**Noname manuscript No.**
(will be inserted by the editor)

# Machine Understandable Policies and GDPR Compliance Checking

**Piero A. Bonatti · Sabrina Kirrane · Iliana M. Petrova · Luigi Sauro**

Received: 31st October 2019 / Accepted: tba

**Abstract** The European General Data Protection Regulation (GDPR) calls for technical and organizational measures to support its implementation. Towards this end, the SPECIAL H2020 project aims to provide a set of tools that can be used by data controllers and processors to automatically check if personal data processing and sharing complies with the obligations set forth in the GDPR. The primary contributions of the project include: (i) a policy language that can be used to express consent, business policies, and regulatory obligations; and (ii) two different approaches to automated compliance checking that can be used to demonstrate that data processing performed by data controllers / processors complies with consent provided by data subjects, and business processes comply with regulatory obligations set forth in the GDPR.

## 1 Introduction

The European General Data Protection Regulation (GDPR), which came into force on the 25th of May 2018, defines legal requirements concerning the processing and sharing of personally identifiable data. In addition, the legislation calls for technical and organizational measures to support its implementation.

When it comes to legal informatics there is a large body of work on legal knowledge representation and reasoning (cf., [3, 5, 13, 20, 24, 26]), however said approaches are usually foundational in nature and as such are not readily accessible for companies looking for technical means to demonstrate GDPR compliance.

Recently we have seen the emergence of GDPR compliance tools (cf., [1, 15, 22, 23]) in the form of predefined questionnaires that enable data controllers and processors to assess the compliance of services and products that process personal data. The primary limitation of said tools is their lack of support for automated compliance checking.

In order to fill this gap, SPECIAL builds upon a rich history of policy language research from the Semantic Web community (cf., [8, 16, 18, 32, 33]), and shows how together machine understandable policies and automated compliance checking can be used to demonstrate compliance with legal requirements set forth in the GDPR.

In particular, we introduce the SPECIAL policy language and discuss how it can be used to express consent, business policies, and regulatory obligations. In addition, we describe two different approaches to automated compliance checking used to demonstrate that: (i) data processing performed by data controllers / processors complies with consent provided by data subjects; and (ii) business processes comply with regulatory obligations set forth in the GDPR. In addition, we provide a high-level overview of our compliance checking algorithm and present the results of our initial performance evaluation.

The remainder of the paper is structured as follows: *Section* 2 describes our analysis of the text of the GDPR. *Section* 3 provides a short background on the standard ontology languages OWL and OWL2. *Section* 4 introduces the SPECIAL policy language, which provides a machine understandable encoding of consent. *Section* 5 discusses how the SPECIAL policy language can be used to encode business policies and regulatory obligations. *Section* 6 presents our compliance checking algorithm and the results of our initial performance evaluation. *Section* 7 points to related

Piero Bonatti · Iliana M. Petrova · Luigi Sauro
Università di Napoli Federico II, Naples, Italy
E-mail: pab@unina.it

Sabrina Kirrane
Vienna University of Economics and Business, Vienna, Austria
E-mail: sabrina.kirrane@wu.ac.at

work on GDPR compliance. Finally, we present our conclusions and interesting directions for future work in *Section* 8.

## 2 Requirements Analysis

The GDPR, which came into effect on the 25th of May 2018, superceeds the Data Protection Directive 95/46/EC [9]. Given that the primary goal of the SPECIAL H2020 project is to provide a set of tools that can be used by data controllers and processors to automatically check if personal data processing and sharing comply with the obligations set forth in the GDPR, we specifically focus on personal data processing that is performed after the GDPR came into effect. A necessary first step in this regard is to better understand the text of the GDPR, its interpretation by legal professionals, and the role of machine understandable representations, and automated compliance checking.

### 2.1 GDPR Analysis

Legal rules are composed of several constructs, prohibitions (used to describe what is not permitted), permissions (used to describe what is permitted), obligations (used to describe requirements that must be fulfilled), and dispensations (used to describe exemptions), commonly referred to as deontic concepts. In addition to these common constructs, the legal language contains constraints (used to limit the scope of permissions, prohibitions, obligations and dispensations), definitions (used to establish meaning), dispositions (used to highlight best practices/ suggestions), and opening clauses (used to indicate the need to consult National or European legislation).

In this paper, we illustrate a policy language used to represent regulative norms in the form of permissions, obligations, and prohibitions. The analysis presented in this section serves to better understand the structure of the GDPR such that it is possible to identify fragments of the legislation that can be modeled in a manner that supports automated compliance checking of personal data processing and sharing with and between companies. Although constitutive norms could be used to provide requirements to organizations with respect to the processing of personal data within and between organizations, instead, we provide companies with a policy language that can be used to model data processing and sharing requirements and to simply attest to the existence of required controls, procedures, and documentation. The development of a fully fledged legal reasoning system, based on the modelling of normative requirements prescribed in legal text (c.f., [10, 24, 25]) is outside of the scope of the SPECIAL project.

When it comes to encoding legislative requirements using machine understandable representations, such that it is possible to perform automated compliance, major considerations include:

*Connectedness* of the various articles, paragraphs, and points, which can either explicitly refer to another piece of legislation (e.g., *"scientific or historical research purposes or statistical purposes in accordance with Article 89(1)"*) or implicitly to knowledge about the law (e.g., *"Personal data shall be: (a) processed lawfully, fairly and in a transparent manner in relation to the data subject (lawfulness, fairness and transparency)"*). In either case, from an automated compliance checking perspective, it is clear that legal requirements are not separate and distinct rules but rather rules need to be linked, clustered, and/or generalized in a manner that enables the validation of a combination of rules.

In SPECIAL we do not try to encode the entire GDPR, but rather focus on encoding legislative obligations (relating to several articles, paragraphs, and points) such that: (i) data processing performed by data controllers / processors complies with consent provided by data subjects; and (ii) business processes comply with regulatory obligations set forth in the GDPR.

*Temporal expressions* provide contextual information that is relevant for the interpretation of actions that need to be taken. Several different types of temporal expressions can be found in the text of the GDPR, for instance:

- ... *"the right to withdraw his or her consent at any time"* (Article 7 paragraph 3);
- ... *"processing based on consent before its withdrawal"* (Article 7 paragraph 3, Article 13 paragraph 2, Article 14 paragraph 2);
- ... *"prior to giving consent"* (Article 7 paragraph 3);
- ... *"at the time when personal data are obtained"* (Article 13 paragraphs 1 and 2);
- ... *"the personal data shall no longer be processed"* (Article 21 paragraph 3);

In SPECIAL we provide support for such temporal requirements by recording in a suitable *transparency ledger* when consent was obtained or when the data processing/sharing happened. This information is used for both ex-ante and ex-post compliance checking (as well as other purposes, discussed later).

### 2.2 Legal Interpretations

The GDPR defines several potential legal bases (consent, contract, legal obligation, vital interest, public interest, exercise of official authority, and legitimate interest) under which companies can legally process personal data. In order to determine if personal data processing is legally valid,

the legal inquiry process usually involves gathering specific information such as: (i) the *personal data* collected from the data subject; (ii) the *processing* that are performed on the personal data; (iii) the *purpose* of such processing; (iv) where data are *stored and for how long*; and (v) with whom data is *shared*. The answers provided to said questions enable legal professionals to determine which articles need to be consulted in order both to assess the lawfulness of processing and to identify relevant legal obligations.

Although, the open textured nature of legal texts is a highly desirable feature, as it leaves room for interpretation on a case by case basis, such ambiguity poses challenges for automatic compliance checking. In terms of legal interpretations, legal professionals also need to interpret the facts of the case with respect to relevant National or European legislation (e.g., opening clauses) and subjective terms (e.g., single words or parts of a sentence that can be interpreted in various ways). Here legal knowledge graphs could potentially play a crucial role as they allow for the modeling of both legislation and cases in a machine-readable format, based on standardization activities such as European Law Identifier (ELI) and the European Case Law Identifier (ECLI), which provide technical specifications for web identifiers and vocabularies that can be used to describe metadata pertaining to legal documents. Such a legal knowledge graph could be used not only to identify case specific legislation, but also to uncover if there have been any prior cases that could be used to reduce ambiguity.

The SPECIAL policy language has been developed together with legal professionals who well-versed in the interpretation of legal texts. Going forward we envisage that legal knowledge graphs could be used to reduce subjectivity thus allowing us to perform automated compliance checking for a broader set of legislative requirements.

### 2.3 Machine Understandable Representations

The GDPR poses at least two requirements that call for a machine-understandable representation of data usage modalities. Article 30 states that each controller shall maintain a record of the personal data processing activities under its responsibility. The first paragraph specifies that such a ledger should describe (among other information) the following aspects of *data usage*:

P1. the *purpose* of processing;
P2. a description of the *categories of data subjects* and of the *categories of personal data*;
P3. the categories of *recipients* to whom the personal data have been or will be disclosed;
P4. *transfers* of personal data to a third country or an international organization (since cross-border data transfer are subject to limitations);
P5. the envisaged *time limits for erasure* of the different categories of data;
P6. information about the *processing*, such as the security measures mentioned in Article 32.

Recital 42 stresses that, where processing is based on the data subject's consent, the controller should be able to demonstrate that the data subject has consented to the processing.

SPECIAL addresses this issue by recording consent in the transparency ledger (cf. Sec. 2.1). The description of consent is similar to the description of processing activities as per Article 30. While Article 6.1.(a) – that introduces consent as a legal basis for personal data processing – and Recital 42 explicitly mention only the purpose of processing, Articles 13 and 14 add the other elements P2–P6 listed above. Concerning P6 (processing), it should be specified whether any automated decision making is involved, including profiling.

### 2.4 Automated Compliance Checking

Once such data usage descriptions are encoded in a machine-understandable way, several tasks, related to GDPR compliance, can be automated, including:

T1. Checking whether the processing complies with several restrictions imposed by the GDPR, such as additional requirements on the processing of sensitive data, restrictions on cross-border transfers, and compatibility of data usage with the chosen legal basis. This kind of validation requires a machine-understandable formalization of the relevant parts of the GDPR.
T2. Checking whether a specific operation is permitted by the available consent.
T3. Running ex-post auditing on the controller's activities. In SPECIAL this task is supported by logging data processing events in the transparency ledger, and comparing such events with consent.
T4. Finding the consent that justifies a specific processing (for auditing or responding to a data subject's inquiry).

The transparency ledger is also used in SPECIAL to provide dashboards to data subjects, that support them in monitoring the use of their data and *explaining* why their consent allowed specific operations. Such dashboards can also be used as a uniform interface to let data subjects exercise their rights (access to data, right to erasure, etc.) as specified by Articles 15–18 and 21–22.

## 3 Background

The Web Ontology Language (OWL)[1] is a World Wide Web Consortium (W3C) standard that allows for the representation of knowledge about both concrete and abstract things and how they relate to one another. OWL models knowledge using classes and properties (commonly called roles) and instances (known as individuals). In particular, OWL provides for: (i) A rich set of relations between classes, roles and individuals (for example, `owl:sameAs`, `owl:equivalentClass`, `owl:differentFrom`); (ii) A number of logical operations (for example, `owl:unionOf` and `owl:intersectionOf`); (iii) Several cardinality constraints (for example, `owl:someValuesFrom`, `owl:allValuesFrom`, `owl:cardinality`, `owl:minCardinality` and `owl:maxCardinality`). OWL2 comes in two flavours (OWL2 Full and OWL2 DL). OWL2 DL is composed of three profiles (OWL2EL, OWL2QL and OWL2RL) that are based on well used DL constructs. OWL 2 EL is designed for applications that require very large ontological. Polynomial time reasoning is achieved at the cost of expressiveness. OWL 2 QL is particularly suitable for applications with lightweight ontologies and a large number of individuals, which need to be accessed via relational queries. Finally, OWL 2 RL provides support for applications with lightweight ontologies and a large number of individuals that make use of rule based inference and constraints mechanisms.

SPECIAL usage policies and the entries of the transparency ledger are encoded using a fragment of OWL2 which we call OWL2 PL, where the letters PL stand for policy language. In particular, we make use of `owl:intersectionOf` in order to define complex classes (e.g., a simple policy) where policy instances are composed of instances of a given list of classes (e.g., data, processing, purpose, storage, and recipients). While, `owl:unionOf` is used to define new classes that are the union of two or more other classes. This construct is particularly useful for constructing usage policies from sets of simple policies. In turn, `owl:someValuesFrom` is used to constrain the type of values that are associated with a property. For instance we can say that a policy must have a processing property that specifies the type of processing. More generally, the SPECIAL policies languages relies on vocabularies that are encoded using lightweight profiles of OWL2 such as OWL2-EL and OWL2-QL.

[1] https://www.w3.org/TR/owl2-overview/

## 4 Consent Compliance Checking

Although there are several potential legal bases that could be used to lawfully process personal data, in SPECIAL we have a particular focus on consent. Thus in this section we present the SPECIAL policy language and demonstrate how it can be used to encode consent in a manner than enables automated compliance checking.

### 4.1 Encoding Usage Descriptions and Consent

The common structure of the activity records and of the consent forms, consisting of properties P1–P6, is called *simple (usage) policy* in SPECIAL. In general, both the controller's activities and the consent of data subjects can be described by a *set* of simple usage policies (covering different data categories and purposes), called *full (usage) policies*. Each simple policy can be specified simply by attaching to each property $P_i$ (such as purpose, data category, recipients, etc.) a term selected from a suitable *vocabulary* (ontology).

*Example 1* A company – call it BeFit – sells a wearable fitness appliance and wants (i) to process biometric data (stored in the EU) for sending health-related advice to its customers, and (ii) share the customer's location data with their friends. Location data are kept for a minimum of one year but no longer than 5; biometric data are kept for an unspecified amount of time. In order to do all this legally, BeFit needs consent from its customers. Consent can be represented with two simple policies, specified using SPECIAL's vocabularies:

```
{
  hasPurpose: FitnessRecommendation,
  hasData: BiometricData,
  hasProcessing: Analytics,
  hasRecipient: BeFit,
  hasStorage: { hasLocation: EU }
}

{
  hasPurpose: SocialNetworking,
  hasData: LocationData,
  hasProcessing: Transfer,
  hasRecipient: DataSubjFriends,
  hasStorage: {
      hasLocation: EU,
      hasDuration: [1year,5year]
      }
}
```

If `HeartRate` is a subclass of `BiometricData` and `ComputeAvg` is a subclass of `Analytics`, then the above consent allows BeFit to compute the average heart rate of the data subject in order to send her fitness recommendations. BeFit customers may restrict their consent, e.g. by picking a specific recommendation modality, like "recommendation via SMS only". Then the first line should be replaced with something like:

```
hasPurpose:{
  FitnessRecommendation,
  contact: SMS}
```

Moreover, a customer of BeFit may consent to the first or the second argument of the union, or both. Their consent would be encoded, respectively, with the first simple policy, the second simple policy, or both. Similarly, each single process in the controller's business application may use only biometric data, only location data, or both. Accordingly, it may be associated to the first simple policy, the second simple policy, or both. ∎

The temporary exemplifying policy language vocabularies reported in SPECIAL's deliverables have been obtained by adapting previous standardized terms introduced by initiatives related to privacy and digital rights management, such as P3P[2] and ODRL,[3]. More refined vocabularies have been recently proposed by W3C's *Data Privacy Vocabularies and Controls Community Group*, (DPVCG) [27], promoted by SPECIAL and spanning a range of stakeholders wider than the project's consortium. The current vocabularies can be found on DPVCG's website[4].

As shown in Example 1, usage policies can be formatted with a minor extension of JSON (in particular, compound terms and policy sets require additional operators), while vocabularies can be encoded in RDFS or lightweight profiles of OWL2 such as OWL2-EL and OWL2-QL.

A grammar for SPECIAL policy expressions in Backus-Naur form (BNF) format is presented in Figure 1. The categories DataVocabExpression, PurposeVocabExpression, ProcessingVocabExpression, RecipientVocabExpression, LocationVocabExpression, DurationVocabExpression are specified by DPVCG's vocabularies.

### 4.2 Compliance Checking

Internally, SPECIAL's components encode also policies and the entries of the transparency ledger with a fragment (profile) of OWL2 called $\mathcal{PL}$ (policy logic) [7]. The adoption of a logic-based description language has manifold reasons, related to its clean, unambiguous semantics, that is a must for policy languages. A formal approach brings the following advantages:

- strong correctness and completeness guarantees on the algorithms for permission checking and compliance checking;
- the mutual coherence of the different reasoning tasks related to policies, such as policy validation, permission checking, compliance checking, and explanations (cf. tasks T1–T4 and the subsequent paragraph);
- correct usage after data is transferred to other controllers (i.e. interoperability). When it comes to so-called *sticky policies* [28], that constitute a sort of a license that applies to the data released to third parties, it is essential

[2] http://www.w3.org/TR/P3P11
[3] https://www.w3.org/TR/odrl/
[4] www.w3.org/community/dpvcg/

Fig. 1: SPECIAL's Usage Policy Language Grammar

**UsagePolicy** :='ObjectUnionOf' '(' **BasicUsagePolicy** { **BasicUsagePolicy** }* ')'
| **BasicUsagePolicy**

**BasicUsagePolicy** :='ObjectIntersectionOf' '(' **Data Purpose Processing Recipients Storage** ')'

**Data** :='ObjectSomeValueFrom' '(' 'spl:hasData' **DataExpression** ')'

**Purpose** :='ObjectSomeValueFrom' '(' 'spl:hasPurpose' **PurposeExpression** ')'

**Processing** :='ObjectSomeValueFrom' '(' 'spl:hasProcessing' **ProcessingExpression** ')'

**Recipients** :='ObjectSomeValueFrom' '(' 'spl:hasRecipient' **RecipientExpression** ')'

**Storage** :='ObjectSomeValueFrom' '(' 'spl:hasStorage' **StorageExpression** ')'

**DataExpression** :='spl:AnyData' | **DataVocabExpression**

**PurposeExpression** :='spl:AnyPurpose' | **PurposeVocabExpression**

**ProcessingExpression** :='spl:AnyProcessing' | **ProcessingVocabExpression**

**RecipientsExpression** :='spl:AnyRecipient' | 'spl:Null' | **RecipientVocabExpression**

**StorageExpression** :='spl:AnyStorage' | 'spl:Null' |
'ObjectIntersectionOf' '(' **Location Duration** ')'

**Location** :='ObjectSomeValueFrom' '(' 'spl:hasLocation' **LocationExpression** ')'

**Duration** :='ObjectSomeValueFrom' '(' 'spl:hasDuration' **DurationExpression** ')'
| 'DataSomeValueFrom' '(' 'spl:durationInDays' **IntervalExpression** ')'

**LocationExpression** :='spl:AnyLocation' | **LocationVocabExpression**

**DurationExpression** :='spl:AnyDuration' | **DurationVocabExpression**

**IntervalExpression** :='DatatypeRestriction' '(' 'xsd:integer' **LowerBound UpperBound** ')'

**LowerBound** :='xsd:minInclusive' **IntegerLiteral**

**UpperBound** :='xsd:maxInclusive' **IntegerLiteral**

**IntegerLiteral** := **stringOfDigits** '^^' 'xsd:integer'

**stringOfDigits** := *a sequence of digits enclosed in a pair of " (U+22)*

that all parties understand the sticky policy in the same way.

Policies are modeled as OWL2 *classes*. If the policy describes a controller's activity, then its instances represent all the operations that the controller may possibly execute. If the policy describes consent, then its instances represent all the operations permitted by the data subject. A description of (part of) the controller's activity – called *business policy* in SPECIAL (possibly represented as a *transparency log entry*) – *complies* with a consent policy if the former is a subclass of the latter, that is, all the possible operations described by the business policies are also permitted by the given consent.

*Example 2* Consider again Example 1. The JSON-like representation used there can be directly mapped onto an OWL2 class `ObjectUnionOf(`$P_1$ $P_2$`)`, where $P_2$ is[5]:

```
ObjectIntersectionOf(
  ObjectSomeValueFrom(
    hasPurpose SocialNetworking )
  ObjectSomeValueFrom(
    hasData LocationData)
  ObjectSomeValueFrom(
    hasProcessing Transfer)
  ObjectSomeValueFrom(
    hasRecipient DataSubjFriends)
  ObjectSomeValueFrom(
    hasStorage ObjectIntersectionOf(
      ObjectSomeValueFrom(hasLocation: EU)
      DataSomeValueFrom(hasDuration
        DatatypeRestriction(xsd:integer
          xsd:minInclusive "365"^^xsd:integer
          xsd:maxInclusive "1825"^^xsd:integer
)))
```

In order to check whether a business policy $BP$ (encoded as an OWL2 class) complies with the above policy one should check whether the former is a subclass of the latter, that is, whether:

`SubClassOf(`$BP$ `ObjectUnionOf(`$P_1$ $P_2$`))`

is a logical consequence of the ontology that defines SPECIAL's vocabularies. ∎

## 5 Business Processes Compliance Checking

Beyond consent, the GDPR defines obligations that apply to the data controllers / processors internal systems and processes. Here are two examples:

- whenever the data controller operates on personal data, it must *acquire explicit consent* from the involved data subjects, unless the purpose of data processing falls within a set of exceptional cases (e.g. the processing is required by law); cf. Article 6.1, (b)–(f);

[5] We omit $P_1$ due to space limitations; the reader may easily derive it by analogy with the above example.

Fig. 2: SPECIAL's Business Policy Language Grammar

**BusinessPolicy** := **BasicBP** |
'ObjectUnionOf' '(' **BasicBP** { **BasicBP** }* ')'

**BasicBP** :='ObjectIntersectionOf' '(' **Data Purpose Processing Recipients Storage** {**Duty**}* {**LegalBasis**} ')'

**Data** := *see* Section *4*

**Purpose** := *see* Section *4*

**Processing** := *see* Section *4*

**Recipients** := *see* Section *4*

**Storage** := *see* Section *4*

**Duty** :='ObjectSomeValuesFrom' '(' 'sbpl:hasDuty' **DutyExpression** ')'

**DutyExpression** :='sbpl:AnyDuty' | **DutyVocabExpression**

**LegalBasis** :='ObjectSomeValuesFrom' '(' 'sbpl:hasLegalBasis **LegalBasisVocabExpression** ')'

- whenever data are transferred to a third country whose data protection regulations do not match the EU requirements, alternative guarantees must be provided, e.g. in the form of company regulations called *binding corporate rules*, cf. Article 47 and, more generally, GDPR Chapter V (Transfer Of Personal Data To Third Countries Or International Organisations).

Moreover, and differently from the above examples, the GDPR sets obligations that are not directly related to the controller's business processes, such as the requirement that data subjects have the right to *access, rectify, and delete* their personal data. In order to fulfill such obligations, data controllers have to set up suitable processes. Last but not least, it is useful to label the controller/processors processes with the legal basis for the processing; this helps in assessing and demonstrating the lawfulness of data processing activities. For automated compliance checking descriptions of internal systems and processes should be adequately formalized in a machine-understandable way; moreover, the formalization should represent accurately the real processes, in order to make the automated compliance verification reliable.

### 5.1 Encoding Business Processes as Policies

In SPECIAL, we address a concrete setting in which a partial and abstract description of processes is available. Each process description is shaped like a *formalized business policy* consisting of the following set of features:

- the file(s) to be processed;
- the software that carries out the processing;
- the purpose of the processing;
- the entities that can access the results of the processing;

- the details of where the results are stored and for how long;
- *the obligations that are fulfilled while (or before) carrying out the processing;*
- *the legal basis of the processing.*

It is not hard to see that the first five elements in the above list match SPECIAL's usage policy language (UPL) introduced in *Section* 4. As far as the above elements are concerned, the only difference between UPL expressions and a business policy is the granularity of attribute values. For example, the involved data (specified in the first element of the above list) are not expressed as a general, content-oriented category, but rather as a concrete set of data sources or data items. Such objects can be modeled as instances or subclasses of the general data categories illustrated in *Section* 4, thereby creating a link between digital artifacts and usage policies. Similar considerations hold for the other attributes:

- processing is not necessarily described in the abstract terms adopted by the processing vocabulary introduced in *Section* 4; in a business policy, this can be specified by naming concrete software procedures;
- the purpose of data processing may be directly related to the data controller's mission and products;
- recipients may consist of a concrete list of legal and/or physical persons, as opposed to general categories such as `Ours` or `ThirdParty`;
- storage may be specified by a list of specific data repositories, at the level of files and hosts.

With this level of granularity, specific authorizations can be derived from the business policy, for example:

> *The indicated software procedure can read the indicated data sources. The results can be written in the specified repositories. The specified recipients can read the repositories...*

This methodology for generating authorizations fosters a close correspondence between the business policy and the actual behavior of the data controller's systems and processes.

The attribute encoding obligations is not part of usage policies. It plays a dual role, representing:

- preconditions authorizations specified by the business policy, e.g. if the obligation is something like `getValidConsent` then the derived authorizations is a *rule* like *the specified software can read the data sources if consent has been given*;
- obligation assertions (under human responsibility) that the data controller has set up *processes for fulfilling the indicated obligations* – e.g. a process to obtain consent from the data subjects – which is relevant to checking compliance with the GDPR.

### 5.2 Business Policies in OWL2

A basic business policy is simply a usage policy (as in *Section* 4) extended with zero or more obligations, and a legal basis, encoded with attributes `hasDuty` and `hasLegalBasis`, for example the following policy associates the collection of personal demographic information to the obligations to get consent and let the data subject exercise her rights:

```
ObjectIntersectionOf(
   ObjectSomeValuesFrom
      (spl:hasData svd:Demographic)
   ObjectSomeValuesFrom
      (spl:hasProcessing svpr:Collect)
   ObjectSomeValuesFrom
      (spl:hasPurpose svpu:Account)
   ObjectSomeValuesFrom
      (spl:hasRecipient svr:Ours)
   ObjectSomeValuesFrom
      (spl:hasStorage
      ObjectIntersectionOf(
         spl:hasLocation svl:OurServers
         spl:hasDuration svdu:Indefinitely
      )
   )
   ObjectSomeValuesFrom
      (sbpl:hasDuty getValidConsent)
   ObjectSomeValuesFrom
      (sbpl:hasDuty getAccessReqs)
   ObjectSomeValuesFrom
      (sbpl:hasDuty getRectifyReqs)
   ObjectSomeValuesFrom
      (sbpl:hasDuty getDeleteReqs)
   ObjectSomeValuesFrom
      (sbpl:hasLegalBasis A6-1-a-explicit-consent)
)
```

Similarly to usage policies, *general* business policies can be composed by enclosing several basic business policies inside the `ObjectUnionOf` operator of OWL2. The syntax and the logical semantics of SPECIAL's Business Policy Language are specified in Figure 2. The values for attributes DutyVocabExpression and LegalBasisVocabExpression are specified in DPVCG's vocabularies.

### 5.3 Partial Encoding of the GDPR in OWL2

The GDPR cannot be fully axomatized due to the usual difficulties that arise in axiomatizing legal text (especially the frequent use of subjective terms as highlighted in *Section* 2). However it is possible to encode some constraints that should hold over the different attributes of a business policy. At the top level, the formalization is organized as follows:

```
ObjectUnionOf(
   ObjectIntersectionOf(
     Chap2_LawfulProcessing
     Chap3_RightsOfDataSubjects
     Chap4_ControllerAndProcessorObligations
     Chap5_DataTransfer
   )
   Chap9_Derogations
)
```

Informally, the above expression says that either the requirements of GDPR Chapters 1–5 are satisfied, or some of the derogations provided by GDPR Chapter 9 should apply. In turn, each of the above terms is equivalent to a compound OWL2 class that captures more details from the regulation. Here we illustrate part of the formalization of GDPR Chapter 2 for an example. `Chap2_LawfulProcessing` is equivalent to the following expression:

```
ObjectUnionOf(
   Art6_LawfulProcessing
   Art9_SensitiveData
   Art10_CriminalData
)
```

The above three conditions apply, respectively, to non-sensitive personal data, sensitive data, and criminal data. At least one of the three conditions should be satisfied. In turn, `Art6_LawfulProcessing` is defined as:

```
ObjectUnionOf(
   ObjectSomeValuesFrom(spl:hasData
     SensitiveData_as_per_Art9
   )
   ObjectSomeValuesFrom(spl:hasData
     CriminalConvictionData_as_per_Art10
   )
   Art6_1_LegalBasis
   Art6_4_CompatiblePurpose
   )
)
```

Roughly speaking, the above union represents an implication in disjunctive normal form, and should be read like this: if the data involved in the processing is neither sensitive nor criminal conviction data, then either the fundamental legal bases of Art. 6(1) apply, or the processing is compatible with the original purpose for collecting the data as per Art. 6(4). In order to capture this meaning, class `Art6_1` is defined as:

```
ObjectSomeValuesFrom(hasLegalBasis
   ObjectUnionOf(
     Art6_1_a_Consent
     Art6_1_b_Contract
     Art6_1_c_LegalObligation
     Art6_1_d_VitalInterest
     Art6_1_e_PublicInterest
     Art6_1_f_LegitimateInterest
   )
)
```

Roughly speaking, this definition means that a business policy satisfies the requirements of Art. 6(1) if it contains a clause

```
ObjectSomeValueFrom( hasLegalBasis X )
```

where $X$ is some of the above classes corresponding to points *a–f* of Art. 6(1). In practice, this means that a human expert has to pick an appropriate legal basis for each business policy. Similarly, the formalization of Article 9 applies to sensitive data categories only, and requires a legal basis from a different list. So the term `SensitiveData_as_per_Art9` is equivalent to:

```
ObjectUnionOf(
   ObjectSomeValuesFrom(spl:hasData
     not_SensitiveData_as_per_Art9
   )
   ObjectSomeValuesFrom(hasLegalBasis
     ObjectUnionOf(
       Art9_2_a_Consent
       Art9_2_b_EmploymentAndSocialSecurity
       Art9_2_c_VitalInterest
       Art9_2_d_LegitimateActivitiesOfAssociations
       Art9_2_e_PublicData
       Art9_2_f_Juducial
       Art9_2_g_PublicInteres
       Art9_2_h_PreventiveOrOccupationalMedicine
       Art9_2_i_PublicHealth
       Art9_2_j_ArchivingResearchStatistics
     )
   )
)
```

The rest of the regulation is formalized with a similar approach.

### 5.4 Compliance Checking

Let us now make an example of compliance checking of a business policy w.r.t. the above axiomatization. Consider the following business policy:

```
ObjectIntersectionOf(
   ObjectSomeValuesFrom( hasData Religion )
   ObjectSomeValuesFrom( hasProcessing Collect )
   ObjectSomeValuesFrom( hasPurpose
     PersonalisedBenefits )
   ObjectSomeValuesFrom( hasStorage
     ObjectSomeValuesFrom( hasLocation EU ))
   ObjectSomeValuesFrom( hasRecipient
     DataProcessor )
   ObjectSomeValuesFrom( hasDuty
     Art12-22_SubjectRights )
   ObjectSomeValuesFrom( hasDuty
     Art32-37_Obligations )
   ObjectSomeValuesFrom( hasLegalBasis
     Art6_1_a_Consent )
)
```

This policy is not a subclass of the formalized GDPR (hence it does not pass the compliance check) because

`Religion` is classified as sensitive data (it is a subclass of `SensitiveData_as_per_Art9`). Then the business policy is not a subclass of `Art9_SensitiveData`, because the legal basis is not among the required list. Moreover, the business policy is not covered by the derogations provided by GDPR Chapter 9 (details are omitted here). As a consequence, the business policy does not satisfy the conditions specified by `Chap2_LawfulProcessing`. Note that this kind of compliance checking is able to verify the coherency of the different parts of a business policy.

If `Religion` was replaced by any non-sensitive data category such as `Demographic`, then the policy would be compliant because it would be a subclass of `Art6_LawfulProcessing`. This satisfies the condition called `Chap2_LawfulProcessing`. The `hasDuty` attributes of the business policy suffice to satisfy `Chap3_RightsOfDataSubjects` and `Chap4 ControllerAndProcessorObligations`. `Chap5_DataTransfer` would also be satisfied since the processing does not involve any transfers outside the EU.

## 6 The Automated Compliance Checking Algorithm

Business policies (that describe the processing of each of the controller's processes) are not only needed to fulfill the requirements of Article 30. They can also be used to check whether a running process complies with the available consent, as a sort of access control system. Several implementation strategies are possible, depending on the controller's system architecture; to fix ideas, the reader may consider the following generic approach: Each of the controller's processes is labeled with a corresponding business policy that describes it, and before processing a piece of data, the business policy is compared with the data subject's consent to check whether the operation is permitted.

In general, such compliance checks occur frequently enough to call for a scalable implementation. Consider, for example, a telecom provider that collects location information to offer location-based services. Locations cannot be stored without a legal basis, such as law requirements or consent – not even temporarily, while a batch process selects the parts that can be legally kept. So compliance checking needs to be executed on the fly. In order to estimate the amount of compliance checks involved, consider that the events produced by the provider's base stations are approximately 15000 per second; the probing records of wi-fi networks are about 850 millions per day.

In order to meet such performance requirements, SPECIAL has developed ad-hoc reasoning algorithms for $\mathcal{PL}$ [7], a new profile of OWL2 developed by SPECIAL that is able to express the usage policies, business policies, and GDPR's formalization illustrated in the previous sections. $\mathcal{PL}$'s assertions support class name inclusions (`SubClassOf` axioms), class name disjointness (`DisjointClasses` axioms), role range restriction axioms (`ObjectPropertyRange`), and role functionality axioms (`FunctionalObjectProperty`). $\mathcal{PL}$'s queries are inclusions of classes that are unions of *simple $\mathcal{PL}$ concepts*. The latter are conjunctions of class names, interval-valued concrete features (`DataSomeValueFrom`), and existential role restrictions (`ObjectSomeValueFrom`) whose range is recursively a simple $\mathcal{PL}$ concept.

| | |
|---|---|
| processor: | Intel Xeon Silver 4110 |
| cores: | 8 |
| cache: | 11M |
| RAM: | 198 GB |
| OS: | Ubuntu 18.4 |
| JVM: | 1.8.0_181 |
| heap: | 32 GB (actually used: less than 700 MB). |

Table 1: The system used in experiments

SPECIAL's reasoning algorithms, described in [7], leverage $\mathcal{PL}$'s simplicity to achieve unprecedented reasoning speed. Compliance checking is split into two phases: first, business policies are normalized and closed under the axioms contained in the vocabularies; in the second phase, business policies are compared with consent policies with a *structural subsumption* algorithm. We have just completed the evaluation of a sequential Java implementation of those algorithms, called PLR. We chose Java to facilitate the comparison with other engines, by exploiting the standard OWL APIs, and we refrained to apply parallelization techniques in order to assess the properties of the basic algorithms. Before discussing more performant implementation options, we report the performance of PLR over the pilots of SPECIAL.[6] The pilots share a common ontology that defines personal data categories, purposes, and the other features that are needed in usage policies, cf. SPECIAL's deliverable D2.5.[7] In the experiments reported here, business policies are those developed for the pilots, while consent policies are generated randomly by simulating the opt-in and opt-out choices of data subjects, picked from the options provided by the data controllers.

PLR can pre-compute the first phase, since the business policies are known in advance and are typically static. So the runtime cost is reduced to structural subsumption. In this way, on the test cases derived from SPECIAL's use cases (cf. Table 2), the performance we achieve, respectively, is $150\mu$sec and $190\mu$sec per compliance check, using the system illustrated in Table 1.

[6] We have also run sets of synthetic experiments with increasing size to assess the scalability of PLR. They are omitted here due to space limitation and will be published in a forthcoming paper. We anticipate that these experiments confirm that PLR is faster than its competitors.

[7] `https://zenodo.org/record/2545177`

| | Pilot 1 | Pilot 2 |
|---|---|---|
| *Ontology* | | |
| inclusions | 186 | 186 |
| disjoint class axioms | 11 | 11 |
| property range axioms | 10 | 10 |
| functional properties | 8 | 8 |
| classification hierarchy height | 4 | 4 |
| *Business policies* | | |
| # generated policies | 120 | 100 |
| avg. simple pol. per full pol. | 2.71 | 2.39 |
| *Consent policies* | | |
| # generated policies | 12,000 | 10,000 |
| avg. simple pol. per full pol. | 3.77 | 3.42 |
| *Test cases* | | |
| # compliance checks | 12,000 | 10,000 |

Table 2: Test cases derived from SPECIAL's pilots

This means that PLR alone can execute about 6000 compliance checks per second and more than 518 million checks per day, that is, 60% of wi-fi probing events and 40% of base station events.

In order to raise performance and match the frequency of events mentioned at the beginning of this section, one can re-engineer PLR using a language more performant than Java, and/or parallelize processing by means of big data architectures. Compliance checking is particularly well suited to parallelization, since each test is independent from the others and no synchronization is required. Additionally, the investigation of parallelization within PLR's algorithms is under investigation.

## 7 Related Work

From a GDPR compliance perspective, there exist several compliance tools (cf. [1, 15, 22, 23]) that enable companies to assess the compliance of applications and business processes via predefined questionnaires, however such tools do not cater for automated compliance checking of business processes with respect to legal obligations. When it comes to automated compliance checking there are two primary bodies of work, one focusing on legal knowledge representation and reasoning, and the other on policy representation and reasoning.

### 7.1 Legal knowledge representation and reasoning

There is a large body of work on legal knowledge representation (cf.[5, 26]) and reasoning (cf. [3, 13, 20, 24]). From a representation perspective, Bartolini et al. [5] and Pandit et al. [26] propose ontologies that can be used to model data protection requirements. While, Palmirani et al. [24] and Athan et al. [3] demonstrate how LegalRuleML can be used to specify legal norms. The work by Lam and Hashmi [20] and Governatori et al. [13] also builds upon LegalRuleML, however the focus is more on ensuring business process compliance.

Despite superficial similarities, SPECIAL's policy framework and the many works on legal reasoning have different goals. The survey [29] lists several applications of logic and reasoning to the legal domain that can be grouped as follows:

a. Supporting legislators in writing less ambiguous, possibly normalized legal documents.
b. Modeling legal concepts and definitions.
c. Interpreting the law.
d. Modeling the debates and pleadings that take place in courts, and deriving legal qualifications.

The work on vocabularies carried out by SPECIAL and the DPVCG can be regarded as a streamlined version of (b), while the use case of business policy validation based on GDPR's partial formalization does not really match any of the above points. Policy validation is less ambitious than legal reasoning; it is only aimed at checking whether the different properties of the policy are mutually coherent (e.g. by checking that the legal basis matches the data category), and whether all relevant parts have been included (such as the appropriate obligations in case of consent-based processing or data transfers outside the EU). The latter is a way to check whether the human responsible has "ticked all the necessary boxes", and by no means tackles the legal reasoning required to assess whether the obligations have been actually and appropriately fulfilled; according to our experience in SPECIAL, algorithms and ontologies are not yet trusted on this matter – especially due to the severe consequences in case of wrong decisions.

Both business policy validation and compliance checking with regard to consent policies shall verify that business policies do the right thing in *all contexts* while the literature on legal reasoning focusses on whether a legal qualification (such as an obligation, a permission, the validity of a contract, etc.) holds in a *specific situation* [29]. This difference has remarkable technical consequences: as we have already pointed out, validation in all contexts (i.e. policy comparison) is intractable in rule-based languages (that are common in the works on legal reasoning, since the seminal paper [30]), and even undecidable if rules are recursive [6]. The DL-based approach we adopted guarantees decidability and – under suitable hypotheses – tractability.

The ambitious goals of legal reasoning have been tackled with sophisticated formalisms, such as deontic, nonmonotonic, and input-output logics, and neo-Davidsonian encodings of natural language sentences, see for example [2, 12, 14, 17, 21, 34]. Fortunately, the different goals of SPECIAL's compliance checking make these complications

unnecessary. Simplicity is strategic for the project, since the personnel that is expected to write the business policies has no background on mathematical logic, deontic logic, nor nonmonotonic logic. The usability of SPECIAL's simple, form-like business policies has been successfully tested by the industrial partners of SPECIAL. Summarizing, SPECIAL trades advanced legal reasoning capabilities for usability and scalability.

### 7.2 Policy representation and reasoning

Both rule languages and OWL2 have already been used as policy languages; a non-exhaustive list is [8, 16, 18, 32, 33]. As noted in [6], the advantage of OWL2 – hence description logics – is that all the main policy-reasoning tasks are decidable (and tractable if policies can be expressed with OWL2 profiles), while compliance checking is undecidable in rule languages, or at least intractable – in the absence of recursion – because it can be reduced to datalog query containment. So an OWL2-based policy language is a natural choice in a project like SPECIAL, where policy comparison is the predominant task. Among the aforementioned languages, both Rei [18] and Protune [8] support logic program rules, which make them unsuitable to SPECIAL's purposes. KAoS [32] is based on a description logic that, in general, is not tractable, and supports role-value maps – a construct that easily makes reasoning undecidable (see [4], Chap. 5). The papers on KAoS do not discuss how to address this issue.

P3P's privacy policies – that are encoded in XML – and simple $\mathcal{PL}$ policies have a similar structure: the tag `STATEMENT` contains tags `PURPOSE`, `RECIPIENT`, `RETENTION`, and `DATA-GROUP`, that correspond to the analogous properties of SPECIAL's usage policies. Only the information on the location of data is missing. The tag `STATEMENT` is included in a larger context that adds information about the controller (tag `ENTITY`) and about the space of web resources covered by the policy (through so-called *policy reference files*). Such additional pieces of information can be directly encoded with simple $\mathcal{PL}$ concepts.

There exist several well-engineered reasoners for OWL2 and its profiles. Hermit [11] is a general reasoner for OWL2. Over the test cases inspired by SPECIAL's use cases, it takes 3.67 ms and 3.96 ms per compliance check, respectively, that is, over 20 times longer than PLR. ELK [19] is a specialized polynomial-time reasoner for the OWL2-EL profile. It does not support functional roles, nor the interval constraints used to model storage duration, therefore it cannot be used to reason on the $\mathcal{PL}$ profile. Konclude [31] is a highly optimized reasoner with "pay-as-you-go" strategies (i.e. it becomes more efficient on less complex profiles of OWL2). Konclude is designed for classification, and is currently not optimized for subsumption tests (i.e. the reasoning task underlying compliance checks). Consequently, it turns out to be slower than Hermit on our test cases.

## 8 Conclusion and Future Work

The overarching goal of the SPECIAL project is to develop tools and technologies that enable data controllers and processors to comply with personal data processing obligations specified in the GDPR. In this paper, we presented the SPECIAL policy language and discussed how it can be used to encode consent, business policies, and regulatory obligations. In addition we described the SPECIAL approaches to GDPR compliance checking and presented the results of our initial performance evaluation.

Ongoing/future work includes: the optimisation of the existing compliance checking algorithm to cater for automated compliance checking for a broader set of legislative requirements; and the development of an algebra that can be used to combine multiple policies, for instance where there is a need to aggregate data from multiple data sources.

## Acknowledgment

This research is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement N. 731601. The authors are grateful to all of SPECIAL's partners; without their contribution this project and its results would not have been possible.

## References

1. S. Agarwal, S. Steyskal, F. Antunovic, and S. Kirrane. Legislative compliance assessment: Framework, model and gdpr instantiation. In *Annual Privacy Forum*, 2018.
2. G. Antoniou, N. Dimaresis, and G. Governatori. A modal and deontic defeasible reasoning system for modelling policies and multi-agent systems. *Expert Syst. Appl.*, 36(2), 2009.
3. T. Athan, H. Boley, G. Governatori, M. Palmirani, A. Paschke, and A. Z. Wyner. Oasis LegalRuleML. In *ICAIL*, volume 13, 2013.
4. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*, 2003. Cambridge University Press. ISBN 0-521-78176-0.
5. C. Bartolini, R. Muthuri, and C. Santos. Using ontologies to model data protection requirements in workflows. In *JSAI International Symposium on Artificial Intelligence*, 2015.

6. P. A. Bonatti. Datalog for security, privacy and trust. In *Datalog Reloaded - First International Workshop, Datalog 2010*, 2010. doi: 10.1007/978-3-642-24206-9_2.
7. P. A. Bonatti. Fast compliance checking in an OWL2 fragment. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI*, 2018. doi: 10.24963/ijcai.2018/241.
8. P. A. Bonatti, J. L. D. Coi, D. Olmedilla, and L. Sauro. A rule-based trust negotiation system. *IEEE Trans. Knowl. Data Eng.*, 22(11):1507–1520, 2010. doi: 10.1109/TKDE.2010.83.
9. E. 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. *Official Journal of the EC*, 23(6), 1995.
10. F. Gandon, G. Governatori, and S. Villata. Normative requirements as linked data. In *The 30th international conference on Legal Knowledge and Information Systems (JURIX)*, 2017.
11. B. Glimm, I. Horrocks, B. Motik, G. Stoilos, and Z. Wang. Hermit: An OWL 2 reasoner. *J. Autom. Reasoning*, 53(3):245–269, 2014. doi: 10.1007/s10817-014-9305-1.
12. G. Governatori, F. Olivieri, A. Rotolo, and S. Scannapieco. Computing strong and weak permissions in defeasible logic. *J. Philosophical Logic*, 42(6), 2013. doi: 10.1007/s10992-013-9295-1.
13. G. Governatori, M. Hashmi, H.-P. Lam, S. Villata, and M. Palmirani. Semantic business process regulatory compliance checking using LegalRuleML. In *European Knowledge Acquisition Workshop*, 2016.
14. J. F. Horty. *Agency and Deontic Logic*. Oxford University Press, 2001.
15. Information Commissioner's Office (ICO) UK. Getting ready for the GDPR, 2017. URL `https://ico.org.uk/for-organisations/resources-and-support/data-protection-self-assessment/getting-ready-for-the-gdpr/`.
16. S. Jajodia, P. Samarati, M. L. Sapino, and V. S. Subrahmanian. Flexible support for multiple access control policies. *ACM Trans. Database Syst.*, 26(2), 2001.
17. A. J. I. Jones and M. J. Sergot. On the characterization of law and computer systems: the normative systems perspective. In J.-J. C. Meyer and R. J. Wieringa, editors, *Deontic Logic in Computer Science: Normative System Specification*, chapter 8. Wiley, 1993.
18. L. Kagal, T. W. Finin, and A. Joshi. A policy language for a pervasive computing environment. In *4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY)*. IEEE Computer Society, 2003. ISBN 0-7695-1933-4.
19. Y. Kazakov, M. Krötzsch, and F. Simancik. The incredible ELK - from polynomial procedures to efficient reasoning with EL ontologies. *J. Autom. Reasoning*, 53(1):1–61, 2014. doi: 10.1007/s10817-013-9296-3.
20. H.-P. Lam and M. Hashmi. Enabling reasoning with LegalRuleML. *Theory and Practice of Logic Programming*, 19(1), 2019.
21. D. Makinson and L. van der Torre. *What is Input/Output Logic?* Springer, 2003. ISBN 978-94-017-0395-6.
22. Microsoft Trust Center. Detailed GDPR Assessment, 2017. URL `http://aka.ms/gdprdetailedassessment`.
23. Nymity. GDPR Compliance Toolkit. URL `https://www.nymity.com/gdpr-toolkit.aspx`.
24. M. Palmirani, G. Governatori, A. Rotolo, S. Tabet, H. Boley, and A. Paschke. LegalRuleML: XML-based rules and norms. In *Workshop on Rules and Rule Markup Languages for the Semantic Web*, 2011.
25. M. Palmirani, M. Martoni, A. Rossi, C. Bartolini, and L. Robaldo. Pronto: Privacy ontology for legal reasoning. In *International Conference on Electronic Government and the Information Systems Perspective*, 2018.
26. H. J. Pandit, K. Fatema, D. OSullivan, and D. Lewis. Gdprtext-gdpr as a linked data resource. In *European Semantic Web Conference*, 2018.
27. H. J. Pandit, A. Polleres, B. Bos, R. Brennan, B. P. Bruegger, F. J. Ekaputra, J. D. Fernández, R. G. Hamed, E. Kiesling, M. Lizar, E. Schlehahn, S. Steyskal, and R. Wenning. Creating a vocabulary for data privacy - the first-year report of data privacy vocabularies and controls community group (DPVCG). In *OTM 2019 Conferences - Confederated International Conferences: CoopIS, ODBASE, C&TC*, 2019.
28. S. Pearson and M. C. Mont. Sticky policies: An approach for managing privacy across multiple parties. *IEEE Computer*, 44(9), 2011.
29. H. Prakken and G. Sartor. Law and logic: A review from an argumentation perspective. *Artif. Intell.*, 227, 2015. doi: 10.1016/j.artint.2015.06.005.
30. M. J. Sergot, F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond, and H. T. Cory. The british nationality act as a logic program. *Commun. ACM*, 29(5), 1986. doi: 10.1145/5689.5920.
31. A. Steigmiller, T. Liebig, and B. Glimm. Konclude: System description. *J. Web Semant.*, 27-28:78–85, 2014. doi: 10.1016/j.websem.2014.06.003.
32. A. Uszok, J. M. Bradshaw, R. Jeffers, N. Suri, P. J. Hayes, M. R. Breedy, L. Bunch, M. Johnson, S. Kulkarni, and J. Lott. KAoS policy and domain services: Towards a description-logic approach to policy representation, deconfliction, and enforcement. In *4th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY)*. IEEE Computer Society,

2003. ISBN 0-7695-1933-4.
33. T. Y. C. Woo and S. S. Lam. Authorizations in distributed systems: A new approach. *Journal of Computer Security*, 2(2-3):107–136, 1993. doi: 10.3233/JCS-1993-22-304.
34. G. P. Zarri. *Representation and Management of Narrative Information - Theoretical Principles and Implementation*. Advanced Information and Knowledge Processing. Springer, 2009. ISBN 978-1-84800-077-3.