scripted

Volume 9, Issue 3, December, 2012

ENGINEERING COMPLIANT SOFTWARE:

ADVISING DEVELOPERS BY AUTOMATING LEGAL REASONING

Daniel Oberle[†] *& Felix Drefs*[*] *& Richard Wacker*[*]
& Christian Baumann[†] *& Oliver Raabe*[*]

Abstract

The impact of software on human interactions is ever increasing. However, software developers are often unaware of statutory provisions that regulate human interactions. As a consequence, software is increasingly coming into conflict with such provisions. Therefore, this paper contributes an approach for advising the developer in designing software that complies to statutory provisions. The approach relies on the formalisation of statutory provisions and semi-automated legal reasoning assisted by the developer.

DOI: 10.2966/scrip.090312.280

[†] SAP Research Karlsruhe, Germany.
d.oberle@sap.com, ch.baumann@sap.com

[*] Institute of Information and Economic Law, Karlsruhe Institute of Technology (KIT), Germany. felix.drefs@kit.edu, richard.wacker@kit.edu, oliver.raabe@kit.edu

1. Introduction

Software increasingly mediates human interactions by regulating actions in web-shops, social networks, or eGovernment. As a response, legislation lays out detailed provisions in specific areas, e.g., the Federal Data Protection Act (FDPA) or Telemedia Act (TMA) in Germany. Such provisions comprise legal requirements including those for software.

The provisions, however are often not conveyed to the addressees, namely, the software developers. In larger software development projects, e.g., when professionally developing comprehensive enterprise applications, the developer either has to painstakingly follow “product guidelines” as part of a strictly defined software engineering process, or a legal expert has to be consulted. In the first case, the guidelines are typically generic, rigid and limited to specific acts, such as Sarbanes-Oxley.[1] In the second case, the legal expert has to assess the given software, which requires a “translation” from the software domain to the legal domain. During this “translation” aspects might be lost or misinterpreted.

In smaller development projects, such as rapid prototyping of mobile apps, the software can be developed rather quickly and effortlessly, even by a single younger programmer, who would be less likely to be aware of legal requirements on software.

Even in the unlikely case that the programmer is aware of all relevant statutory provisions, he or she is typically unable to determine and interpret the “legal norms” (typically represented by a paragraph, section or sentence in a provision). In addition, the assessment of compliance to legislation cannot always be given in advance because Internet-based, cloud and mobile apps are often mashed-up with other apps. This newly created mash-up might result in a different legal assessment than its individual parts.

In all cases, a proper assessment requires in-depth legal reasoning which is a complex intellectual process, so far only executable manually by experienced legal experts. Somewhat simplified, the legal expert – most likely by experience – has some idea of the legal consequence he or she wishes to attain. This, in turn, determines a limited set of norms from which to start. Each of these norms must be inspected in order to determine whether the given situation matches the state of affairs required for the norm to be applicable. This may in turn require the legal expert to look for norms in other statutory provisions and so on. The legal expert mentally constructs a “norm graph” that enables him to decide whether the originally intended legal consequence can be reached or not. As depicted in Figure 1, a norm graph consists of legal concepts (represented by nodes) and links between them (represented by arrows).

[1] I Zhang, “Economic consequences of the Sarbanes–Oxley Act of 2002” (2007) 44 *Journal of Accounting and Economics* 74-115.

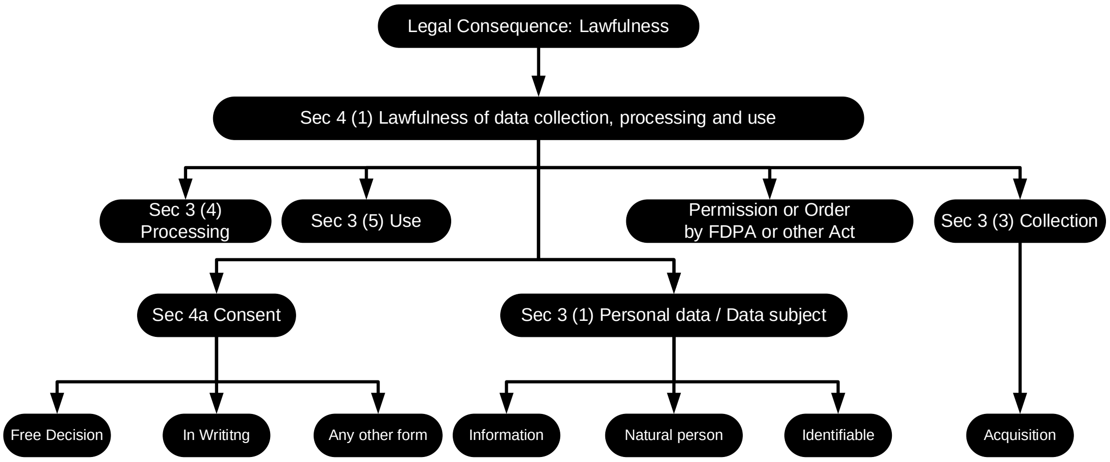

Legal Consequence: Lawfulness
Sec 4 (1) Lawfulness of data collection, processing and use
Sec 3 (4) Processing
Sec 3 (5) Use
Permission or Order by FDPA or other Act
Sec 3 (3) Collection
Sec 4a Consent
Sec 3 (1) Personal data / Data subject
Free Decision
In Writitng
Any other form
Information
Natural person
Identifiable
Acquisition

Figure 1 : Norm graph for the legal consequence of "Lawfulness" in the Sec. 4 (1) FDPA.

We claim that a technology-supported framework must be established to convey and manifest legal requirements in the software as early as possible. Since software developers are usually legal laymen, this paper contributes an automated approach for advising developers about legal reasoning.[2] This support is integrated in the development environment of the software developer.

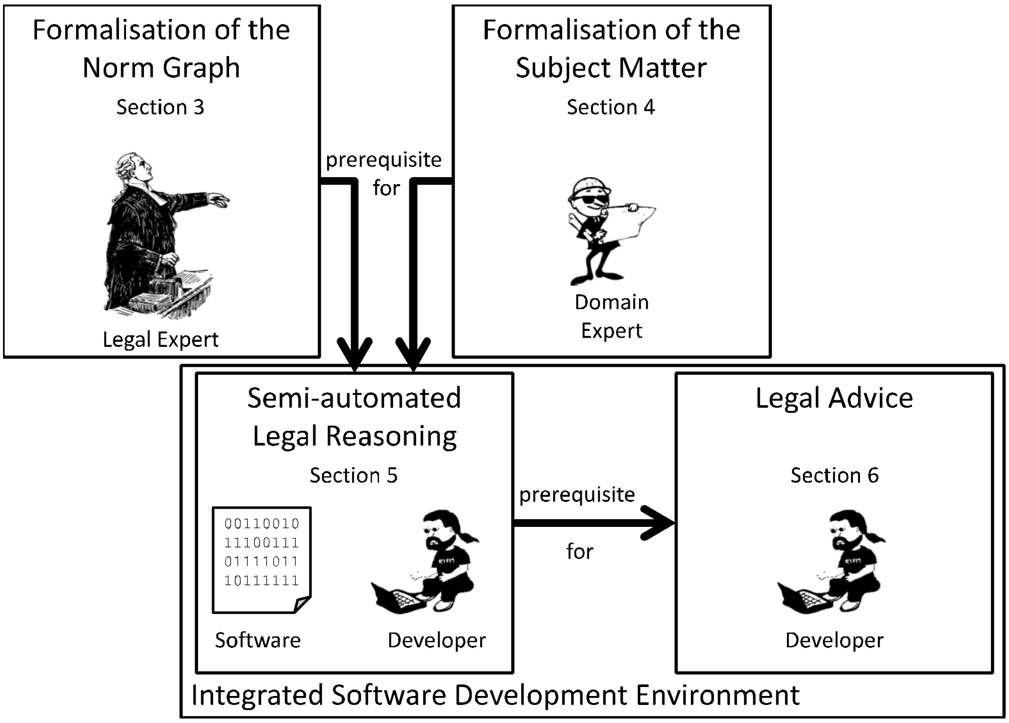


Figure 2 : Overview of our approach and outline of this paper.

[2] There are a number of legal philosophies that try to explicate the legal reasoning process. For the German legal system see, e.g., S Ring, *Computergestützte Rechtsfindungssysteme, Voraussetzungen, Grenzen und Perspektiven,* (Köln, Berlin, Bonn, München: Heymanns, 1994). Our approach is based on K Larenz, *Methodenlehre der Rechtswissenschaft,* 6th ed (Berlin Heidelberg New York: Springer, 1991).

The wizard is based on the formalisation of the norm graph as depicted in Figure 2. A legal expert is required for capturing both the semantics of legal concepts as well as the rule-like nature of legal norms and their relationships. Another prerequisite is the formalisation of the subject matter, namely, software and its environment. This is the task of a domain expert. The semi-automation of the legal reasoning process then happens within an integrated software development environment based on a given software, and requiring the interaction of the developer. Finally, the developer is advised in “translating” and manifesting individual consequences in the software.

2. Motivating Scenario

The following section introduces a motivating scenario as a running example throughout this paper. In this scenario, we present a manual legal reasoning process that reveals violations of the FDPA. Understanding the manual legal reasoning process facilitates the understanding of our automated approach in Sections 3 to 6. Our automated approach avoids such violations during the development phase of the software.

2.1. Setting

For our motivating scenario, we limit ourselves to the German data privacy law. Data privacy law has a small number of legal consequences that can be checked. Note, however, that our approach is independent of a particular field of law.

The motivating scenario concerns the fictitious company, “KnowWhere,” that offers a “person locator app.” This application can be used to track the location of a user who has installed the app on his smartphone. For instance, private customers can use the app to obtain information about the location of their Facebook friends.

As depicted in Figure 3, the person locator app accesses the GPS module of the smartphone and sends the coordinates and a specific Facebook ID to the server application. The server updates the corresponding database entry which also comprises additional information about the person. For obtaining and displaying maps, KnowWhere relies on Google Maps, a service provided by the Google Corporation. Interesting points and locations, defined by GPS coordinates, can be highlighted on a map and marked with the Facebook ID.

Furthermore, KnowWhere offers the “person locator portal” showing maps with the positions of all users that belong to a specified group. The user has to identify himself and specify a group-ID. The server collects all user locations that belong to the given group and uses Google Maps to highlight their positions on the map.

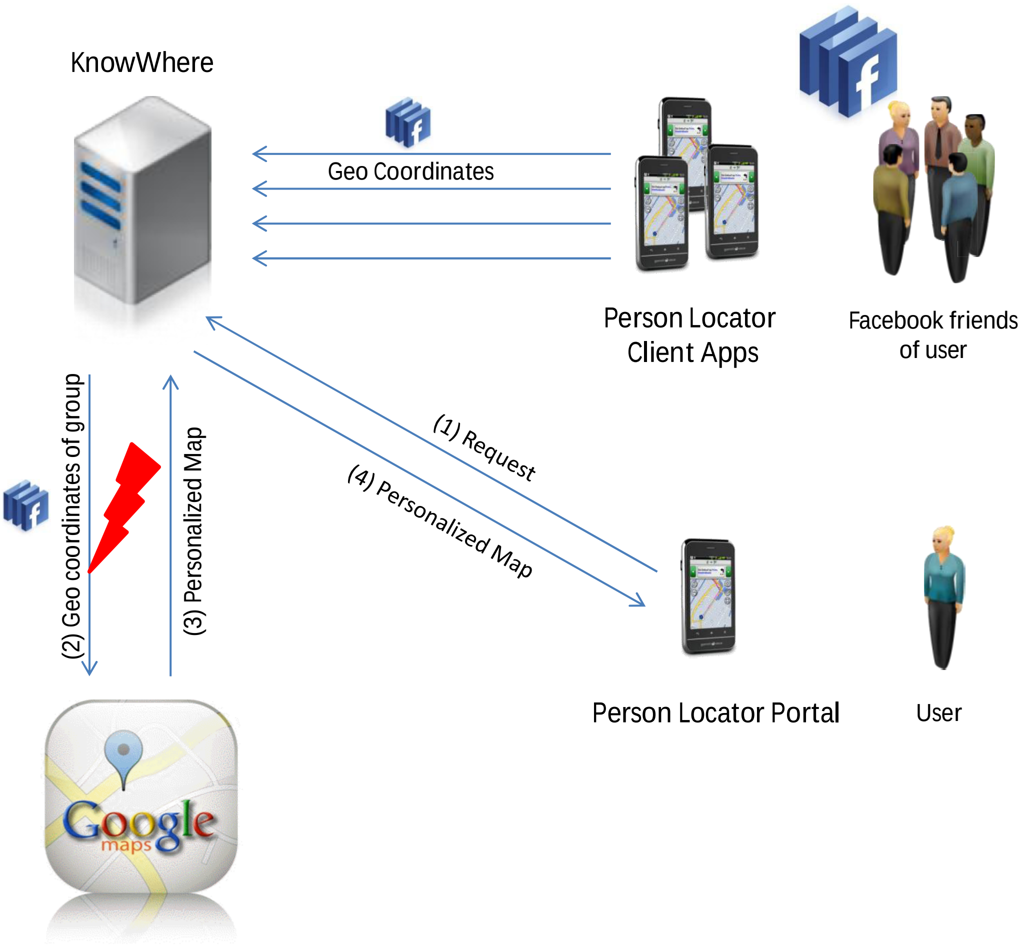


Figure 3 : A violation of the German FDPA.

In the following subsection 2.2, we present a manual legal reasoning process which shows how the software violates the privacy law by using Google Maps to pinpoint the Facebook friends of the user.

2.2. Manual Legal Reasoning

The first step of the manual legal reasoning process is to check whether the FDPA is applicable at all.

Question 1: Which provision is applicable?

The FDPA defines its scope at the beginning in:

> [**Sec 1 (2) FDPA – Purpose and scope**: This Act shall apply to the collection, processing and use of personal data by … private bodies.]

KnowWhere, a "private body" as described by Sec. 2 (4) of the FDPA, discloses Facebook IDs and geo coordinates to other parties, namely, Google Corporation. That would qualify as either a “transfer” of data (according to Sec. 3 (4) No. 3 of the FDPA) or as “use” of data (according to Sec. 3 (5) of the FDPA).

Therefore, it has to be checked whether the data can be classified as *personal*:

> [**Sec. 3 (1) FDPA – Further definitions**: “Personal data” shall mean any information concerning the personal or material circumstances of an identified or identifiable natural person (“data subject”).]

Both the user and his friends are “natural persons.” Facebook IDs can be identifying information, especially if the IDs feature the full name. In most cases, it takes only modest effort to “identify” the person behind the Facebook IDs. The geo coordinates attributed to the IDs provide further information about the “personal circumstances” of the users. Hence, most Facebook IDs and the additional geo data associated with the ID would be classified as “personal data” according to Sec. 3 (1) of the FDPA.

According to Sec. 1 (3) S. 1, the FDPA shall only be applicable if the case at hand is not covered by special legal provisions on data privacy in another legal act. Concerning data handling by private bodies, Sec. 91-107 of the German Telecommunications Act (TCA) and Sec. 11-15a of the German Telemedia Act (TMA) are most relevant.

The TCA contains special data privacy provisions solely for the handling of data by providers of “telecommunication services” as defined in Sec. 3 No. 24. As KnowWhere does not maintain telecommunication infrastructure itself, the TCA is not applicable. Yet, the portal provided by KnowWhere falls within the definition of “Telemedia” in Sec. 1 (1) of the TMA. Whether the data privacy norms in Sec. 11-15a of the TMA overrule the provisions of the FDPA depends on the type of data handled by the telemedia provider. The TMA only covers “inventory data” as defined in Sec. 14 and “usage data” as defined in Sec. 15. The Facebook IDs and the GPS data disclosed by KnowWhere are neither necessary for establishing the contractual relationship with the users, nor for submitting, or invoicing the usage of the portal. KnowWhere does not identify its users by their Facebook IDs, but by their telephone connection. Also, providing GPS data is not necessary to use the personal locator portal as such, but only to enhance its functionalities. Hence, these types of data do not fall under the regime of the TMA.

As a result, the FDPA is applicable.

Question 2: Is the disclosure of user data to Google lawful?

The next step is to check for lawfulness according to:

> [**Sec. 4 (1) FDPA – Lawfulness of data collection, processing and use**: The collection, processing and use of personal data shall be lawful only if permitted or ordered by this Act or other law, or if the data subject provided consent.]

As shown above, the disclosure of Facebook IDs and geo coordinates to Google is either to be qualified as “transfer” that is “processing of data” (Sec. 3 (4) of the

FDPA), or "use of data." Therefore, lawfulness requires either the "permission or order by this Act or other law", or that the "data subject provided consent."

Question 2.1: Is* permission or order by this Act or other law *provided?

Part III of the FDPA contains the provisions applicable for private bodies (compare Sec. 27 of the FDPA). Therefore, a legal norm for permission has to be primarily sought in this part. Upon close inspection we find:

> [**Sec. 28 (1) S. 1 No. 1 FDPA – Collection and recording of data for one's own commercial purposes**: The collection, recording, alteration or transfer of personal data or their use as a means to pursue one's own commercial purposes shall be lawful if necessary to create, perform or terminate a legal obligation or quasi-legal obligation with the data subject, …]

Sec. 28 of the FDPA is applicable only for the handling of data for one's "own commercial purposes." KnowWhere discloses the data to Google in order to be able to provide information about the location of participants and, thus, fulfills the obligation it accepted in the course of providing the app for the users. As this is KnowWhere's "own commercial purpose," Sec. 28 of the FDPA is a suitable permission norm.

Sec. 28 (1) S. 1 No. 1 of the FDPA covers "collection, recording, alteration or transfer" of personal data. According to Sec. 3 (4) No. 3 of the FDPA, disclosure to a "third party" falls within the definition of transfer. According to Sec 3 (8) S. 2, 3 of the FDPA, a third party is any party other than the controller of private data to whom the FDPA is being applied, in this case KnowWhere, (Sec. 3 (7) of the FDPA) but excluding the persons who are the subjects of the data, and also excluding any parties acting "on behalf of" the data controller. The question is whether Google is a third party, or whether it acts on behalf of the data controller, KnowWhere, as defined in Sec. 11 of the FDPA.

Sec. 11 of the FDPA lists a variety of requirements ensuring that the data controller is able to monitor and control every step of data handling. KnowWhere has neither negotiated contractual requirements with Google, nor is it able to control or monitor Google's handling of data. Hence, Google does not handle the data on behalf of KnowWhere. Rather, Google handles the data on its own behalf (Sec. 3 (7) of the FDPA). Therefore, Google is a "third party" as defined in Sec. 3 (4) No. 3 of the FDPA. Therefore, the disclosure of the Facebook IDs and geo coordinates of the user's friends is an act of "transfer."

The next question is whether the transfer is "necessary" for KnowWhere "to create, perform or terminate a (quasi-)legal obligation with the data subjects." As a key function of the person locator app, KnowWhere promises to provide a service that monitors the current location of the user's friends. Even if KnowWhere is not willing to incur contractual obligations, this relationship can at least be qualified as quasi-legal. Thus, the key question is whether the transfer of the Facebook IDs and the geo coordinates to Google is "necessary" to perform the obligation of monitoring the users' locations. This criterion is two-fold: on the one hand, the processing of data as such is only necessary if the contractual performance cannot be delivered without it in an appropriate way. On the other hand, the data controller has to restrict the amount of processed data to the necessary minimum.

KnowWhere is not reliant on the visual interface of Google Maps in order to monitor current locations. Even if it was, it could use the freely accessible Google Maps data and mark the locations by itself. If KnowWhere still wanted to involve Google in the data provision, it would be sufficient to transfer anonymised or aliased data. All in all, the transfer of the Facebook IDs and geo data to Google is not “necessary” in the sense of Sec. 28 (1) S.1 No. 1 of the FDPA. Thus, the data transfer cannot be justified by this provision.

Other statutory provisions that permit or order the transfer are not apparent. In particular, the transfer cannot be legitimated by Sec. 28 (1) S.1 No. 2 FDPA, because there is an overriding interest of the app users in not having data as sensitive as their current location transferred to third parties. Therefore, there is no law “permitting or ordering” the data handling by KnowWhere.

Question 2.2: Has the data subject provided consent?

Proceeding to the second alternative of Sec. 4 (1) of the FDPA, a lawyer would check for “consent” provided by the data subjects. Operating systems generically ask the user during installation of an app for access to the smartphone’s resources such as the GPS module. An affirmative response would count as a declaration of consent. In order to function as effective consent, declarations would have to fulfill the conditions of:

> [**Sec. 4a (1) FDPA – Effective Consent**: Consent shall be effective only when based on the data subject’s free decision. Data subjects shall be informed of the purpose of collection, processing or use and, as necessary in the individual case, or on request, of the consequences of withholding consent. …]

Due to the generic nature of such questions (“May the app use the GPS module?”), the user is not appropriately informed about the purpose of the collection, processing, and use of his personal data. In particular, the users are not informed about the transfer of personal data from KnowWhere to Google. Therefore, effective consent is not given.

Result

As a result, the data transfer from KnowWhere to Google can neither be justified by law nor by consent. Therefore, according to Sec. 4 (1) of the FDPA, the conduct of KnowWhere violates data privacy law.

3. Formalisation of the Norm Graph

The prerequisite for applying our approach of engineering compliant software, and, thus, to prevent violations such as discussed in the previous section, is the formalisation of the norm graph. It is the task of a legal expert to capture both the semantics of legal concepts (Section 3.1) as well as the rule-like nature of legal norms and their relationships (Section 3.2). Both can be captured by means of an ontology.

An ontology is a conceptual, i.e., user-targeted, data model that relies on formal logic.[3]

In the following section we sketch how a "statutory ontology" is built. According to our motivating scenario, we build a "data privacy ontology for private bodies" which requires the formalisation of the German FDPA Part III as well as parts of the more special German Telemedia (Sec. 1, 11-15a) and Telecommunications Acts (Part VII, Chapter 2).

3.1. Formalisation of Legal Concepts

Automatic preprocessing (3.1.1) supports the legal expert in extracting the required vocabulary out of a legal text, e.g., the German FDPA, as a starting point for formalising legal concepts. The vocabulary forms the basis of a lexicon which complements extracted terms by additional information (3.1.2). In turn, the lexicon serves as a basis for creating classes and relations of the ontology (3.1.3). Legal concepts are eventually represented by a class, as well as relations to other classes, that capture the legal concept's definition.

3.1.1. Automatic Preprocessing

Automatic preprocessing aims at decreasing the word pool of a legal text to the necessary minimum. The input is represented by a legal text in electronic form, such as the FDPA on the Web, and the output is a list of candidate words for formalisation of legal concepts, e.g., "personal data." Automatic preprocessing applies

- Filtering: The word pool of a legal text is reduced to words with relevant meaning. Stop-words,[4] such as "and," are eliminated and stemming[5] reduces declined verbs to their corresponding noun. For example, "processes" and "processed" are reduced to the noun "processing."
- Identification of Relationships: Words that frequently occur together and have a semantic relationship are identified.[6] For example, the words "personal" and "data" frequently occur together and, thus, are likely to be considered as one concept: "personal data."

3.1.2. Creating the Lexicon

While the pre-processing is done automatically, creating the lexicon is a manual endeavour. The purpose of building a lexicon is to enrich the extracted terms with

[3] N Guarino, D Oberle and S Staab, "What is an Ontology?" in S Staab and R Studer (eds), *Handbook on Ontologies*, 3rd ed. (Berlin, Heidelberg: Springer, 2009) 1-17.

[4] H Luhn, "The Automatic Creation of Literature Abstracts" (1958) 2(2) *IBM Journal of Research and Development* 159–165.

[5] M Porter, "An Algorithm for Suffix Stripping" (1980) 14(3) *Program: electronic library and information systems* 130–137.

[6] H Ahonen-Myka, "Discovery of Frequent Word Sequences in Text" in D Hand, N Adams and R Bolton (eds), *Pattern Detection and Discovery* (Berlin, Heidelberg: Springer, 2002) 319–328.

additional information required for building the ontology in the subsequent step.[7] Besides the term itself, the legal expert is prompted to fill in lexicon entries with fields such as "definition," "relations to other terms," "classification," "commentary," "layman explanation" and "questions" or links to "external sources," etc. An example entry for the legal concept "personal data" is shown in Figure 5 (left side).

3.1.3. Classes and Relations

Each lexicon entry represents a legal concept that now has to be formalised by a "class." A class in an ontology represents all instances with common characteristics. For example, the class `Physical Object` represents all instances that are tangible.[8] Common characteristics are expressed by "relations" to other classes. For example, a `Physical Object` might feature a relation `hasWeight,` which points to a `Physical Quality` such as `Kilograms`. The super-/subclass relation (also called "specialisation") holds a special place in an ontology since it puts classes into a "taxonomy." For instance, the class `Natural Person` is a subclass of `Physical Object`, because the first carries all characteristics of the latter and adds special characteristics, such as `hasAge`.

In a first step, the legal expert has to put classes into a taxonomy. In order to support this intellectual process, a foundational ontology can be used as a sound starting point by prescribing basic classes of human cognition.[9] First, the class `Object` denotes all instances that are always present with all their essential parts at a given time, e.g., a `Natural Person`. Second, characteristics, such as the `Age` of a person, are specialised from the class `Quality`. Third, `Processes` are instances that have temporal parts, e.g., the `Processing` of data. Last, `Abstract` denotes instances that do not have extension in space and time, e.g., the number `5`.

By specialising legal concepts, i.e., their corresponding classes, from the correct class of the foundational ontology, the intended meaning can be captured. For example, the concept "collection" can refer both to an `Object` (a set of items) and an `Event` (the intended meaning in the case of the FDPA). The resulting specialisation is shown in Figure 4.

[7] Our approach for building a lexicon as intermediate step for an ontology rests on K Breitman and J Sampaio do Prado Leite, "Lexicon Based Ontology Construction" in C Lucenaet al (eds), *Software Engineering for Multi-Agent Systems II* (Berlin, Heidelberg: Springer, 2004) 19–34.

[8] Classes and relations are written in `fix-width font` in the following.

[9] Our approach rests on the foundational ontology called DOLCE (see A Gangemi et al "Sweetening Ontologies with DOLCE" in A Gómez-Pérez and V Benjamins (eds), *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings* (Berlin, Heidelberg: Springer, 2002) 166-181.

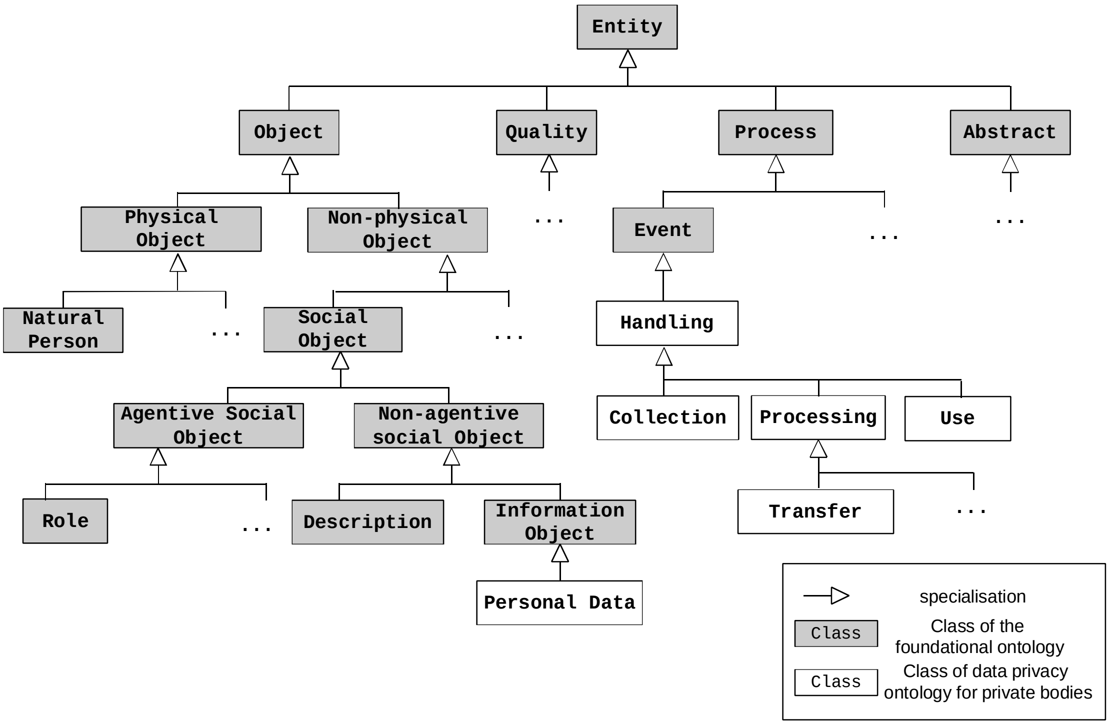
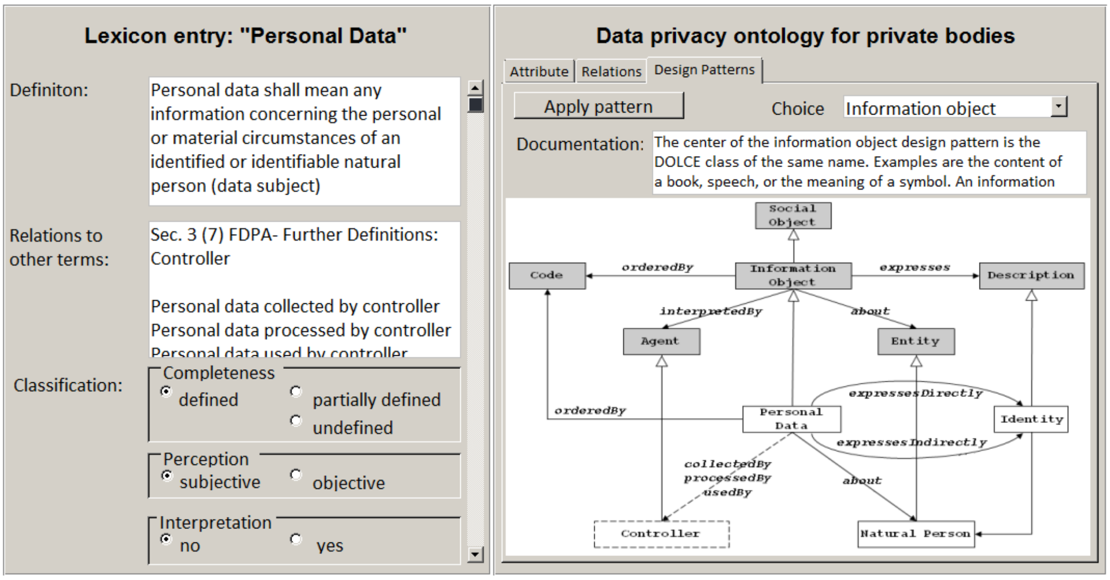


Figure 4 : Specialising classes from the foundational ontology.

After creating the taxonomy, the legal expert has to establish relations between classes. Hints for relations are contained in the lexicon entries "definition" and "relations to other terms." Similar to prescribing a basic set of classes, the foundational ontology also prescribes generic relations, such as `partOf`, which can be used as a starting point.

As with all other steps, creating relations is supported by an intuitive graphical user interface especially designed for the legal expert who usually is no expert in ontology engineering. Figure 5 shows the design for a user interface where the left side depicts the current lexicon entry of the term to be formalised ("personal data"). The right side allows for choosing from a catalogue of established ontology design patterns. In this particular case, the Information Object design pattern can be instantiated to capture the required relations for `Personal Data`.[10]

[10] Ontology design patterns are templates for reoccurring modelling needs (see V Presutti and A Gangemi, *Content Ontology Design Patterns as Practical Building Blocks for Web Ontologies* in Q Li et al (eds) *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings* (Berlin, Heidelberg: Springer, 2008) 128-141 and A Nitin et al, "Information Object Design Pattern for Modeling Domain Specific Knowledge" in *Proceedings of the 1st ECOOP Workshop on Domain-Specific Program Development (DSPD),* (2006)).



Lexicon entry: "Personal Data"

Definiton: Personal data shall mean any information concerning the personal or material circumstances of an identified or identifiable natural person (data subject)

Relations to other terms: Sec. 3 (7) FDPA- Further Definitions: Controller

Personal data collected by controller
Personal data processed by controller
Personal data used by controller

Classification:
Completeness ● defined ○ partially defined ○ undefined
Perception ● subjective ○ objective
Interpretation ● no ○ yes

Data privacy ontology for private bodies

Choice: Information object

Documentation: The center of the information object design pattern is the DOLCE class of the same name. Examples are the content of a book, speech, or the meaning of a symbol. An information

Figure 5 : User interface design for capturing relations.

3.2. Legal Norms

Capturing legal concepts in a statutory ontology is also the prerequisite for capturing the rule-like structure of legal norms, and thus, the norm graph. Since ontology languages blend conceptual modeling capabilities with the possibility for expressing logical rules, both formalised legal concepts and formalised legal norms can be captured in the same formalism.[11]

3.2.1. Individual Norms

Typically, legal norms determine a "legal consequence" (LC), given one or more "state of affairs" ("SF"), which fall within the scope of the norm. Schematically, this can be expressed in an ontology language as a logical "rule" as follows:

$$SF \rightarrow LC \qquad (1)$$

This is to be read as: "when state of affairs (SF) is given, then the legal consequence (LC) applies." Both SF and LC usually consist of several parts, i.e., there are several states of affairs required to derive one or more legal consequences:

$$SF_1, SF_2, \dots SF_n \;\rightarrow\; LC_1, LC_2, \dots LC_m \qquad (2)$$

[11] In our case, we apply the logic programming dialect called F-Logic, see M Kifer, G Lausen and J Wu, "Logical Foundations of Object-Oriented and Frame-Based Languages" (1995) 42(4) *Journal of the ACM* 741-843.

However, for computational reasons, ontology languages constrain the structure of such logical rules to one consequence only. Thus, the first intellectual task of the legal expert is to decompose each norm into the following scheme:

$$\begin{array}{l} SF_1, SF_2, \ldots SF_n \rightarrow LC_1 \\ SF_1, SF_2, \ldots SF_n \rightarrow LC_2 \\ \ldots \\ SF_1, SF_2, \ldots SF_n \rightarrow LC_m \end{array} \qquad (3)$$

In the example of Sec. 4 (1) of the FDPA (see Section 2.2), the word “only” indicates that there are indeed two LCs that must be handled and formalised individually:

- The collection, processing and use of personal data is *lawful* if permitted or ordered by this Act or other law, or the data subject consented; and
- The collection, processing and use of personal data is *unlawful* if neither ordered by this Act or other law, nor prescribed by this Act, nor consented to by the data subject

The next intellectual task for the legal expert is to replace each SF and LC by elements of the ontology. In a first step, this is achieved by identifying relevant classes. In the example, the SFs are replaced by the following classes: `Collection`, `Processing`, `Use`, `Personal Data`, `Permission`, `Order`, `Data Subject`, and `Consent`. LC is replaced by `Lawfulness`.

In a second step, explicit links between the chosen classes are inserted by means of relations. The legal norm typically contains indications for such explicit links, e.g., Sec. 4 (1) of the FDPA contains the phrase “use of personal data,” which requires the insertion of a `performedUpon` relation between `Use` and `Personal Data`.

In a third step, the legal expert checks for implicit links which are not directly mentioned in the legal norm but are mentally complemented by the interpreter. For example, there exists an implicit link between `Consent` and the `Collection`, `Processing` and `Use`. Namely, it is the consent that `permits` such actions.

Finally, the inserted elements of the ontology are logically combined (in formulae 1-3 this is represented schematically by the “,”). Ontology languages offer bracketing as well as logical operators (`AND`, `OR`) for this purpose. As an example, Sec. 4 (1) of the FDPA entailing lawfulness is formalised as follows

```
    ((Collection(X) OR Processing(X) OR Use(X))
   AND performedUpon(X,Y) AND Personal Data(Y))
                                             AND
(Permission(P) OR Order(P)) AND givenFor(P,X)))
                                              OR        (4)
  (Consent(C) AND Data Subject(D) AND about(Y,D)
    AND gives(D,C) AND permits(C,X))
                                   →
     Lawfulness(A) AND givenFor(A,X)
```

X, Y, P, C, D, and A are “variables” that stand for instances of the corresponding class or relation. Their names can be chosen arbitrarily. An expression such as `Data Subject(D)` can be read as a sentence with an unknown part labeled as D: “D is a data subject.” In turn, D can be bound to a concrete instance, e.g., to the user Daniel.

3.2.2. Integration of Several Norms

Legal norms have complex interrelationships that feed back into the formalisation of an individual norm. First, two or more legal norms might have overlapping *SFs* that lead to different *LC*s depending on whether they are given or not. An example for this case has been discussed at the beginning of Section 3.2.1. This case can be solved by negating the overlapping *SFs* via the `NOT` operator as follows:

$$\begin{aligned} &SF_1, SF_2, \ldots \quad (\ldots\ SF_n) \rightarrow LC_1 \\ &SF_1, SF_2, \ldots NOT\ (\ldots SF_n) \rightarrow LC_2 \end{aligned} \tag{5}$$

Second, a legal norm might be an explicit exception to another legal norm. In this case, both norms must be rewritten as shown in formula (6). The occurrence of *SF* in both norms indicates that overlapping states of affairs might be given at the same time.

$$\begin{aligned} SF_1, SF_2, \ldots\ OR\ \ SF_{Exception} &\rightarrow LC \\ SF_1, SF_3, \ldots &\rightarrow SF_{Exception} \end{aligned} \tag{6}$$

As an example, consider Sec. 4a (1) of the FDPA, which requires consent in written form unless there is an exception. One exception is mentioned in the same norm (special circumstances). However, there might also be exceptions in other provisions, e.g., Sec. 13 (2) of the TMA, which lists requirements for the use of an electronic form in the case of Telemedia services. This situation is formalised as sketched in formulae (7) – (9).

> [**Sec 4a (1) FDPA – Effective Consent**: … Consent shall be given in writing unless special circumstances warrant any other form. …]

```
(Consent(C) AND givenIn(C,F)
AND WrittenForm(F)) OR Exception(F)
AND ...              →                   (7)
Effectiveness(E) AND givenFor(E,C)
```

```
SpecialCircumstances(SC) AND
warrants(SC,F) AND OtherForm(F)           (8)
                    →  Exception(F)
```

```
ElectronicForm(F) AND Provider(P)
AND assures(P,U) AND ...                  (9)
                    →  Exception(F)
```

The third case is an implicit exception between legal norms with contradicting consequences, e.g., when one norm takes precedence over the other due to *lex specialis derogat legi generali*. When just considering the text, this leads to logical rules that both overlap in their *SFs* and lead to conflicting LCs:

$$\begin{aligned} SF_1, SF_2, \ldots &\rightarrow LC \qquad (10)\\ SF_1, SF_3, \ldots &\rightarrow NOT\ LC \end{aligned}$$

A possible solution for resolving the conflict is to rewrite both rules according to the scheme depicted in (6), i.e., as an explicit exception.

4. Formalisation of the Subject Matter

The subject matter has to be formalised as completely as possible for automated processing. This is also a prerequisite before being able to advise the developer in a software development environment. Fortunately, in our case, a major part of the subject matter is already inherent in the software. It is more a matter of representing this information suitably, such that an automated processing is facilitated.

The following section presents a suitable way for formalisation by means of a "subject matter ontology." Here it is important to point out that an ontology consists of a schema and instances. The schema (further discussed in Section 4.1) formalises relevant classes, such as "software" or "data," and puts them in relation. The schema has to be manually designed only once by a domain expert. Instances of such classes then represent a specific subject matter (further discussed in Section 4.2). These instances can be obtained automatically from the software.

Using the same ontology language and also the same foundational ontology facilitates finding correlations with the statutory ontology during the legal reasoning process in Section 5.

4.1. Subject Matter Schema

It is the task of a domain expert to build the schema, i.e., to identify relevant classes and relations to represent common concepts in the realm of software.[12] Therefore, the schema is independent of a *specific* subject matter such as the one outlined in the scenario. Figure 6 shows a small excerpt of the ontology's major classes, e.g., `Data`, `Software`, or `Invocation`. They are aligned under, i.e. specialised from, the (grey) classes of a foundational ontology.

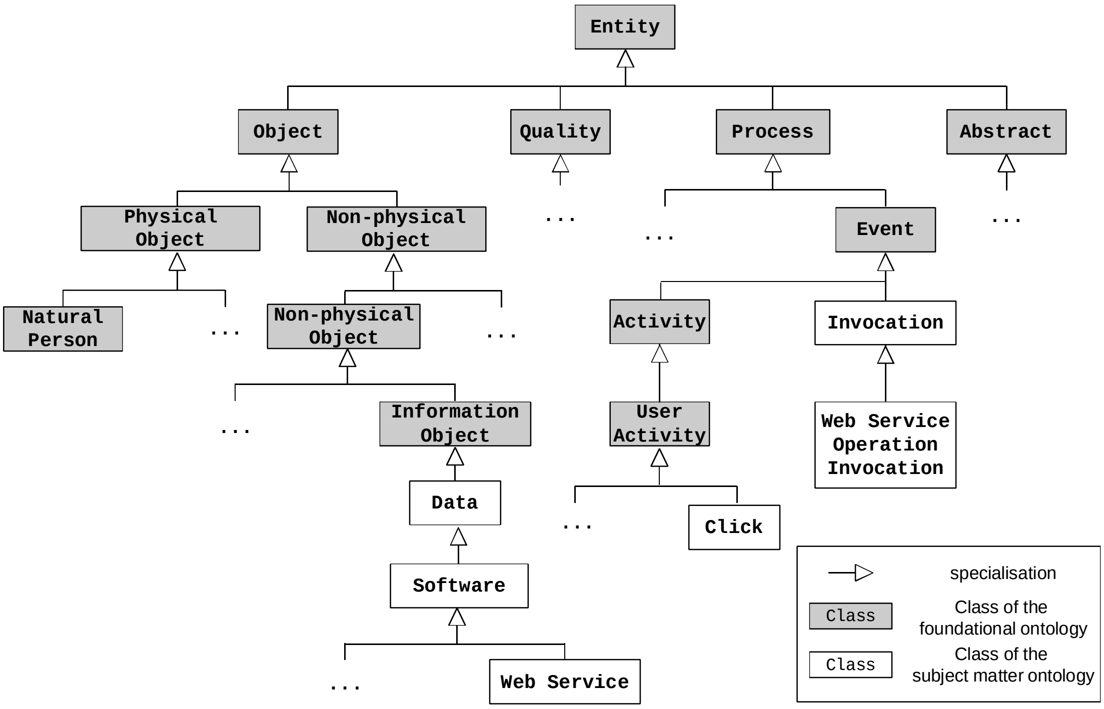


Figure 6 : Snippet of the subject matter ontology's schema.

Figure 6 only shows super-/subclass relations (specialisation). However, each class can have an arbitrary number of relations with different semantics. It is an intellectual and manual task of the domain expert to identify and model such relevant relations. Figure 7 shows some examples for arbitrary relations around the class `Web Service`. A Web service is an application programming interface whose functionality can be utilised ("invoked") by using Web protocols, such as HTTP. A piece of `Software`, e.g., a software component, `performs` a `Web Service Operation Invocation`. Each invocation `targets` a `Web Service Operation`, which in turn is `partOf` a `Web Service`. Invocations `request` and `respond` with `Data` that is `representedBy` an `XML` document and about some `Entity`.

[12] The Ontology of UIs and Interactions (see H Paulheim, *Ontology-based Application Integration* (Berlin, Heidelberg: Springer), at ch 6) is reused for this purpose which builds on the Core Software Ontology (see D Oberle et al, "Towards Ontologies for Formalizing Modularization and Communication in large Software Systems" (2006) 1(2) *Applied Ontology* 163-202. Both ontologies are based on the foundational ontology DOLCE.

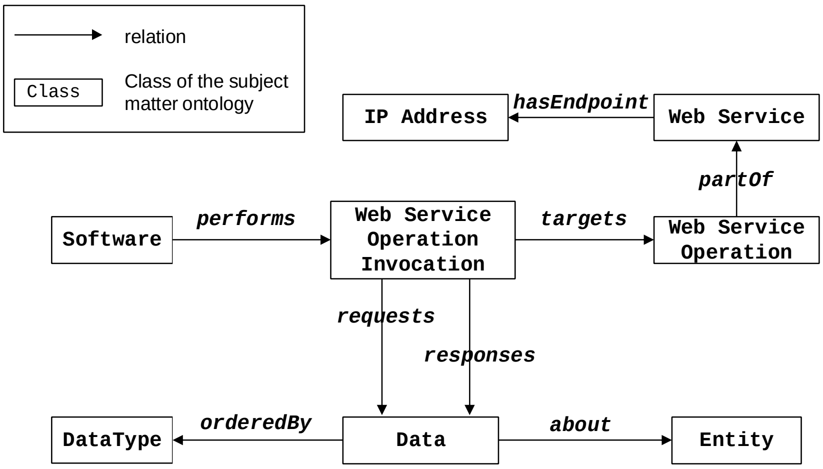


Figure 7 : Representing invocations in the subject matter ontology.

4.2. Subject Matter Instances

While the schema of the subject matter ontology only captures relevant concepts in form of classes and relations, it is the instances that represent a *specific* subject matter depending on a given software. That means the schema does not change but rather the instances depend on the actual subject matter. Instances are concrete elements of classes that have relations to other instances according to the schema. For example, an instance `WSOpI`$_1$ of class `Web Service Operation Invocation` might represent the transfer data to Google Maps in our scenario. In contrast to the schema, the instances are obtained automatically in different ways, as discussed in the following section.

4.2.1. Automatic Extraction of Instances

Most of the relevant instances are inherent in the code or accompanying descriptors and are extracted automatically. Concrete instances for automatic extraction from the software in the subject matter ontology have multiple sources. One primary source is integrated software development environments that keep track of software components and their interdependencies. In addition, software visualisation tools[13] perform a more in-depth analysis of invocations for better understanding complex software. This information is leveraged for our purposes. Another source is descriptor files, typically given in XML. For instance, WSDL (Web Service Description Language) files contain information about a Web service, its operations and endpoints as well as requests and responses.

For each source, an adaptor has to be developed that realises a mapping from the source information to the subject matter ontology. Adaptors have to be designed such that they only extract *relevant* information. For example, the code of the person locator app features a programming variable, called `FBID`, of type `xsd:FacebookAccount`, which contains the Facebook ID at runtime.

[13] S Bassil and R Keller, “Software Visualization Tools: Survey and Analysis” in *9th International Workshop on Program Comprehension (IWPC 2001), 12-13 May 2001, Toronto, Canada* (IEEE Computer Society, 2001) 7-17.

4.2.2. Representing Unknown Instances

Not all information can be extracted and is actually known during the design time of the software. In this case, proxy instances are put in place that have to be resolved by the developer during the legal reasoning process (see Section 5).

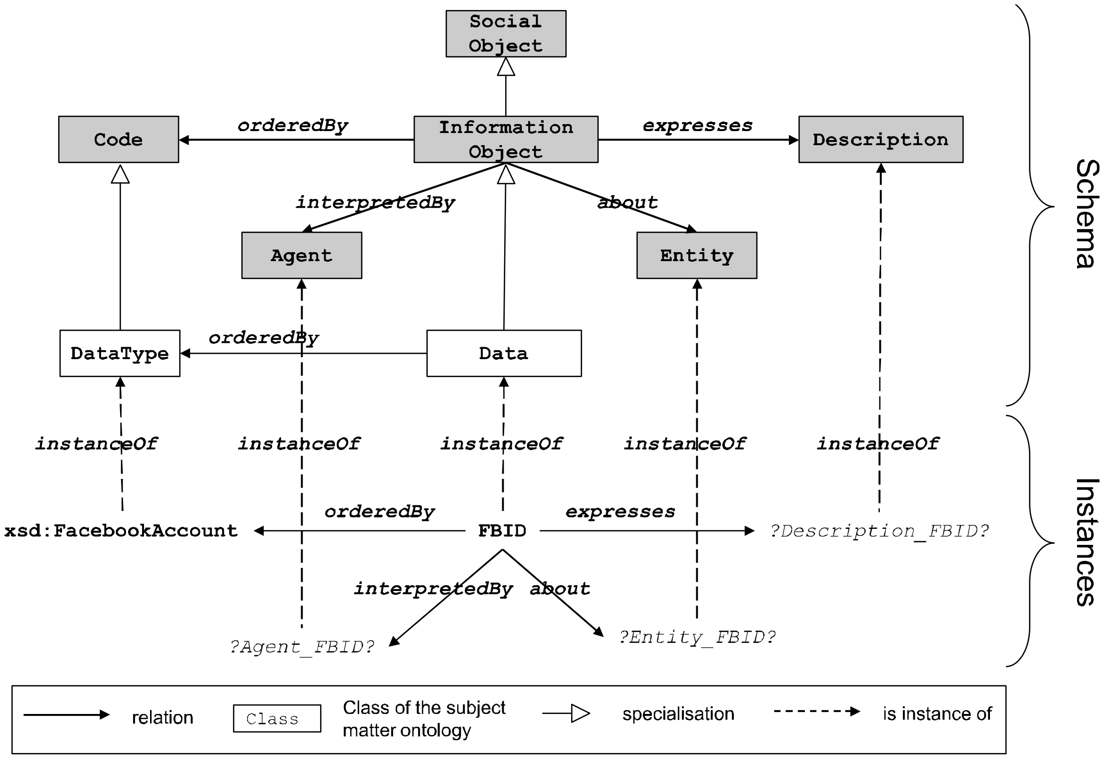


Figure 8: Representation of the variable `FBID` with unknown information.

An example is depicted in Figure 8. We can see that `Data` is a specialisation of `Information Object` in the foundational ontology and the variable `FBID` is an instance of `Data`. The class `Information Object` is associated with an ontology design pattern of the same name, prescribing the depicted relations to the remaining (grey) classes in the figure. An automatic extraction from the code can only yield `FBID`'s `DataType`, however. For all remaining relations, proxy instances have to be put in place (identified by a common naming scheme: `?<class>_<instance>?`).

4.2.3. Representing Default Knowledge as Instances

Some information is known in advance depending on the context of our approach. This information is exploited and represented as instances in the subject matter ontology. Simple examples are the user and provider of the software. Both roles are always present and result in default instances `!User!` (of class `Natural Person`) and `!Provider!` (of class `Organisation`), respectively.

A more sophisticated example is the software context. In our scenario, we deal with mobile apps and corresponding operating systems. Such operating systems typically prompt the user to “accept” certain permissions before installing an app. For example, the user is asked whether the app is allowed to transmit the coarse- (network based) or fine-grained (GPS) location of the device. The click on the “accept and download” button, therefore, is a default that happens before using the app. A corresponding default instance `!Click_Accept!` (of class `Click`) is added to the subject matter ontology, correspondingly.

5. Semi-automated Legal Reasoning

Formalising the norm graph, as well as the subject matter, is the prerequisite for (semi-)automating the legal reasoning process. The semi-automation of the legal reasoning process happens within an integrated software development environment based on a given software. This process essentially finds correspondences between the subject matter and one or more statutory ontologies and requires the interaction with the developer.

Information of the software to be analysed is represented by a set I of instances in the subject matter ontology. These instances are automatically extracted from the source code as described in Section 4.2. Semi-automated legal reasoning transfers the instances to a statutory ontology, i.e., subsumes them under corresponding classes. For example, the instance $\texttt{WSOpI}_1$ of class `Web Service Operation Invocation` in the subject matter ontology corresponds to the class `Transfer` in the data privacy ontology for private bodies. In this case, $\texttt{WSOpI}_1$ must be subsumed under, i.e. become an instance of, `Transfer` as well.

In the following we present a subsumption algorithm that relieves the developer as much as possible from finding such correspondences. The algorithm starts with the question for applicability of a specific statutory provision (see Question 1 in Section 2.2). This question is reduced to a selection box where the developer can choose a statutory ontology for a specific field of law, e.g., the data privacy ontology for private bodies, or for public bodies, etc.

Depending on the chosen field of law, the developer is prompted to select one, several, or all desired legal consequences. In the data privacy ontology for private bodies, legal consequences are *lawfulness* as well as the obligations of the data controller defined by the FDPA. Each legal consequence is associated with a norm graph, e.g., the norm graph for lawfulness sketched in Figure 1. Consequently, the algorithm tries to subsume the subject matter under the nodes of the chosen norm graph.

The five process steps in Figure 9 (each of them further explained in Sections 5.1 to 5.5) have to be executed for each legal concept `LC` at the bottom of the norm graph, i.e., ones which are not further decomposed. Whether legal concepts above `LC` are given can be automatically inferred (further discussed in Section 6).

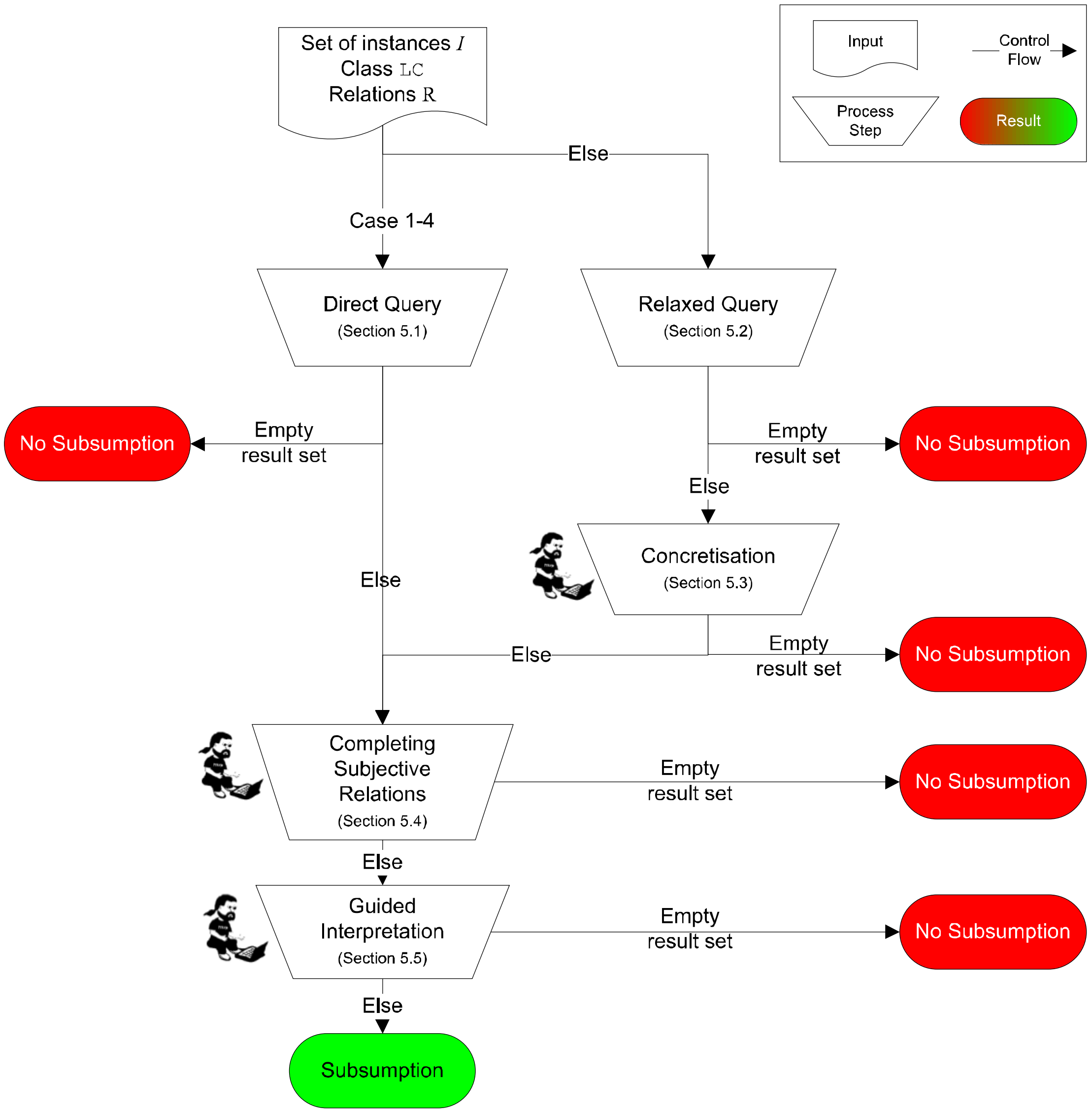


Figure 9: Subsumption algorithm. The steps of *5.3 Concretisation, 5.4. Completing Subjective Relations* and *5.5. Guided Interpretation* require the interaction of the developer.

Besides *I* and `LC`, another input to the algorithm is `LC`'s set of relations R. For Personal Data, R consists of the following relations and accompanying target classes: `expressesDirectly Identity`, `expressesIndirectly Identity`, `about Natural Person` (according to the right side of Figure 5).

In each process step of the algorithm, the set *I* of all instances of the subject matter ontology is reduced to a result set containing potential candidates for subsumption under the current legal concept `LC`. As soon as the result set is empty, the algorithm can terminate since no subsumable instance exists. If all steps are executed and *I* remains not empty, its instances are transferred to the corresponding statutory ontology.

5.1. Direct Query

The first step tries to query directly for "*all instances of class X in the subject matter ontology*" where X depends on the following cases:

Case 1 – Preliminary Check: Our approach allows to establish general correspondences between a class in a statutory ontology and a class `C` in the subject matter ontology. For instance, `C` might be the class `User` or `Provider` whose instances always correspond to `Data Subject` and `Data Controller` in the data privacy ontology for private bodies. If such a correspondence is present for the current legal concept `LC`, the subject matter ontology can be queried directly for instances of `C(`i.e., `X` = `C)`.

Case 2 – `LC` is part of the common foundational ontology: There might be general concepts that are contained in the foundational ontology, and, thus, are valid in both a statutory and the subject matter ontology. An example for such a concept is `Natural Person` (see Figure 4 and Figure 6). In this case, the subject matter ontology can be directly queried for the legal concept, i.e., `X` = `LC`.

Case 3 – `LC` is part of the subject matter ontology: A legal concept might be a terminus technicus. For example, the Telemedia Act might talk of "IP addresses" which is also a class in the subject matter ontology. In this case, the subject matter ontology can be directly queried for the legal concept, i.e., `X` = `LC`.

Case 4 - `LC` features relations that are contained in the subject matter ontology: Although `LC` might not be part of the subject matter ontology, all the relations of `LC` might point to classes contained in the subject matter ontology. In this case, the query is constructed out of all instances that feature the corresponding relations independent of a specific class.

If none of the cases applies, the algorithm continues with the step *5.2. Relaxed Query*.

In case the query yields an empty result set, the algorithm can be terminated for the current legal concept `LC`, i.e., no subsumption is possible. Otherwise the algorithm continues with steps *5.4. Completing Subjective Relations* and *5.5. Guided Interpretation*, respectively.

5.2. Relaxed Query

As soon as at least one relation of the current legal concept `LC` is neither part of the foundational nor the subject matter ontology, a direct query is not possible. Instead, the query constructed in 5.1 must be relaxed, i.e., classes and relations are generalised to their respective superclasses and –relations in the foundational ontology.

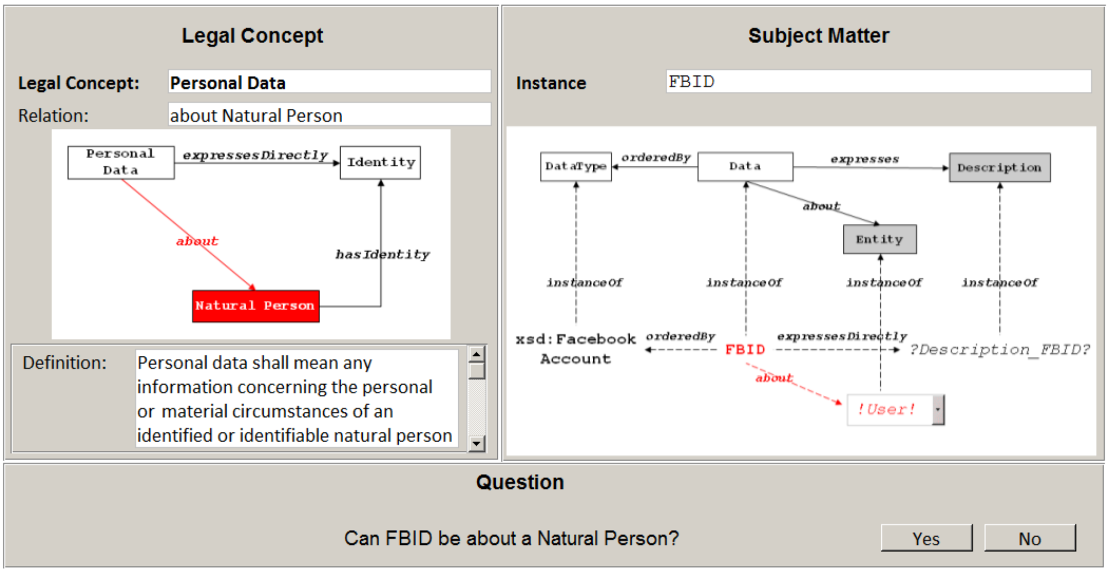
In our running example, `Personal Data` is relaxed to its superclass `Information Object` (see Figure 5). Its relation `about Natural Person` is relaxed to `about Entity`, correspondingly. A query for "all instances of `Information Object` that are `about` an `Entity`" yields a result set that contains `FBID` of Figure 8, since it is an `instanceOf Data` which is also an `Information Object`.

Because of the generalisation, a non-empty query result might contain instances that do not classify as personal data. These must be manually selected by the user in the subsequent step *5.3 Concretisation*. If the query result is empty, there are no subsumable instances and the algorithm can be terminated for the current legal concept `LC`.

5.3 Concretisation

This step iterates over all generalised relations and prompts the developer to concretise each relation for every instance in the result set. This is supported by a wizard as depicted in Figure 10 where the left side represents the legal concept and current relation and the right side represents the current instance.

In the example, the `about Natural Person` relation of `Personal Data` has been relaxed to `about Entity`. The corresponding query in the previous step yielded the instance `FBID` as a result. The context of `FBID` can be seen in the right side of Figure 10. Consequently, the wizard asks the developer whether `FBID` might be `about` a `Natural Person` and even lets him choose concrete instances of `Natural Person` (the default instance `!User!` in this particular case). The developer answers the question with “yes” since he is aware that `FBID` is a programming variable that represents a Facebook ID at runtime.



Legal Concept

Legal Concept: Personal Data

Relation: about Natural Person

Definition: Personal data shall mean any information concerning the personal or material circumstances of an identified or identifiable natural person

Subject Matter

Instance FBID

Question

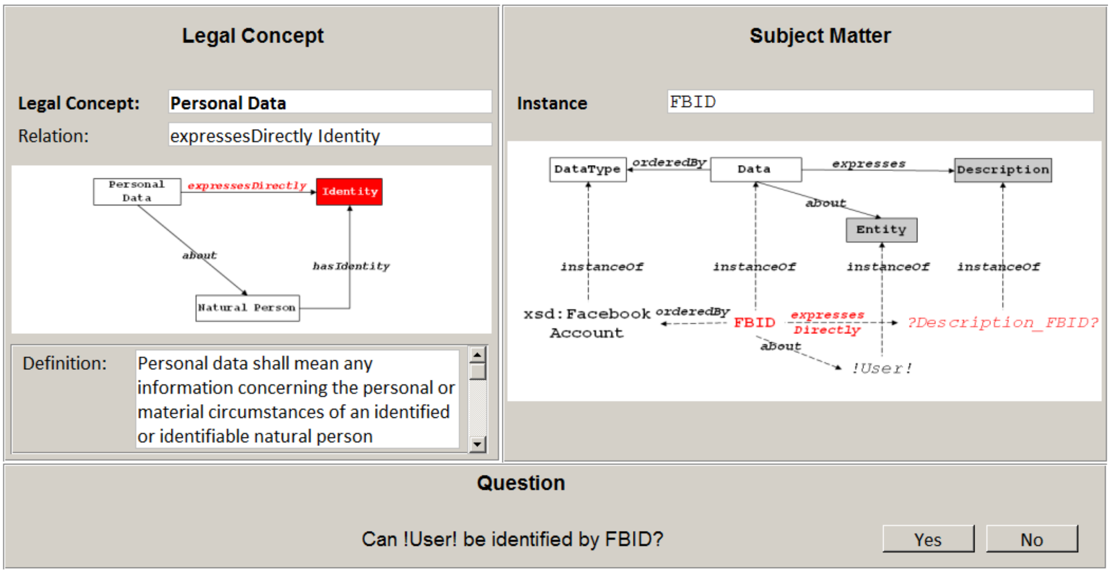
Can FBID be about a Natural Person?

Figure 10: User interface design for step concretisation.

If this step ends with an empty result of instances, the algorithm can be terminated for the current legal concept `LC`. Otherwise, the algorithm proceeds with completing subjective relations in the subsequent steps.

5.4. Completing Subjective Relations

Subjective relations rely on perception and require information about subjective or personal attitudes which are not accessible to another person. In our approach, legal concepts which cannot be formalised, such as whether a person is identified by a given data, are also treated as subjective.



Legal Concept

Legal Concept: Personal Data

Relation: expressesDirectly Identity

Definition: Personal data shall mean any information concerning the personal or material circumstances of an identified or identifiable natural person

Subject Matter

Instance FBID

Question

Can !User! be identified by FBID?

Figure 11: User interface design for step completing subjective relations.

Accordingly, this step iterates over all subjective relations and all instances of the result set. The step requires the interaction of the developer via a wizard (see Figure 11). The left side of the wizard displays a graphical rendition of the current legal concept `LC` where the subjective relation is highlighted with accompanying explanations. As an example, the relation `expressesDirectly Identity` of `Personal Data` is classified as being subjective in the scenario. The right side shows an instance of the current result set, e.g., the programming variable `FBID` introduced in Section 4.2.2. The developer has to decide if the subjective relation is given for every instance on the basis of the provided information. In the example, `FBID` represents the Facebook ID and the developer knows that such an ID often reflects the name of a natural person. Consequently, the developer positively answers the corresponding question of the wizard.

In case of an empty result set after the iteration, the algorithm can be terminated for the current legal concept `LC`. Otherwise, the algorithm proceeds with step *5.5. Guided Interpretation*.

5.5. Guided Interpretation

When legal experts interpret a statute, their approach is usually eclectic: they first try to interpret the *statutory text* of the relevant legal concept and norm, respectively. If this interpretation yields ambiguities or uncertainties, the legislative context, history, and purpose have to be consulted for a resolution.[14] The *legislative context* considers

[14] Statutory interpretation in Civil Law jurisdictions, e.g., Germany, follows a defined procedure, see K Larenz, *Methodenlehre der Rechtswissenschaft*, 6th ed (Berlin Heidelberg New York: Springer, 1991) at 313. An explanation for the less strict but comparable procedure in Common Law jurisdictions can be found in W Eskridge and P Frickey, "Statutory Interpretation as Practical Reasoning" (1990) 42 *Stanford Law Review* 321-384.

associated legal concepts and norms such that the interpretation is coherent altogether. The *legislative history* considers interpretations of the evolution of the statutory text over time and its original intention. Finally, the *legislative purpose* considers the intended overall legal policy of the corresponding provision.

The steps so far dealt with relations that can be interpreted by considering the statutory text only. In the following, we present a wizard that guides the developer in dealing with ambiguous or uncertain relations that require additional interpretation. An automated processing of such relations is not possible since the required material for further statutory interpretation (legislative context, history, and purpose) is not available in machine understandable form.

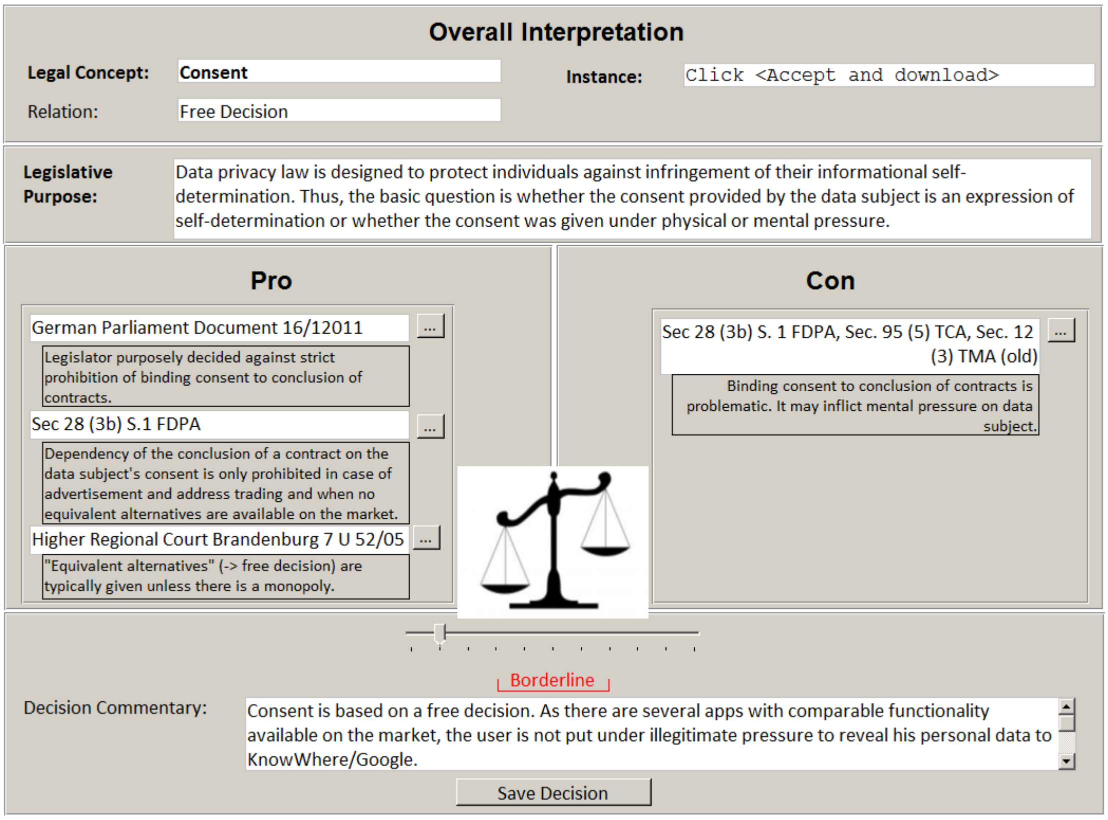
Suppose the current legal concept `LC` is `Consent`. Then, an example for a relation that requires additional interpretation is `basedOn FreeDecision`. Although mentioned in Sec. 4a (1) of the FDPA, *free decision* is not further defined in the FDPA leaving room for a very broad interpretation.

> [**Sec 4a (1) FDPA – Effective Consent**: Consent shall be effective only when based on the data subject's free decision. …]

The wizard first presents material and sources for the legislative purpose, history and context of the legal concept or norm. Such material and sources have been provided by the legal expert when building the lexicon (see Section 3.1.2). Consequently, the developer selects material and sources that speak in favor of (pro) or against (con) a subsumption via checkboxes.

Suppose the current instance represents a click on "accept and download" - a consent required by the operating system of the smart phone before installing and using the person locator app (see default instance `!Click_Accept!` in Section 4.2.3). Accordingly, the wizard lists Sec. 28 (3b) S.1 of the FDPA, Sec. 95 (5) of the TCA, and Sec. 12 (3) of the TMA (old version). These provisions express that dependency between consent and conclusion of a contract can indeed restrict the data subject's freedom of decision.

However, the wizard also lists further material that speaks in favor of `!Click_Accept!` being a "free decision." One example is legislative background documents deciding against strict prohibition of binding consent to the conclusion of contracts. Instead, such dependencies are only explicitly forbidden for advertisement and address trading by Sec. 28 (3, 3b) S.1 of the FDPA. Even in these cases, freedom of decision is only restricted when there is no access to equivalent contractual benefits. A court decision found "equivalent alternatives" to be available even for a company with an extremely high market share.



Overall Interpretation

Legal Concept: Consent

Instance: Click <Accept and download>

Relation: Free Decision

Legislative Purpose: Data privacy law is designed to protect individuals against infringement of their informational self-determination. Thus, the basic question is whether the consent provided by the data subject is an expression of self-determination or whether the consent was given under physical or mental pressure.

Pro

German Parliament Document 16/12011

Legislator purposely decided against strict prohibition of binding consent to conclusion of contracts.

Sec 28 (3b) S.1 FDPA

Dependency of the conclusion of a contract on the data subject's consent is only prohibited in case of advertisement and address trading and when no equivalent alternatives are available on the market.

Higher Regional Court Brandenburg 7 U 52/05

"Equivalent alternatives" (-> free decision) are typically given unless there is a monopoly.

Con

Sec 28 (3b) S. 1 FDPA, Sec. 95 (5) TCA, Sec. 12 (3) TMA (old)

Binding consent to conclusion of contracts is problematic. It may inflict mental pressure on data subject.

Borderline

Decision Commentary: Consent is based on a free decision. As there are several apps with comparable functionality available on the market, the user is not put under illegitimate pressure to reveal his personal data to KnowWhere/Google.

Figure 12 : User interface design for step "guided interpretation."

After selection, the wizard displays a dialogue for an overall interpretation (see Figure 12). The dialogue displays the current legal concept, relation and instance, as well as the selected materials and sources. The overall interpretation cannot be reduced to a binary "yes/no" decision since the materials and sources might contradict each other. Instead, the overall interpretation is adjusted by a slider with an underlying numerical value between -1 and +1. In the example, the developer positions the slider towards "Pro" because the majority of material and sources speak in favor of a subsumption.

The slider values are interpreted as follows

- Positive: Only when the slider value is above a configurable threshold (e.g., +0.5), the current relation is interpreted as *given* for the current instance.
- Negative: If the slider value is under a configurable threshold (e.g., -0.5), the current relation is interpreted as *not given* for the current instance.
- Borderline: In all other cases, (e.g., value between -0.5 and +0.5), we have a borderline case. The developer must be prompted to either terminate the subsumption algorithm with uncertain result and/or get advice from a legal expert. Otherwise, coherence to legal methodology is not given.

After adjusting the slider, the developer can save the interpretation along an additional free text commentary, as well as the selected material and the numerical slider value.

This step ends the subsumption process for the current legal concept. All remaining instances of the result set now become instances of, i.e., are subsumed under, the current legal concept LC of the chosen statutory ontology. All manual decisions taken by the developer in steps 5.3 and 5.5 are logged for an offline analysis by the legal expert. The analysis might lead to optimisations of future runs of the subsumption algorithm, e.g., by establishing general correspondences (see preliminary check in step 5.1).

6. Legal Advice

Semi-automated legal reasoning determines which legal concepts are given based on the current software. For example, in our scenario, the legal concepts "processing," "use" and "personal data" are given. However, the desired consequence of "Lawfulness" is not given since both "Consent" and "Permission or Order by FDPA or other Act" are not given. The developer is advised of this situation by visualising and marking the norm graph with unknown, given, or not given states of affairs by a wizard (see Figure 13).

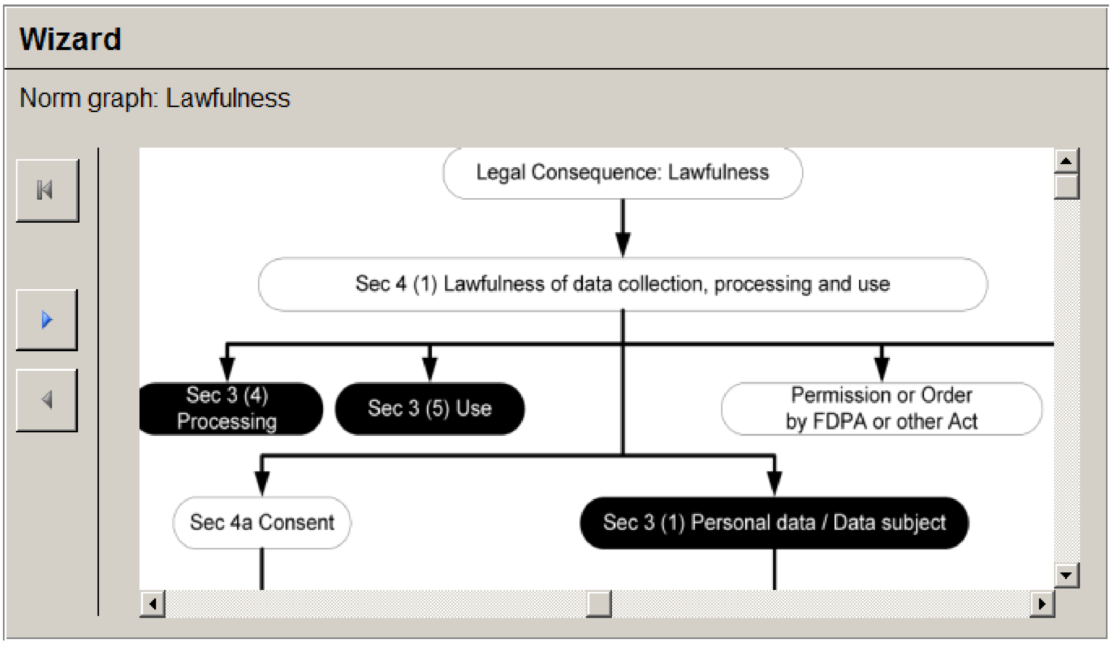

Wizard
Norm graph: Lawfulness
Legal Consequence: Lawfulness
Sec 4 (1) Lawfulness of data collection, processing and use
Sec 3 (4) Processing
Sec 3 (5) Use
Permission or Order by FDPA or other Act
Sec 4a Consent
Sec 3 (1) Personal data / Data subject

Figure 13: Design of a wizard showing the marked norm graph in our running example. A black node represents a "given" state of affairs; grey nodes are "unknown" and white nodes are "not given."

The developer may click on a node in the norm graph to further understand why it is marked as given, not given or unknown. If the developer clicks on the not given node "Sec. 4a Consent," the wizard shows the corresponding legal norm in more detail. This subsequent view essentially depicts the corresponding formalised legal norm of the data privacy ontology for private bodies in a graphical way.

The resulting user interface is shown in Figure 14 where the desired "Effectiveness" of a "Consent" is not given. Similar to the norm graph, the legal norm is also marked depending on whether a state of affairs is given, not given or unknown. Figure 14 also shows that, besides the graphical depiction, the norm text, sources and links to further information, are offered on additional tabs. This information has been provided by the legal expert when building the lexicon (see Section 3.1.2).

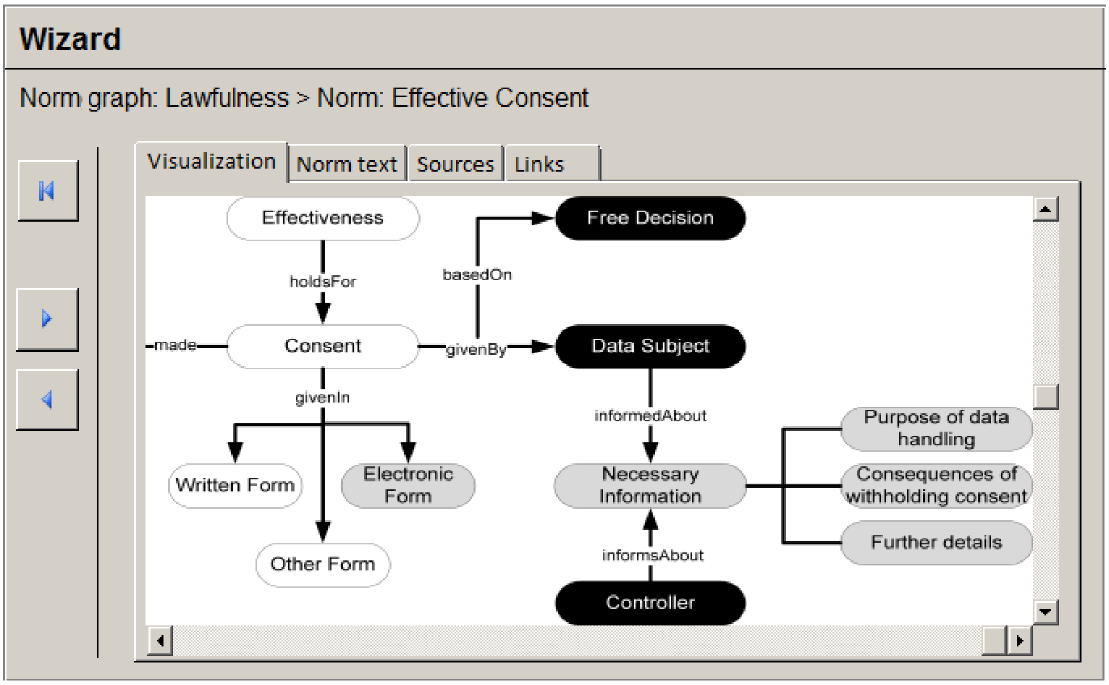
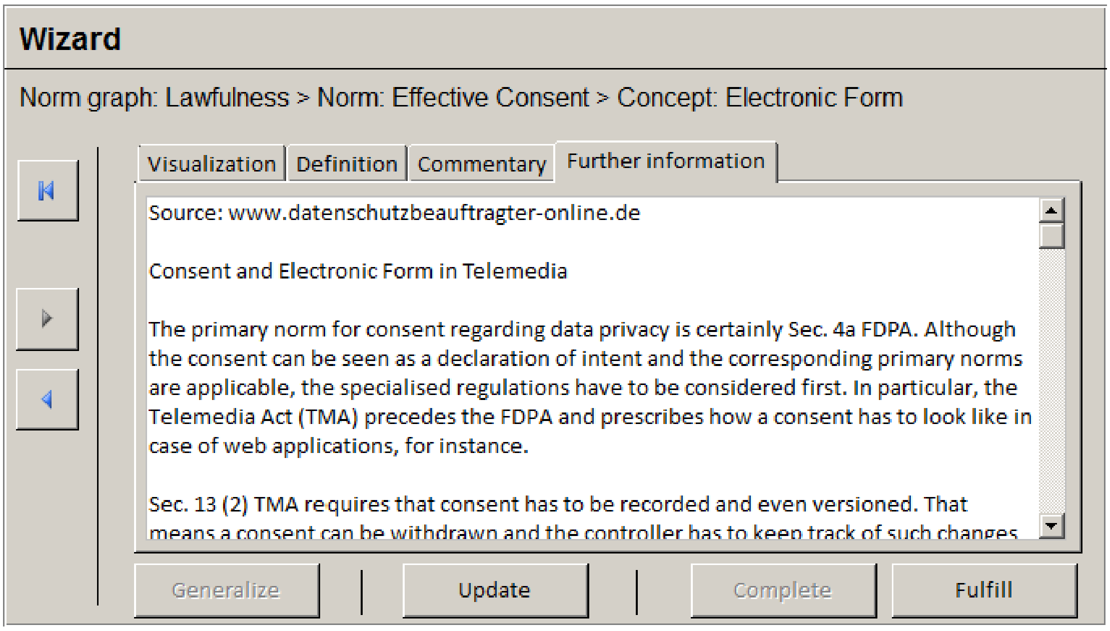


Figure 14: Design of a wizard showing the marked legal norm for Sec 4a (1) FPDA (Effective Consent).

Upon inspecting the graphical visualisation, the developer realises that the "Effectiveness" is not given because: a) it is unknown whether the requirements for "Electronic Form" are given and b) it is unknown whether the "Data Subject" has been properly "informedAbout" the "Necessary Information." Subsequently, the developer clicks on either node to learn more about their current state.

Figure 15 shows the user interface after clicking on "Electronic Form." The developer is informed about requirements for electronic consent according to Sec. 13 (2) of the TMA. The corresponding view offers a visualisation of the legal concept as formalised in the data privacy ontology for private bodies. In addition, the view offers tabs for the concept's definition, additional commentaries, or further information – all of which are part of the lexicon provided by the legal expert.

The developer is informed that effectiveness requires the declaration of consent to be recorded and even versioned in the case of a Web application, i.e., Telemedia (Sec. 13 (2) No 2, 3 of the TMA, Sec. 94 No 2, 3 of the TCA). It is now upon the developer to realise this requirement in the software. The developer continues by clicking on "Fulfill" what leads to an update of the corresponding instance in the ontology. The developer might navigate back to the norm or norm graph view by using the navigation bar on the left to see what has changed. For instance, "Consent" (see Figure 14) is now given but not "Effectiveness" since the user is not properly informed about the "Necessary Information" of processing personal data. Hence, the developer might click on "Necessary Information" to learn how this state of affairs can be fulfilled and so on. The developer continues until "Lawfulness" in the norm graph view (see Figure 13) is given.



Wizard

Norm graph: Lawfulness > Norm: Effective Consent > Concept: Electronic Form

Source: www.datenschutzbeauftragter-online.de

Consent and Electronic Form in Telemedia

The primary norm for consent regarding data privacy is certainly Sec. 4a FDPA. Although the consent can be seen as a declaration of intent and the corresponding primary norms are applicable, the specialised regulations have to be considered first. In particular, the Telemedia Act (TMA) precedes the FDPA and prescribes how a consent has to look like in case of web applications, for instance.

Sec. 13 (2) TMA requires that consent has to be recorded and even versioned. That means a consent can be withdrawn and the controller has to keep track of such changes

Figure 15 : Design of a wizard showing further information for Sec 13 (2) TMA (Electronic Form).

7. Proof of Concept

The main bottleneck of our approach is the formalisation of the norm graph. We expect a legal expert to formalise legal concepts, as well as the rule-like nature of legal norms and their relationships, in one or more statutory ontologies. This requires a major intellectual effort, so that the legal expert has to be supported as efficiently as possible and the correctness of modeling has to be ensured. Thus, the following questions have to be addressed:

(i) Is the proposed automatic extraction of legal concepts really useful?

(ii) Is the ontology language expressive enough for capturing legal concepts and norms?

(iii) Is the legal expert capable of modeling an ontology?

(iv) Do formalised concepts and norms really capture the semantics of the provisions?

With respect to (i), the automatic pre-processing proposed in Section 3.1.1 has been realised and applied to the German FDPA.[15] The FDPA contains 16669 words overall. The first step of automatic pre-processing allowed a reduction to 2291 words by singling out duplicates. The second step reduced this initial word list of 2291 words to 1556 candidates for modeling. 31% of the words were reduced through stemming (17.9%) and stop words (13.3%) as can be seen in Figure 16. There were 698 words with common stem which in turn have been reduced to 287 word stems. This resulted

[15] For a detailed discussion please refer to U Reppel, *Computergestützte Extraktion modellierungsrelevanter Rechtsbegriffe aus Gesetzestexten am Beispiel des BDSG* (Karlsruhe: Bachelor thesis, Karlsruhe Institute for Technology (KIT), Department of Informatics, 2010).

in 411 words removable by stem. There were an additional 21 removable other words what resulted in an overall 736 words that could be eliminated as not relevant for modeling. Concluding, the word count for modeling candidates has been reduced significantly and lowered the initial modeling effort for the legal expert.

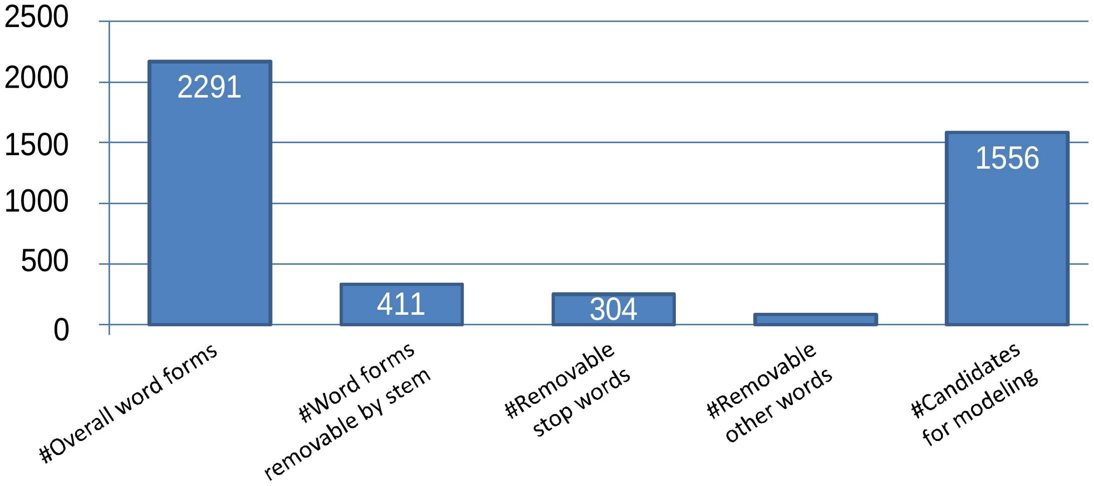


Figure 16: Summary of the results for automatic pre-processing of the German FDPA.

Regarding (ii), our work found that exceptions require special workarounds in the target language.[16] We classified legal norms according to their syntactic structure and derived representation requirements. Based on the requirements we have chosen the target language and ensured that each category can be properly represented.

With respect to (iii), our proof of concept comprised mock-ups for potential graphical user interfaces. One mock-up is shown in Figure 5, further ones are contained and documented in our prior work.[17] First experiences indeed have shown that the legal expert has to have basic knowledge in ontological modeling. The design of a proper graphical user interface requires experts in visualisation of law and is left as future work.

In order to ensure a semantically correct modeling (iv) we have designed a test case editor for the legal expert.[18] The editor allows defining test cases and comparing the automated legal reasoning with the manual legal reasoning result for specific legal concepts and rules. In case of deviations, the legal expert is supported in correcting the formalized legal concept or norm.

Besides the bottleneck of formalising statutory provisions, the following questions require a discussion:

(v) Does the automated legal reasoning provide correct results?

(vi) Is the advice for the developer presented properly?

[16] O Raabe, R Wacker, D Oberle, C Baumann and C Funk, *Recht ex machina* (Heidelberg: Springer Vieweg, 2012) at ch 13, 14.

[17] *Ibid*, sec 19.1 – 19.4.

[18] *Ibid*, sec 19.5.

With respect to (v), the main goal of this work is to provide decision support for the developer. Advices for achieving lawfulness represent a benefit since manual legal reasoning can often not be sought due to lack of a legal expert, high costs and time consumption. In order to be methodologically sound, our approach indeed aborts explicitly when no clear result can be obtained (e.g., during guided interpretation in Section 5.5) and advises to consult a legal expert.

In general, the prerequisite for correct results is a correct formalisation of the statutory provisions. This is ensured by the test case editor as we have learned above. However, errors in representing the subject matter, the automatic extraction of instances, and the subsumption algorithm can occur. Errors can be due to bugs in the realisation or due to wrong interactions of the developer during subsumption. In this case, wrong results would indeed be produced. Note, however, that the manual legal reasoning process and advice is also error-prone. One reason is that the developer has to communicate and explain the subject matter to the legal expert. Many misinterpretations usually occur during this process. The developer has to describe any information in natural language using terms a legal expert can understand. An alternative for consulting a legal expert is handbooks for developers that cover legal aspects. In contrast to our approach, such handbooks are often not up-to-date, the information is not case-specifically offered (has to be found first), and might not match the developer's needs.

With respect to (vi), our proof of concept comprised mock-ups for potential graphical user interfaces. Several mock-ups are shown in Section 6, further ones are contained and documented in our prior work.[19] The design of a proper graphical user interface requires more controlled experiments with a significant number of developers and is left as future work.

8. Related Work

Related work can be grouped in approaches that tackle *formalisation of legal norms*, *formalisation of legal concepts*, or *automated legal reasoning*. To the best of our knowledge there is no single approach that combines all three features for the purpose of engineering compliant software. Each of the groups will be discussed and positioned in the following subsections.

8.1. Formalisation of Legal Norms

The first group of related approaches analyses how and to what degree the rule-like structure of legal norms can be formalised and the required expressiveness of the formal language. Table 1 positions our approach to related work according to the following criteria:

- Focus on Statutory Provisions: The approach formalises statutory provisions (including legal concepts and norms) and not other directives, e.g., IT policies or general terms in the realm of law.
- Legal Methodology: The approach considers legal methodology and maintains the original structure of the norms.

[19] *Ibid,* ch 18, 21.

- Practicability: The approach is geared at a running system and not limited to pure theoretical discussions.
- Generality: The approach is usable for statutory provisions in any field of law.

Table 1 : Approaches that tackle the formalisation of legal norms.

Criterion Approach	Statutory Provisions	Legal Methodology	Practicability	Generality
McCarty[20]	-	✓	✓	✓
Kowalski[21]	✓	✓	✓	-
Sartor[22]	✓	✓	-	-
Gordon[23]	✓	✓	✓	-
Ringelstein[24]	-	-	✓	-
Prior work[25]	✓	✓	✓	-
This work	✓	✓	✓	✓

8.2. Formalisation of Legal Concepts

The second group of related approaches focuses on the formalisation of legal concepts or special legal knowledge. Most approaches apply ontologies to capture the intended meaning of concepts but do not consider the formalisation of the subject matter. However, the latter is of importance for semi-automation of the legal reasoning process (see approaches in Section 8.3). Table 2 positions our approach to related work according to the following criteria:

[20] L McCarty, "A Language for Legal Discourse I: Basic Features" in *Proceedings of the 2nd International Conference on Artificial Intelligence and Law (ICAIL '89)* (New York, NY, USA: ACM, 1989) 180–189.

[21] R Kowalski, "Legislation as Logic Programs" in G Comyn, N Fuchs and M Ratcliffe (eds) *Logic Programming in Action, Second International Logic Programming Summer School, LPSS '92, Zurich, Switzerland, September 7-11, 1992, Proceedings* (Berlin, Heidelberg: Springer, 1992) 203–230.

[22] R Marín and G Sartor, "Time and Norms: A Formalisation in the Event-calculus" in *Proceedings of the 7th International Conference on Artificial Intelligence and Law (ICAIL '99)* (New York, NY, USA: ACM, 1999) 90-99; A Artosi, G Governatori and G Sartor, "Towards a Computational Treatment of Deontic Defeasibility" in M Brown and J Carmo (eds), *Deontic Logic Agency and Normative Systems* (New York: Springer, 1995) 27-46; G Sartor, "The Structure of Norm Conditions and nonmonotonic Reasoning in Law" in *Proceedings of the 3rd international conference on Artificial intelligence and law (ICAIL '91)* (New York, NY, USA: ACM, 1991) 155-164.

[23] T Gordon, "Oblog-2: A Hybrid Knowledge Representation System for Defeasible Reasoning" in *Proceedings of the 1st International Conference on Artificial Intelligence and Law (ICAIL '87)* (New York, NY, USA: ACM, 1987) 231–239; T Gordon, „Constructing Arguments with a Computational Model of an Argumentation Scheme for Legal Rules: Interpreting Legal Rules as Reasoning Policies" in *Proceedings of the 11th International Conference on Artificial Intelligence and Law (ICAIL 2007)* (New York, NY, USA: ACM 2007) 117–121.

[24] C Ringelstein and S Staab, "Papel: Provenance-Aware Policy Definition and Execution" (2011) 15(1) *IEEE Internet Computing* 49–58.

[25] A Dietrich, P Lockemann and O Raabe: "Agent Approach to Online Legal Trade" in J Krogstie, A Opdahl and S Brinkkemper (eds), *Conceptual Modelling in Information Systems Engineering* (Berlin, Heidelberg: Springer, 2007) 177-194.

- Legal Methodology: The approach considers common legal procedures in the interpretation of legal concepts and fosters traceability and reproducibility.
- Subject Matter: The approach also considers the formalisation of the subject matter for semi-automation of the subsumption process.
- Foundational Ontology: The approach applies ontological analysis supported by a foundational ontology as starting point.
- Design Patterns: The approach applies ontology design patterns as best practices for re-occurring modelling needs.
- Quality Criteria: The approach relies on ontological quality criteria for optimal design of the ontology.

Table 2 : Approaches that tackle the formalisation of legal concepts.

Criterion Approach	Legal Methodology	Subject Matter	Foundational Ontology	Design Patterns	Quality criteria
Valente[26]	✓	-	-	-	-
Van Kralingen[27]	✓	-	-	-	-
Hoekstra[28]	✓	-	✓	✓	✓
Schweighofer[29]	✓	-	✓	✓	✓
Visser[30]	✓	✓	-	✓	-
Gangemi[31]	-	✓	✓	✓	-
Saias[32]	-	✓	✓	-	-
This work	✓	✓	✓	✓	✓

8.3. Automated Legal Reasoning

The third group of related approaches focuses on the automation of legal reasoning as a whole. There are diverse approaches resting on different technologies, e.g., case-

[26] A Valente and J Breuker, "A functional ontology of law" in G Bargellini and S Binazzi (eds) *Towards a Global Expert System in Law* (Padua: CEDAM Publishers, 1994) 201-212.

[27] R van Kralingen, "A Conceptual Frame-based Ontology for the Law" in *Proceedings of the First International Workshop on Legal Ontologies* (1997) 15–22.

[28] R Hoekstra et al, "The LKIF Core Ontology of Basic Legal Concepts" in P Casanovas, M Biasiotti, E Francesconi and M Sagri (eds), *Proceedings of the 2nd Workshop on Legal Ontologies and Artificial Intelligence Techniques* (CEUR, 2008) 43–63.

[29] E Schweighofer, "Semantic Indexing of Legal Documents" in E Francesconi et al (eds), *Semantic Processing of Legal Texts* (Berlin, Heidelberg: Springer, 2010) 157–169.

[30] P Visser, R van Kralingen and T Bench-Capon, "A Method for the Development of Legal Knowledge Systems" in *Proceedings of the 6th International Conference on Artificial Intelligence and Law (ICAIL '97)* (New York, NY, USA: ACM, 1999) 151–160.

[31] A Gangemi, "Introducing Pattern-based Design for Legal Ontologies" in J Breuker et al (eds), *Law, Ontologies and the Semantic Web - Channelling the Legal Information Flood* (IOS Press, 2009) 53–71.

[32] J Saias and P Quaresma, "A Methodology to Create Legal Ontologies in a Logic Programming Information Retrieval System" in V Benjamins et al (eds), *Law and the Semantic Web* (Berlin, Heidelberg: Springer, 2005) 185–200.

based reasoning or artificial neural networks. Table 3 positions our approach to related work according to the following criteria:

- Civil Law: Our approach is based on the continental European civil law system. Considering the differences to Common Law would require adaptations.
- Legal Methodology: The approach does not rest on probabilistic technologies, such as artificial neural networks, or fuzzy logic. While automated legal reasoning might be possible, it would not adhere to the strict reproducibility of statutory interpretation in Civil Law jurisdictions.
- Generalisation: The approach is not geared at a specific legal norm or provision, but can be applied generically.
- Layman Support: The approach provides intuitive support for semi-automated legal reasoning and the advice for the developer.

Table 3 : Approaches that tackle the automation of legal reasoning.

Criterion Approach	Civil Law	Legal Methodology	Generality	Layman Support
Bench-Capon[33]	-	✓	✓	-
Gordon[34]	✓	✓	-	-
Philipps[35]	-	-	✓	-
Bohrer[36]	✓	✓	-	-
Ring[37]	✓	✓	-	-
Prior Work[38]	✓	✓	-	-
This work	✓	✓	✓	✓

9. Conclusion

This paper presented an approach for engineering compliant software consisting of a subject matter ontology, a statutory ontology, the user-guided subsumption between both, and the realisation of consequences. The approach supports software developers in achieving legal compliance by design as a reaction to increasingly decentralised and dynamically combined software components.

[33] T Bench-Capon and G Sartor, "A Model of Legal Reasoning with Cases Incorporating Theories and Values" (2003) 150 (1-2) *Artificial Intelligence* 97–143.

[34] There is an inference system for the formal language Oblog already referenced in Section 8.1.

[35] L Philipps and G Sartor, "Introduction: From Legal Theories to Neural Networks and Fuzzy Reasoning" (1999) 7(2-3) *Artificial Intelligence and Law* 115–128.

[36] A Bohrer, *Entwicklung eines internetgestützten Expertensystems zur Prüfung des Anwendungsbereichs urheberrechtlicher Abkommen* (Kassel: University Press, 2003).

[37] S Ring, *Computergestützte Rechtsfindungssysteme, Voraussetzungen, Grenzen und Perspektiven,* (Köln, Berlin, Bonn, München: Heymanns, 1994).

[38] M Conrad et al, "Legal Compliance by Design: Technical Solutions for Future Distributed Electronic Markets" (2010) 21(3) *Journal of Intelligent Manufacturing* 321–333.

Future work will apply the same principles to software execution. Major differences to the engineering phase are: (i) less information is unknown and (ii) the duration of the user-guided legal reasoning process is more critical.

Further future work concerns the generalisation of the approach to other legal frameworks. The approach can in principle be applied to other legal frameworks, e.g., other domains such as the German Energy Industry Act, or other jurisdictions such as the Anglo-American case law. We also plan to consider contractual agreements, such as general terms and conditions.

Acknowledgements

The authors would like to express their gratitude to Christian Funk, Raphael Volz, Burkhard Schafer, and Susan Marie Thomas for their helpful advice and contributions.