. Springer, 2016a. ISBN 978-3-319-30141-9. doi: 10.1007/978-3-319-30142-6_5. URL http://dx.doi.org/10.1007/978-3-319-30142-6_5. (Cited on pages 151, 171 and 343.)

Rene Meis and Maritta Heisel. Computer-aided identification and validation of privacy requirements. *Information*, 7(28), 2016b. ISSN 2078-2489. doi: 10.3390/info7020028. URL http://www.mdpi.com/2078-2489/7/2/28. (Cited on pages 97 and 205.)

Rene Meis and Maritta Heisel. Understanding the privacy goal intervenability. In *Trust, Privacy, and Security in Digital Business*, volume 9830 of *LNCS*, pages 79–94. Springer, 2016c. doi: 10.1007/978-3-319-44341-6_6. URL https://link.springer.com/chapter/10.1007/978-3-319-44341-6_6. (Cited on page 85.)

Rene Meis and Maritta Heisel. Aspect frames – describing cross-cutting concerns in aspect-oriented requirements engineering. In *Proceedings of the 22nd European Conference on Pattern Languages of Programs*, number 25 in EuroPLoP '17, page 28. ACM, 2017a. doi: 3147704.3147732. URL https://doi.org/10.1145/3147704.3147732. (Cited on page 281.)

Rene Meis and Maritta Heisel. Computer-aided identification and validation of intervenability requirements. *Information*, 8(30), 2017b. ISSN 2078-2489. doi: 10.3390/info8010030. URL http://www.mdpi.com/2078-2489/8/1/30. (Cited on pages 85, 97 and 205.)

Rene Meis and Maritta Heisel. Towards systematic privacy and operability (PRIOP) studies. In *ICT Systems Security and Privacy Protection*, volume 502 of *IFIP AICT*, pages 427–441. Springer, 2017c. doi: 10.1007/978-3-319-58469-0_29. URL http://dx.doi.org/10.1007/978-3-319-58469-0_29. (Cited on page 245.)

Rene Meis and Maritta Heisel. Pattern-based representation of privacy enhancing technologies as early aspects. In *Trust, Privacy, and Security in Digital Business*, volume 10442 of *LNCS*, pages 49–65, Cham, 2017d. Springer International Publishing. doi: 10.1007/978-3-319-64483-7_4. URL https://doi.org/10.1007/978-3-319-64483-7_4. (Cited on page 307.)

Rene Meis, Maritta Heisel, and Roman Wirtz. A taxonomy of requirements for the privacy goal transparency. In *Trust, Privacy, and Security in Digital Business*, LNCS 9264, pages 195–209. Springer, 2015. doi: 10.5220/0005518500430052. URL http://dx.doi.org/10.5220/0005518500430052. (Cited on page 73.)

Peter Mell and Timothy Grance. The NIST definition of cloud computing. Special Publication 800-145, National Institute of Standards and Technology (NIST), 2011. (Cited on page 137.)

Seiya Miyazaki, Nancy R. Mead, and Justin Zhan. Computer-aided privacy requirements elicitation technique. In *IEEE Asia-Pacific Services Computing Conf. (APSCC)*, pages 367–372, Dec 2008. doi: 10.1109/apscc.2008.263. (Cited on pages 41, 82 and 94.)

Nazila Gol Mohammadi, Azadeh Alebrahim, Thorsten Weyer, Maritta Heisel, and Klaus Pohl. A framework for combining problem frames and goal models to support context analysis during requirements engineering. In Alfredo Cuzzocrea, Christian Kittl, Dimitris E. Simos, Edgar Weippl, and Lida Xu, editors, *Proceedings of the 5th International Cross-Domain Conference on Availability, Reliability, and Security in Information Systems and HCI (CD-ARES)*, LNCS 8127, pages 272–288. Springer, 2013. URL `https://link.springer.com/`. (Cited on pages 62 and 278.)

Ana Moreira, João Araújo, and Awais Rashid. A concern-oriented requirements engineering model. In *Proc. Int'l Conf. on Advanced Information Systems Engineering (CAiSE) 2005, LNCS*, pages 293–308. Springer, 2005. (Cited on page 278.)

Haralambos Mouratidis and Paolo Giorgini. Secure tropos: A security-oriented extension of the tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, 17(2):285–309, April 2007. URL `http://roar.uel.ac.uk/415/`. (Cited on pages 45 and 51.)

Haralambos Mouratidis, Shareeful Islam, Christos Kalloniatis, and Stefanos Gritzalis. A framework to support selection of cloud providers based on security and privacy requirements. *Journal of Systems and Software*, 86(9):2276 – 2293, 2013. ISSN 0164-1212. doi: 10.1016/j.jss.2013.03.011. URL `http://www.sciencedirect.com/science/article/pii/S0164121213000575`. (Cited on pages 82 and 94.)

Erik T. Mueller. *Commonsense Reasoning*. Morgan Kaufmann, 2006. (Cited on page 40.)

Deirdre K. Mulligan. The enduring importance of transparency. *Security Privacy, IEEE*, 12(3): 61–65, May 2014. ISSN 1540-7993. doi: 10.1109/msp.2014.58. (Cited on page 83.)

Pradeep K. Murukannaiah, Anup K. Kalia, Pankaj R. Telangy, and Munindar P. Singh. Resolving goal conflicts via argumentation-based analysis of competing hypotheses. In *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, pages 156–165, Aug 2015. doi: 10.1109/re.2015.7320418. (Cited on page 51.)

Pradeep K. Murukannaiah, Nirav Ajmeri, and Munindar P. Singh. Engineering privacy in social applications. *IEEE Internet Computing*, 20(2):72–76, Mar 2016. ISSN 1089-7801. doi: 10.1109/mic.2016.30. (Cited on pages 51, 123 and 337.)

Helen Nissenbaum. Privacy as contextual integrity. *Washington Law Review*, 79(1):119–158, 2004. (Cited on pages 40, 120, 121 and 123.)

Nicolás Notario, Alberto Crespo, Yod-Samuel Martín, José M. del Álamo, Daniel Le Métayer, Thibaud Antignac, Antonio Kung, Inga Kroener, and David Wright. Pripare: Integrating privacy best practices into a privacy engineering methodology. In *2015 IEEE Security and Privacy Workshops*, pages 151–158, May 2015. doi: 10.1109/spw.2015.22. (Cited on pages 36 and 53.)

Obeo and Thales. Sirius - the easiest way to get your own modeling tool, 2017. URL `http://www.eclipse.org/sirius/`. accessed on 13 June 2017. (Cited on page 20.)

Object Management Group. *Object Constraint Language*. Object Management Group, 2014. URL `http://www.omg.org/spec/OCL/2.4/`. (Cited on pages 21 and 158.)

OECD. OECD guidelines on the protection of privacy and transborder flows of personal data. Technical report, Organisation of Economic Co-Operation and Development, 1980. (Cited on pages 27, 41, 42, 43, 75 and 86.)

Marie Caroline Oetzel and Sarah Spiekermann. A systematic methodology for privacy impact assessments: a design science approach. *European Journal of Information Systems*, 23(2): 126–150, 2014. (Cited on pages 51, 242, 264, 337 and 354.)

Ian Oliver. Experiences in the development and usage of a privacy requirements framework. In *2016 IEEE 24th International Requirements Engineering Conference (RE)*, pages 293–302, Sept 2016. doi: 10.1109/re.2016.59. (Cited on pages 54, 169, 202, 203, 242, 264 and 337.)

Paul N. Otto, Annie I. Antón, and David L. Baumer. The ChoicePoint dilemma: How data brokers should handle the privacy of personal information. *Security Privacy, IEEE*, 5(5): 15–23, Sept 2007. ISSN 1540-7993. doi: 10.1109/msp.2007.126. (Cited on page 83.)

David L. Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, December 1972. ISSN 0001-0782. doi: 10.1145/361598.361623. (Cited on pages 269 and 270.)

Michalis Pavlidis, Haralambos Mouratidis, Emmanouil Panaousis, and Nikolaos Argyropoulos. Selecting security mechanisms in secure tropos. In Javier Lopez, Simone Fischer-Hübner, and Costas Lambrinoudakis, editors, *Trust, Privacy and Security in Digital Business*, pages 99–114, Cham, 2017. Springer International Publishing. ISBN 978-3-319-64483-7. (Cited on page 339.)

Charith Perera, Ciaran McCormick, Arosha K. Bandara, Blaine A. Price, and Bashar Nuseibeh. Privacy-by-design framework for assessing internet of things applications and platforms. In *Proceedings of the 6th International Conference on the Internet of Things*, IoT'16, pages 83–92, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4814-0. doi: 10.1145/2991561.2991566. (Cited on pages 38 and 242.)

Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, August 2010. URL `http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf`. v0.34. (Cited on pages 28, 97, 105, 107, 230, 311 and 316.)

Shari Lawrence Pfleeger. *Software Engineering: Theory and Practice.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1998. ISBN 0-13-624842-X. (Cited on page 13.)

Jan Porekar, Aljosa Jerman-Blazic, and Tomaz Klobucar. Towards organizational privacy patterns. In *Second International Conference on the Digital Society*, pages 15–19, Feb 2008. doi: 10.1109/icds.2008.27. (Cited on page 325.)

Stefanie Pötzsch. Privacy awareness: A means to solve the privacy paradox? In *The Future of Identity in the Information Society*, IFIP AICT 298, pages 226–236. Springer, 2009. ISBN 978-3-642-03314-8. doi: 10.1007/978-3-642-03315-5_17. (Cited on page 82.)

Athanasia Pouloudi. Aspects of the stakeholder concept and their implications for information systems development. In *HICSS*, 1999. (Cited on page 149.)

Thomas Probst and Marit Hansen. Privacy protection goals in privacy and data protection evaluations. Working paper, Unabhängiges Landeszentrum für Datenschutz Schleswig-Holstein, July 2013. (Cited on page 73.)

Audrey Mei Yi Quah and Uwe Röhm. User awareness and policy compliance of data privacy in cloud computing. In *Proceedings of the First Australasian Web Conference - Volume 144*, AWC '13, pages 3–12, Darlinghurst, Australia, Australia, 2013. Australian Computer Society, Inc.

ISBN 978-1-921770-29-6. URL http://dl.acm.org/citation.cfm?id=2527208.2527209. (Cited on page 85.)

Silvio Ranise and Hari Siswantoro. *Automated Legal Compliance Checking by Security Policy Analysis*, pages 361–372. Springer International Publishing, Cham, 2017. ISBN 978-3-319-66284-8. doi: 10.1007/978-3-319-66284-8_30. URL https://doi.org/10.1007/978-3-319-66284-8_30. (Cited on pages 45 and 337.)

Awais Rashid. Aspect-oriented requirements engineering: An introduction. In *Proceedings of the 16th IEEE International Requirements Engineering Conference*, pages 306–309. IEEE Computer Society, 2008. (Cited on pages 270, 272 and 278.)

Awais Rashid, Peter Sawyer, Ana Moreira, and João Araújo. Early aspects: A model for aspect-oriented requirements engineering. In *Joint Int'l Conf. Requirements Engineering (RE)*, pages 199–202. IEEE Computer Society Press, 2002. ISBN 0-7695-1465-0. (Cited on page 8.)

Joseph Reagle, Lorrie Faith Cranor, and Mark S. Ackerman. Privacy in e-Commerce: Examining User Scenarios and Privacy Preferences. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, pages 1–8, New York, NY, USA, 1999. ACM. (Cited on page 85.)

Lena Reinfelder, Zinaida Benenson, and Freya Gassmann. Differences between Android and iPhone users in their security and privacy awareness. In *Trust, Privacy, and Security in Digital Business*, LNCS 8647, pages 156–167. Springer, 2014. ISBN 978-3-319-09769-5. doi: 10.1007/978-3-319-09770-1_14. (Cited on page 82.)

Suzanne Robertson and James Robertson. *Mastering the Requirements Process (2nd Edition)*. Addison-Wesley Professional, 2006. ISBN 0321419499. (Cited on page 149.)

Sasha Romanosky, Alessandro Acquisti, Jason Hong, Lorrie Faith Cranor, and Batya Friedman. Privacy patterns for online interactions. In *Proceedings of the 2006 Conference on Pattern Languages of Programs*, PLoP '06, pages 12:1–12:9, New York, NY, USA, 2006. ACM. ISBN 978-1-60558-372-3. doi: 10.1145/1415472.1415486. (Cited on page 325.)

Martin Rost and Andreas Pfitzmann. Datenschutz-Schutzziele – revisited. *Datenschutz und Datensicherheit - DuD*, 33(6):353–358, 2009. ISSN 1614-0702. doi: 10.1007/s11623-009-0072-9. (Cited on page 82.)

Bruce Schneier. Attack trees: Modeling security threats. *Dr. Dobb's Journal*, 1999. URL https://www.schneier.com/academic/archives/1999/12/attack_trees.html. (Cited on page 261.)

Markus Schumacher. Security patterns and security standards - with selected security patterns for anonymity and privacy. In *European Conference on Pattern Languages of Programs (EuroPLoP)*, 2003. (Cited on page 325.)

Robert Seater, Daniel Jackson, and Rohit Gheyi. Requirement progression in problem frames: deriving specifications from requirements. *Requirements Engineering*, 12(2):77–102, 2007. ISSN 1432-010X. doi: 10.1007/s00766-007-0048-y. (Cited on pages 14 and 24.)

Awanthika Senarath, Nalin A. G. Arachchilage, and Jill Slay. *Designing Privacy for You: A Practical Approach for User-Centric Privacy*, pages 739–752. Springer International Publishing, Cham, 2017. ISBN 978-3-319-58460-7. doi: 10.1007/978-3-319-58460-7_50. URL https://doi.org/10.1007/978-3-319-58460-7_50. (Cited on pages 55, 56, 242 and 337.)

Helen Sharp, Anthony Finkelstein, and Galal Galal. Stakeholder identification in the requirements engineering process. In *DEXA Workshop*, pages 387–391, 1999. (Cited on pages 136 and 149.)

Kim Bartel Sheehan and Mariea Grubbs Hoy. Dimensions of privacy concern among online consumers. *Journal of Public Policy & Marketing*, 19(1):62–73, 2000. ISSN 07439156. URL `http://www.jstor.org/stable/30000488`. (Cited on page 82.)

Swapneel Sheth, Gail Kaiser, and Walid Maalej. Us and them: A study of privacy requirements across North America, Asia, and Europe. In *Proc. of the 36th Int. Conf. on Software Engineering*, ICSE 2014, pages 859–870. ACM, 2014. ISBN 978-1-4503-2756-5. doi: 10.1145/2568225.2568244. (Cited on pages 82 and 94.)

H. Jeff Smith, Tamara Dinev, and Heng Xu. Information privacy research: An interdisciplinary review. *MIS Q.*, 35(4):989–1016, dec 2011. ISSN 0276-7783. URL `http://dl.acm.org/citation.cfm?id=2208940.2208950`. (Cited on page 94.)

Daniel J. Solove. A taxonomy of privacy. *University of Pennsylvania Law Review*, 154(3): 477–560, Januar 2006. (Cited on pages 51 and 83.)

Sarah Spiekermann and Lorrie Faith Cranor. Engineering privacy. *IEEE Transactions on Software Engineering*, 35(1):67–82, Jan 2009. ISSN 0098-5589. doi: 10.1109/tse.2008.88. (Cited on pages 5, 55, 82, 94, 123, 264 and 337.)

Dave Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework*. Addison-Wesley Professional, 2nd edition edition, 2008. (Cited on page 20.)

Dominik Stöhr. Presenting transparency enhancing technologies (tets) as patterns. Bachelor thesis, Universität Duisburg-Essen, 2018. (Cited on page 324.)

Lee S. Strickland and Laura E. Hunt. Technology, security, and individual privacy: New tools, new threats, and new public perceptions: Research articles. *J. Am. Soc. Inf. Sci. Technol.*, 56(3):221–234, feb 2005. ISSN 1532-2882. doi: 10.1002/asi.v56:3. (Cited on page 94.)

Stanley M. Sutton, Jr. and Isabelle Rouvellou. Modeling of software concerns in cosmos. In *Proceedings of the 1st International Conference on Aspect-oriented Software Development*, AOSD '02, pages 127–133, New York, NY, USA, 2002. ACM. ISBN 1-58113-469-X. doi: 10.1145/508386.508402. (Cited on page 278.)

Latanya Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, October 2002. ISSN 0218-4885. doi: 10.1142/s0218488502001648. (Cited on pages 319 and 322.)

Symantec. State of Privacy Report 2015. `https://www.symantec.com/content/en/us/about/presskits/b-state-of-privacy-report-2015.pdf` (accessed on 20 June 2016), 2015. (Cited on page 85.)

Yung Shin Van Der Sype and Jean-Marc Seigneur. Case study: legal requirements for the use of social login features for online reputation updates. In Yookun Cho, Sung Y. Shin, Sang-Wook Kim, Chih-Cheng Hung, and Jiman Hong, editors, *Symposium on Applied Computing, SAC*, pages 1698–1705. ACM, 2014. ISBN 978-1-4503-2469-4. doi: 10.1145/2554850.2554857. URL `http://dl.acm.org/citation.cfm?id=2554850`. (Cited on pages 83 and 94.)

John P. Tomaszewski. Are you sure you had a privacy incident? *Security Privacy, IEEE*, 4(6): 64–66, Nov 2006. ISSN 1540-7993. doi: 10.1109/msp.2006.143. (Cited on page 83.)

Thein T. Tun, Arosha K. Bandara, Blaine A. Price, Yijun Yu, Charles Haley, Inah Omoronyia, and Bashar Nuseibeh. Privacy arguments: Analysing selective disclosure requirements for mobile applications. In *2012 20th IEEE International Requirements Engineering Conference (RE)*, pages 131–140, Sept 2012. doi: 10.1109/re.2012.6345797. (Cited on pages 39 and 40.)

Fatih Turkmen, Jerry den Hartog, Silvio Ranise, and Nicola Zannone. *Analysis of XACML Policies with SMT*, pages 115–134. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015. ISBN 978-3-662-46666-7. doi: 10.1007/978-3-662-46666-7_7. URL https://doi.org/10.1007/978-3-662-46666-7_7. (Cited on page 45.)

Nelufar Ulfat-Bunyadi, Rene Meis, and Maritta Heisel. The six-variable model - context modelling enabling systematic reuse of control software. In *Proceedings of the 11th International Joint Conference on Software Technologies (ICSOFT 2016)*, pages 15–26. SciTePress, 2016. doi: 10.5220/0005944100150026. URL http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0005944100150026. (Cited on page 17.)

US Federal Trade Commission. Privacy online: A report to congress, June 1998. (Cited on pages 27, 28, 55, 75 and 86.)

G. W. van Blarkom, John J. Borking, and J. G. Eddy Olk. *Handbook of Privacy and Privacy-Enhancing Technologies*. College bescherming persoonsgegevens, 2003. URL https://www.andrewpatrick.ca/pisa/handbook/handbook.html. (Cited on pages 51, 52, 61, 169, 202, 264 and 337.)

Axel van Lamsweerde. From worlds to machines, 2009. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.156.2270. Software Requirements and Design: A Tribute to Michael Jackson, co-located with ICSE 2009. (Cited on page 15.)

Axel van Lamsweerde, Emmanual Letier, and Robert Darimont. Managing conflicts in goal-driven requirements engineering. *IEEE Trans. Softw. Eng.*, 24(11):908–926, November 1998. ISSN 0098-5589. doi: 10.1109/32.730542. (Cited on pages 13 and 15.)

Sauro Vicini, Francesco Alberti, Nicolás Notario, Alberto Crespo, Juan R. T. Pastoriza, and Alberto Sanna. Co-creating security-and-privacy-by-design systems. In *2016 11th International Conference on Availability, Reliability and Security (ARES)*, pages 768–775, Aug 2016. doi: 10.1109/ares.2016.74. (Cited on pages 57, 58, 264 and 337.)

Yang Wang, Gregory Norcie, Saranga Komanduri, Alessandro Acquisti, Pedro Giovanni Leon, and Lorrie Faith Cranor. "i regretted the minute i pressed share": A qualitative study of regrets on facebook. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, SOUPS '11, pages 10:1–10:16, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0911-0. doi: 10.1145/2078827.2078841. (Cited on page 4.)

Samuel D. Warren and Louis D. Brandeis. The Right to Privacy. *Harvard Law Review*, pages 193–220, 1890. (Cited on page 25.)

Tim Wellhausen and Andreas Fießer. How to write a pattern?: a rough guide for first-time pattern authors. In *16th European Conference on Pattern Languages of Programs, EuroPLoP 2011*, page 5. ACM, 2011. doi: 10.1145/2396716.2396721. (Cited on pages 281, 308 and 416.)

Ina Wentzlaff and Markus Specker. Pattern-based development of user-friendly web applications. In *Workshop Proceedings of the 6th International Conference on Web Engineering (ICWE'06)*, New York, USA, 2006. ACM Press. ISBN 1-59593-435-9. doi: 10.1145/1149993.1149996. URL http://dl.acm.org/citation.cfm?id=1149996. (Cited on page 19.)

Ole Werger. Presenting intervenability enhancing technologies (iets) as patterns. Bachelor thesis, Universität Duisburg-Essen, 2018. (Cited on page 324.)

Alan F. Westin. *Privacy and Freedom.* Atheneum, New York, 1967. (Cited on page 25.)

Jon Whittle and João Araújo. Scenario modelling with aspects. *IEE Proceedings Software*, 151 (4):157–171, Aug 2004. ISSN 1462-5970. (Cited on page 278.)

Stephen B. Wicker and Dawn E. Schrader. Privacy-aware design principles for information networks. *Proceedings of the IEEE*, 99(2):330–350, Feb 2011. ISSN 0018-9219. doi: 10.1109/jproc.2010.2073670. (Cited on pages 82 and 94.)

Roman Wirtz. Betrachtung von transparenzanforderungen in der anforderungsanalyse. Bachelor thesis, Universität Duisburg-Essen, 2014. (Cited on page 73.)

Roman Wirtz, Maritta Heisel, Rene Meis, Aida Omerovic, and Ketil Stølen. Problem-based elicitation of security requirements - the procor method. In *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2018, Funchal, Madeira, Portugal, March 23-24, 2018.*, pages 26–38. SciTePress, 2018. doi: 10.5220/0006669400260038. URL `https://doi.org/10.5220/0006669400260038`. (Cited on page 422.)

David Wright. The state of the art in privacy impact assessment. *Computer Law & Security Review*, 28(1):54 – 61, 2012. ISSN 0267-3649. doi: 10.1016/j.clsr.2011.11.007. URL `http://www.sciencedirect.com/science/article/pii/S026736491100183X`. (Cited on page 83.)

David Wright and Charles Raab. Privacy principles, risks and harms. *International Review of Law, Computers & Technology*, 28(3):277–298, 2014. (Cited on pages 83 and 94.)

David Wright, Kush Wadhwa, Paul De Hert, and Dariusz Kloza. A privacy impact assessment framework for data protection and privacy rights – Deliverable D1. Technical report, PIAF consortium, 2011. URL `https://piafproject.files.wordpress.com/2018/03/piaf_d1_21_sept2011revlogo.pdf`. (Cited on pages 10 and 343.)

George O. M. Yee. Adding privacy protection to distributed software systems. In *Proceedings of the 14th International Joint Conference on e-Business and Telecommunications (ICETE 2017)*, pages 351–358, 2017. ISBN 978-989-758-259-2. doi: 10.5220/0006434903510358. (Cited on pages 50, 51, 169, 202, 203, 264 and 337.)

Jessica D. Young. Commitment analysis to operationalize software requirements from privacy policies. *Requirements Engineering*, 16(1):33–46, Mar 2011. ISSN 1432-010X. doi: 10.1007/s00766-010-0108-6. URL `https://doi.org/10.1007/s00766-010-0108-6`. (Cited on page 47.)

William Young and Nancy G. Leveson. An integrated approach to safety and security based on systems theory. *Commun. ACM*, 57(2):31–35, February 2014. ISSN 0001-0782. doi: 10.1145/2556938. (Cited on page 265.)

Wu Youyou, Michal Kosinski, and David Stillwell. Computer-based personality judgments are more accurate than those made by humans. *Proceedings of the National Academy of Sciences*, 112(4):1036–1040, 2015. doi: 10.1073/pnas.1418680112. URL `http://www.pnas.org/content/112/4/1036.abstract`. (Cited on page 4.)

Eric Yu and Luiz Marcio Cysneiros. Designing for privacy and other competing requirements. In *Proceedings of the 3rd Symposium on Requirements Engineering for Information Security*, CERIAS '02, pages 5:1–5:15, West Lafayette, IN, 2002. CERIAS - Purdue University. URL `http://dl.acm.org/citation.cfm?id=2835417.2835422`. (Cited on pages 43 and 337.)

Eric S. K. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering*, RE '97, pages 226–, Washington, DC, USA, 1997. IEEE Computer Society. ISBN 0-8186-7740-6. URL `http://dl.acm.org/citation.cfm?id=827255.827807`. (Cited on page 13.)

Linfei Yu. Tool-support for the identification of privacy threats from propan models. Master thesis, Universität Duisburg-Essen, 2017. (Cited on page 255.)

Yijun Yu, Cesar S. Sampaio do Prado Leite, and John Mylopoulos. From goals to aspects: Discovering aspects from requirements goal models. In *Proceedings of the 12th IEEE International Requirements Engineering Conference*, pages 38–47. IEEE Computer Society, 2004. (Cited on page 278.)

Pamela Zave and Michael Jackson. Four dark corners of requirements engineering. *ACM Trans. Softw. Eng. Methodol.*, 6(1):1–30, January 1997. ISSN 1049-331X. doi: 10.1145/237432.237434. (Cited on pages 14, 328 and 335.)

Moshe Zviran. User's perspectives on privacy in web-based applications. *Journal of Computer Information Systems*, 48(4):97–105, 2008. (Cited on page 82.)