Citation for published version:
De Vos, M, Kirrane, S, Padget, J & Satoh, K 2019, ODRL Policy Modelling and Compliance Checking. in *Rules and Reasoning. RuleML+RR 2019: International Joint Conference on Rules and Reasoning, RuleML+RR 2019, Bolzano, Italy, September 16-19, 2019, Proceedings.* Lecture Notes in Computer Science, vol. 11784, Springer, pp. 36-51. https://doi.org/10.1007/978-3-030-31095-0_3

DOI:
10.1007/978-3-030-31095-0_3

Publication date:
2019

Document Version
Peer reviewed version

Link to publication

ODRL policy modelling and compliance checking

Marina De Vos[1], Sabrina Kirrane[2], Julian Padget[1], and Ken Satoh[3]

[1] University of Bath, Bath, United Kingdom
{m.d.vos,j.a.padget}@bath.ac.uk
[2] Vienna University of Economics and Business, Vienna, Austria
sabrina.kirrane@wu.ac.at
[3] National Institute of Informatics and Sokendai, Tokyo, Japan
ksatoh@nii.ac.jp

Abstract. This paper addresses the problem of constructing a policy pipeline that enables compliance checking of business processes against regulatory obligations. Towards this end, we propose an Open Digital Rights Language (ODRL) profile that can be used to capture the semantics of both business policies in the form of sets of required permissions and regulatory requirements in the form of deontic concepts, and present their translation into Answer Set Programming (via the Institutional Action Language (InstAL)) for compliance checking purposes. The result of the compliance checking is either a positive compliance result or an explanation pertaining to the aspects of the policy that are causing the non-compliance. The pipeline is illustrated using two (key) fragments of the General Data Protect Regulation, namely Articles 6 (Lawfulness of processing) and Articles 46 (Transfers subject to appropriate safeguards) and industrially-relevant use cases that involve the specification of sets of permissions that are needed to execute business processes. The core contributions of this paper are the ODRL profile, which is capable of modelling regulatory obligations and business policies, the exercise of modelling elements of GDPR in this semantic formalism, and the operationalisation of the model to demonstrate its capability to support personal data processing compliance checking, and a basis for explaining why the request is deemed compliant or not.

1 Introduction

The General Data Protection Regulation (GDPR), which came into effect in May 2018, provides data controllers and processors with legal requirements and guidelines concerning the processing and sharing of personal data. Although there are a number of self assessment tools that can be used by companies to manually assess GDPR compliance (cf., Information Commissioner's Office (ICO) UK [19], Microsoft Trust Center [24], Nymity [26]), automated compliance checking of business processes with respect to legal obligations is highly desirable, especially when such systems are regularly updated (e.g., maintenance and feature updates).

In order to support automated compliance checking it is necessary to encode both GDPR requirements and business processes in a machine understandable format. Existing work in the area focuses on modelling legal requirements using semantic technologies [4, 5, 6, 31]; reasoning over legal rules [2, 11, 29]; and reasoning over business

policies [11, 18, 22]. At the same time there are several semantic technology based general policy languages, such as KAoS [9], Rei [21] and Protune [8], which could potentially be used to model and reason over legal requirements and business policies. However, there are no standardised mechanisms for representing policies, such that compliance can be checked automatically.

When it comes to Web standardisation activities relating to semantic based policy languages, the closest match is the Open Digital Rights Language (ODRL) information model and associated vocabularies, which enables various parties to specify permissions, prohibitions, and duties relating to actions performed on assets. Although ODRL is primarily used to specify licenses, it could be adapted/extended, via the ODRL profile mechanism, such that it is also possible to specify a broader set of policies.

Thus, in this paper, we introduce our ODRL Regulatory Compliance Profile (ORCP), which can be used to model both regulatory requirements and sets of permissions required to execute a business process, and discuss how these policies can be translated into Answer Set Programming (ASP) [17] rules such that it is possible to automatically check compliance, identify conflicts and propose resolution strategies. ASP is a declarative programming language with a mathematical foundation guaranteeing sound and completeness of the results. The translation to ASP is facilitated through InstAL [10, 23, 27], a domain specific action language for modelling normative systems and legal frameworks.

Summarising our contributions, we: (i) propose an ODRL profile, which is capable of modelling regulatory permissions, prohibitions, obligations, and dispensations, and permissions needed to execute business processes within a company; (ii) show how elements of GDPR can be modelled using this semantic formalism; (iii) demonstrate how ASP rules based on InstAL can be used for automatic compliance checking; and (iv) propose a mechanism to provide evidence in support of the compliance decision and identify what is missing.

The remainder of the paper is structured as follows: *Section* 2 discusses related work on modelling and reasoning over legal requirements. *Section* 3 introduces our ODRL profile that can be used to encode both regulatory requirements and permissions required by business processes. *Section* 4 demonstrates how InstAL can be used for ODRL policy compliance checking and explanation generation. *Section* 5 offers some evaluation of and reflection on our proposal. Finally, *Section* 6 concludes the paper and identifies several open research questions.

2 Related Work

Over the years there have been several prominent works that focus specifically on modelling legal requirements using semantic technologies [4, 5, 6, 31]. Boer et al. [6] build upon the MetaLex [5] eXtensible Markup Language legislation encoding mechanism, in order to define a language and vocabularies that cater for the interchange of legal knowledge, known as the Legal Knowledge Interchange Format (LKIF). Bartolini et al. [4] propose an ontology that can be used to model data protection requirements, and demonstrates how it can be integrated into existing business workflows. Like Bartolini et al. [4], Pandit et al. [31] focus specifically on data protection, however they demon-

strate how the European Legislation Identifier (ELI) ontology can be used to model the GDPR as linked data. Outputs include a DCAT[1] catalog containing the official text of the GDPR and a SKOS[2] ontology defining concepts related to GDPR.

In addition, there is a body of work relating to legal reasoning [2, 11, 18, 18, 22, 29]. Palmirani et al. [29] and Athan et al. [2] demonstrate how LegalRuleML, an extension of RuleML[7], can be used to specify legal norms, guidelines, and policies. Dimyadi et al. [11] compares LegalRuleML and LKIF formalisms and highlights that one of the primary benefits of Semantic technology based approaches is the availability of mature reasoning engines. While, Lam and Hashmi [22] demonstrate how LegalRuleML can be translated into defeasible logic, which allows for modelling and reasoning over business policies. Governatori et al. [18] in turn shows how LegalRuleML together with Semantic technologies is used for business process regulatory compliance checking.

In the early days of Semantic Web research, researchers developed general policy languages with formal semantics (such as KAoS [9], Rei [21] and Protune [8]), that could potentially be used to model and reason over legal permissions, prohibitions, obligations, and dispensations. More recently, the Open Digital Rights Language, which was primarily intended to define rights to or to limit access to digital resources (cf. [30]), has demonstrated its potential as a general policy language. For instance, researchers have hinted as it how it could be used to express: access policies [33]; requests, data offers and agreements [32]; and basic regulatory policies [1]. While, Fornara and Colombetti [12] consider how to add obligations to (an earlier version) of ODRL and subsequently in Fornara et al. [13] how to reason over such ODRL extensions, using additional ontologies and semantic rules.

In this paper, we propose an ODRL regulatory compliance profile that can be used to model both regulatory requirements in terms of deontic concepts (permissions, prohibitions, obligations and dispensations), and business policies in the forms of sets of permissions required to execute the policy. The ODRL policies are subsequently translated into ASP rules, which not only cater for automatic compliance checking, but also for non compliance detection and explanation.

3 Modelling legislative requirements and business policies using ODRL

We start by presenting our generalised ODRL information model and subsequently demonstrate how it can be used to encode legal requirements and permissions required by business processes. At this stage we do not aim to be exhaustive in terms of modelling, but rather our objective is to demonstrate that ODRL can be extended to cater for the modelling of regulatory policies (in the form of nested permissions, prohibitions, obligations, and dispensations) and business policies (in the form of discrete permissions that are needed to execute a business process). The full ODRL Regulatory Compliance Profile, including examples in both Turtle and JSON-LD serialisations are

[1] DCAT, `https://www.w3.org/TR/vocab-dcat/`
[2] SKOS, `https://www.w3.org/TR/skos-reference/`

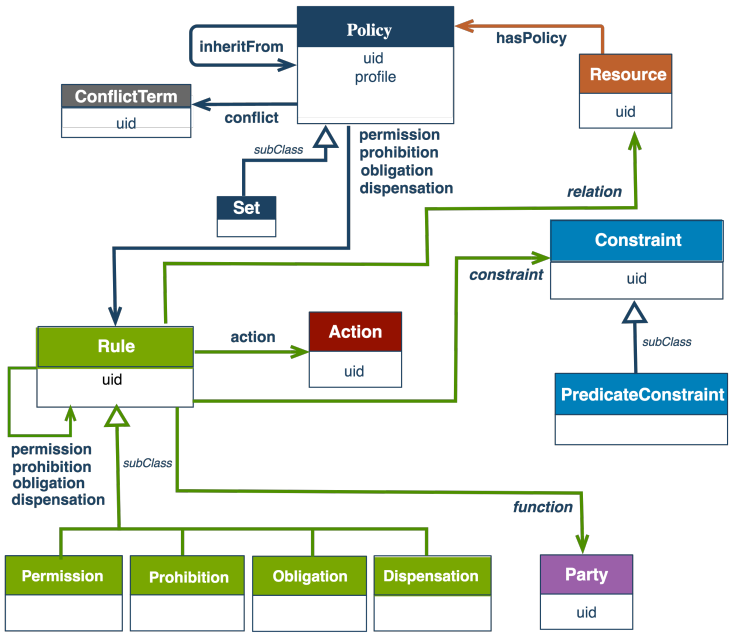

Fig. 1. A generalised ODRL Information Model

available in the form of a draft specification[3] and an ontology[4]. Our expectation is that the ODRL Regulatory Compliance profile ontology will be extended by domain experts with additional classes and properties to support not only the modelling of relevant articles from the GDPR but also other legislation.

3.1 Generalising the ODRL information model

Figure 1 provides a high level overview of a generalised ODRL information model. Like ODRL, a `Policy` is composed of a `Set` of `Rule`s each of which govern an `Action` that is performed by `Party`. Given that `Asset` is too specific, it is replaced by a more general `Resource` class, which is a superclass of `Asset`. As per ODRL, the `ConflictTerm` class is used to specify the conflict resolution strategy.

In terms of exclusions, the `Agreement` and `Offer` policy subclasses, the `Duty` rule subclass, and the `duty`, `failure`, `remedy`, and `consequence` properties are removed in the new model, as these classes and properties were strongly motivated by use cases relating to licensing.

[3] ODRL Regulatory Compliance Profile, `https://ai.wu.ac.at/policies/orcp/regulatory-model.html`

[4] ODRL Regulatory Compliance Profile Ontology, `https://ai.wu.ac.at/policies/orcp/odrl_regulatory_profile.ttl`

From an inclusions perspective, in addition to the `Permissions`, `Prohibitions` rule subclasses already provided for by ODRL, `Obligation` and `Dispensation` rule subclasses are added to the profile, such that it is possible to express deontic concepts needed for better modeling regulatory policies, both structurally and semantically. In addition, in order to support the modelling of nested rules, `permission`, `prohibition`, `obligation` and `dispensation` properties have been added to the abstract `Rule` class. For instance, the following text could be modelled as a permission with a nested prohibition: "*processing is necessary for the purposes of the legitimate interests pursued by the controller or by a third party, except where such interests are overridden by the interests or fundamental rights and freedoms of the data subject which require protection of personal data, in particular where the data subject is a child.*".

Additionally it is worth noting that the ODRL constraint functionality has been curtailed, such that the model now contains an abstract `Constraint` class, which needs to be subclassed in order to support specific constraints with well defined semantics necessary for automated compliance checking. The current model includes a `PredicateConstraint`, which is used to specify object assertions expected for a given predicate. However, it is expected that additional `Constraint` subclasses (with well defined semantics) will be added as the need arises.

3.2 The ODRL Regulatory Compliance Profile

In addition to the core classes and properties outlined in the previous section, based on our analysis of Article 6 and Article 46 of the GDPR, the profile defines several additional classes (e.g., `LegalBasis`, `Purpose`, and `Location`) and properties (e.g., `legalBasis`, `purpose`, `processingLocation`, `recipientLocation`, `organisationType`, `appropriateSafeguards`, and `dataSubjectProvisions`), which are needed in order to check the compliance of business processes with said articles. The example ODRL Regulatory Compliance Profile policies (based on paragraphs 1 and 2 of Article 46 of the GDPR) presented in this paper are encoded using the Turtle serialisation syntax, with the `odrl` prefix used to denote the ODRL ontology[5] and the `orcp` prefix used to denote the proposed regulatory compliance ontology[6]. Figure 2 depicts an extract from Article 46 of the GDPR, where colour coding is used to highlight parties, resources, actions, and constraints that need to be modelled using our regulatory profile. Given that the objective is to enable companies to assert that various data subject provisions and safeguards exist, we treat each point under a paragraph as a single resource and do not model the parties, actions and constraints relating to these resources.

The ODRL Regulatory Model representation of Article 46 is presented in Listing 1 and the permission needed to execute the business process is presented in Listing 2. More specifically, the regulatory policy presented in Listing 1 states that the `Transfer` (action) of `PersonalData` (resource) to an `InternationalOrganisation` or `ThirdCountry` is permitted if `EnforceableDataSubjectRights`, `LegalRemediesForDataSubjects`, and `appropriateSafeguards` of type

[5] `<http://www.w3.org/ns/odrl/2/>`

[6] `<http://example.com/odrl:profile:regulatory-compliance/>`

Art. 46 GDPR – Transfers subject to appropriate safeguards

1. In the absence of a decision pursuant to Article 45(3), a controller or processor may transfer personal data to a third country or an international organisation only if the controller or processor has provided appropriate safeguards, and on condition that enforceable data subject rights and effective legal remedies for data subjects are available.
2. The appropriate safeguards referred to in paragraph 1 may be provided for, without requiring any specific authorisation from a supervisory authority, by:
   (a) a legally binding and enforceable instrument between public authorities or bodies;
   (b) binding corporate rules in accordance with Article 47;
   (c) standard data protection clauses adopted by the Commission in accordance with the examination procedure referred to in Article 93(2);
   (d) standard data protection clauses adopted by a supervisory authority and approved by the Commission pursuant to the examination procedure referred to in Article 93(2);
   (e) an approved code of conduct pursuant to Article 40 together with binding and enforceable commitments of the controller or processor in the third country to apply the appropriate safeguards, including as regards data subjects' rights; or
   (f) an approved certification mechanism pursuant to Article 42 together with binding and enforceable commitments of the controller or processor in the third country to apply the appropriate safeguards, including as regards data subjects' rights.

Annotation key: party, resource, action, constraint

Fig. 2. Paragraphs 1 and 2 excerpted from GDPR Article 46

```
<http://example.com/policy:gdpr-article46> a orcp:Set ;
    odrl:profile <http://example.com/odrl:profile:regulatory-compliance> ;
    orcp:permission
        [ odrl:action orcp:Transfer ;
          orcp:data orcp:PersonalData ;
          odrl:predicateConstraint
            [ odrl:or (
                [ odrl:leftOperand orcp:organisationType ;
                  odrl:operator odrl:isA ;
                  odrl:rightOperand orcp:InternationalOrganisation
                ]
                [ odrl:leftOperand orcp:recipientLocation ;
                  odrl:operator odrl:isA ;
                  odrl:rightOperand orcp:ThirdCountry
                ] )
            ] ;
          orcp:obligation
            [ odrl:predicateConstraint
                [ odrl:leftOperand orcp:dataSubjectProvisions ;
                  odrl:operator odrl:isA ;
                  odrl:rightOperand orcp:EnforceableDataSubjectRights
                ]
            ],
            [ odrl:predicateConstraint
                [ odrl:leftOperand orcp:dataSubjectProvisions ;
                  odrl:operator odrl:isA ;
                  odrl:rightOperand orcp:LegalRemediesForDataSubjects
                ]
            ],
            [ odrl:predicateConstraint
                [ odrl:leftOperand orcp:appropriateSafeguards ;
                  odrl:operator odrl:isAnyOf ;
                  odrl:rightOperand ( orcp:LegallyBindingEnforceableInstrument
                                      orcp:BindingCorporateRules
                                      orcp:StandardDataProtectionClauses
                                      orcp:ApprovedCodeOfConduct
                                      orcp:ApprovedCertificateMechanism )
                ]
            ]
        ] .
```

Listing 1. ODRL/TTL representation of paragraphs 1 and 2 of GDPR Article 46

Listing 2. ODRL/TTL request for permission to transfer personal data

```
<http://example.com/policy:bp-transfer> a orcp:Set ;
    odrl:profile <http://example.com/odrl:profile:regulatory-compliance> ;
    orcp:permission
        [ odrl:action orcp:Transfer ;
          orcp:data orcp:PersonalData ;
          orcp:responsibleParty orcp:Controller ;
          orcp:organisationType orcp:InternationalOrganisation ;
          orcp:sender <http://example.com/CompanyA_Ireland> ;
          orcp:recipient <http://example.com/CompanyA_US> ;
          orcp:recipientLocation orcp:ThirdCountry ;
          orcp:purpose orcp:PersonalRecommendations ;
          orcp:legalBasis orcp:Consent ;
          odrl:dataSubjectProvisions orcp:EnforceableDataSubjectRights ;
          odrl:dataSubjectProvisions orcp:LegalRemediesForDataSubjects
        ] .
```

`LegallyBindingEnforceableInstrument`, `BindingCorporateRules`, `StandardDataProtectionClauses`, `ApprovedCodeOfConduct`, or `ApprovedCertificateMechanism` are asserted in the company policy. While, the permission needed to execute a business process, presented in Listing 2, states that a `Controller` (party) who is an `InternationalOrganisation` wishes to perform a `Transfer` (action) of `PersonalData` (resource), from `CompanyA_Ireland` to `CompanyA_USA` in a `ThirdCountry`, for generating `PersonalRecommendations`, where the lawfulness for processing is `Consent`. In addition the company policy asserts that the company has `EnforceableDataSubjectRights` and `LegalRemediesForDataSubjects` in place within the company.

4 Compliance checking

The previous section describes and justifies the design of the ODRL policy compliance profile, which provides us with a means to represent a regulatory policy – such as fragments of GDPR – and to represent a company's particular business process, but it does not give the means to determine whether an implementation is compliant with the regulatory obligation. ODRL is not written using OWL-DL or even full OWL, it is purely RDF, and as such poses technical problems for some existing tools, for example the standard reasoners Pellet and Hermit cannot handle ODRL out of the box, while open-world reasoning over RDF has computational tractability issues. A practical, and common solution, is to switch from open-world to closed-world reasoning [25], which is the approach we take here, while also translating the ODRL policy representations into InstAL, which is subsequently compiled into an Answer Set Program [3], so that we may benefit from the computational capabilities of an Answer Set solver, in our case CLINGO [15]. We choose to use InstAL partly since it is a familiar tool for us, and partly because InstAL has been designed as a domain specific language for representation and reasoning about regulations, and thus offers modelling elements suitable for the task.

In the rest of this section, we provide a brief primer on InstAL, before describing how compliance check data is encoded, and how ODRL policies are translated to InstAL. We conclude the section with an example of a data transfer compliance check,

based on the content of Listing 2, which illustrates how the model detects conflicts between the process description and the regulatory policy.

4.1 Institutional Action Language

The Institutional Action Language (InstAL) is a domain-specific language (DSL) for writing models in terms of events and states, which translates to Answer Set Programming (ASP) for model evaluation under closed-world (non-monotonic) reasoning. Model state is expressed in terms of fluents – facts that are true if present and false if absent – which can be either inertial – true once initiated, until explicitly terminated – or non-inertial – whose presence is the result of a condition expressed over the model state. Inertial fluents model so-called institutional or normative facts, following the concepts of deontic logic [34], namely permission/prohibition and obligation, institutional power [20] and domain-specific facts. An InstAL specification has five kinds of rules: (i) x generates y: x is an event (action) and y is one or more events, whose generation can be conditional on the model state; (ii) x initiates y: x is an event and y is one or more fluents to add to the model state, subject to a condition as above; (iii) x terminates y: as initiates, but the fluents are deleted; (iv) x when y: x is a non-inertial fluent and y is a condition over the model state; and (v) initially x: x is one or more fluents that shall be part of the initial model state, again possibly subject to a condition.

4.2 Data representation

The approach taken for the purposes of this paper is to map the ODRL representation into three-element term fluents, reflecting the underlying RDF triples:

```
type Subject;
type Predicate;
type Object;
fluent triple(Subject,Predicate,Object);
```

which while simplistic, offers a uniform representation to which it is straightforward to translate. Consequently, we may represent the data for a given compliance-check, such as the transfer process description in Listing 2, as a set of triples:

```
triple(bp_transfer,action,transfer)
triple(bp_transfer,resource,personalData)
triple(bp_transfer,responsibleParty,controller)
triple(bp_transfer,organisationType,internationalOrganisation)
triple(bp_transfer,sender,companyA_Ireland)
triple(bp_transfer,recipient,companyA_US)
triple(bp_transfer,recipientLocation,thirdCountry)
triple(bp_transfer,purpose,personalRecommendations)
triple(bp_transfer,legalBasis,consent)
triple(bp_transfer,dataSubjectProvisions,enforceableDataSubjectRights)
triple(bp_transfer,dataSubjectProvisions,effectiveLegalRemedies)
```

4.3 Policy representation

The GDPR article fragments are represented as rules which determine the compliance of the process description depending on the data supplied (such as that given above), by

means of non-inertial fluent (when) rules, whose left-hand side is true, if the expression on the right-hand side is true, to determine whether permission is given or not. The translation is driven off the tree structure of the ODRL specification, so that where an article (such as the fragment of Article 46 in Listing 1) has several sub-terms, checking is broken into one condition for each term:

```
article46_body(Process) when
  article46_body_term1(Process),
  article46_body_term2(Process),
  article46_body_term3(Process),
  article46_body_term4(Process);
```

and the (non-inertial) fluent `article46` is true when the corresponding body is true:

```
noninertial fluent article46(Subject);
article46(Process) when
  triple(Process,resource,personalData),
  article46_body(Process);
```

while the process starts through the `_doCheck` event, if the action is a transfer:

```
_doCheck(Process) initiates
   permission(Process,article46)
   if article46(Process), triple(Process,action,transfer);
_doCheck(Process) initiates
   prohibition(Process,article46)
   if not article46(Process), triple(Process,action,transfer);
```

which initiates one of the (inertial) fluents `permission` or `prohibited`, corresponding to the permission on line 3 of Listing 1.

Predicates in an ODRL policy are either implied – a sequence of terms is a conjunction, so all the elements must be true (e.g. for the permission to hold as in Listing 1) – or explicit, introduced by a `predicateConstraint`, whose body may be a disjunction of terms, or a binary operator, being either `isA`, which tests a subclass relationship, or `isAnyOf`, which tests that at least one of the right operands holds.

`or`: The `or` operator is, as expected, defined to be true when either or both its sub-terms are true, which is achieved by defining two rules, one for each sub-term:

```
article46_body_term1(Process) when
  article46_term1_or(Process);
```

```
article46_term1_or(Process) when
   article46_organisationType(Process);
article46_term1_or(Process) when
   article46_recipientLocation(Process);
```

In addition there are the corresponding `supports` and `lacks` fluent rules for the explanation process (see below).

`isA`: There is no defined class hierarchy yet for the classes of the policy profile, so the InstAL model currently uses equality rather than a proper implementation of `isA`. Since the hierarchy would be defined in its entirety at the time of translation, the subclass relationship can be grounded and subsequently queried according to the needs of a given compliance request.

`isAnyOf`: The `isAnyOf` operator is defined to be true when at least one of the right hand sides is true. The encoding is verbose and predictable, reflecting the (tree) structure of the source ODRL. The usage in Article 46 specifies five alternatives,

but for the purposes of illustration we here show two cases. The first two fragments introduce the `isAnyOf` part of the article 46 encoding:

```
article46_body_term4(Process) when
  article46_term4_isAnyOf(Process);
```

```
article46_term4_isAnyOf(Process) when
   article46_appropriateSafeguards(Process);
```

This is followed by positive tests for the presence of each of the specified appropriate safeguards (of which we list two):

```
article46_appropriateSafeguards(Process) when
   triple(Process,appropriateSafeguards,bindingCorporateRules);
article46_appropriateSafeguards(Process) when
   triple(Process,appropriateSafeguards,legallyBindingEnforceableInstrument);
```

4.4 Explanation representation

The purpose of the model is firstly to establish whether the process description is GDPR-compliant (noting the relevant article), and secondly, if not, the cause of non-compliance. Consequently, we define the following fluents to capture such justifications:

```
type Article;
fluent permission(Subject,Article);
fluent prohibition(Subject,Article);
noninertial fluent supports(Subject,Article,Predicate,Object);
noninertial fluent lacks(Subject,Article,Predicate,Object);
```

Then, we address the matter of how the model reports the absence of data in the description. As can be seen below, the second term checks for the presence of enforceable data subject rights. If this term is true, then the description `supports` the term:

```
article46_body_term2(Process) when
  triple(Process,dataSubjectProvisions,enforceableDataSubjectRights);
supports(Process,article46,dataSubjectProvisions,
         enforceableDataSubjectRights) when
  article46_body_term2(Process),
  triple(Process,dataSubjectProvisions,enforceableDataSubjectRights);
```

if not, the description `lacks` such data, and the respective non-inertial fluents become true:

```
lacks(Process,article46,dataSubjectProvisions,
      enforceableDataSubjectRights) when
   applies(Process,article46),
   not supports(Process,article46,dataSubjectProvisions,
                enforceableDataSubjectRights);
```

Note that we check which article applies to the description in order not to report lacks that are not relevant to the description. Similarly, we can determine whether appropriate safeguards are supported and which are lacking (corresponding to the two cases listed under the discussion of isAnyOf above):

```
supports(Process,article46,appropriateSafeguards,X) when
   article46_appropriateSafeguards(Process),
   triple(Process,appropriateSafeguards,X);
lacks(Process,article46,appropriateSafeguards,bindingCorporateRules) when
```

```
   applies(Process,article46),
   not article46_appropriateSafeguards(Process);
lacks(Process,article46,appropriateSafeguards,
      legallyBindingEnforceableInstruments) when
   applies(Process,article46),
   not article46_appropriateSafeguards(Process);
```

The complete implementation is published through the InstAL repository[7].

Operationalization of the encoding Putting the above together, we arrive at a specification for a partial model of Articles 6 and 46 of the GDPR. This model, along with its grounding data can now be fed into the answer set solver, outputting either a confirmation of permission for the process description, along with the facts that support the permission, or a prohibition, along with the facts that provide partial support and the facts that are lacking.

4.5 Data transfer example

The transfer process described in ODRL in Listing 2 is represented in InstAL as listed in the previous section (called `bp_transfer`), identifying action, resource, responsible party, organization type, sender, recipient, recipient location, purpose , legal basis (for the transfer) and two kinds of data subject provisions. We have taken some syntactic liberties to accommodate InstAL's constraints on the naming of terms and literals, while still conveying the intention. As is conventional in logic programming languages, variables start with a capital letter, e.g. Party, while lower case are literals, e.g. `bp_transfer`.

Solving for this data, results in an answer set that includes the facts given below, in which we can see that the transfer is prohibited, since it lacks any of the specified appropriate safeguards:

```
prohibition(bp_transfer,article46)

supports(bp_transfer,article6,responsibleParty,controller)
supports(bp_transfer,article6,legalBasis,consent)
supports(bp_transfer,article46,organisationType,internationalOrganisation)
supports(bp_transfer,article46,dataSubjectProvisions,enforceableDataSubjectRights)
supports(bp_transfer,article46,dataSubjectProvisions,effectiveLegalRemedies)

lacks(bp_transfer,article46,appropriateSafeguards,standardProtectionClauses)
lacks(bp_transfer,article46,appropriateSafeguards,
      legallyBindingEnforceableInstruments)
lacks(bp_transfer,article46,appropriateSafeguards,bindingCorporateRules)
lacks(bp_transfer,article46,appropriateSafeguards,approvedCodeOfConduct)
lacks(bp_transfer,article46,appropriateSafeguards,approvedCertificateMechanism)
```

If we add the following data to the description:

```
triple(bp_transfer,appropriateSafeguards,bindingCorporateRules)
```

and solve again, the description is compliant according to the conditions of Article 46, thanks to the support of binding corporate rules (in addition to the same supports noted above):

[7] https://github.com/instsuite/instsuite.github.io/blob/master/gdpr.ial

```
supports(bp_transfer,article46,appropriateSafeguards,bindingCorporateRules)
permission(bp_transfer,article46)
```

5 Evaluation

In the evaluation of what we have presented in this paper, we assess the following aspects: (i) The adequacy of the modelling of the sample articles in ODRL; (ii) The adequacy of the mapping from ODRL to InstAL; and (iii) The performance of the InstAL model and corresponding ASP.

We start by assessing the suitability of ODRL for modelling legal requirements. The generalised ODRL Information Model appears to be quite similar to the ODRL Information Model, with the main changes relating to the replacement of `Asset` class with the more general `Resource` class, the conversion of the `Constraint` class to an abstract class and the inclusion of a single `PredicateConstraint` subclass, the replacement of the `Duty` class with the `Obligation` class, and the inclusion of a new `Dispensation` class. We deliberately excluded legal concepts from this generalised ODRL Information Model as we believe it can serve as the foundations for other ODRL profiles, for instance to express usage policies, social norms, and privacy policies. The profile itself and the corresponding ontology are quite different from the original ontology, due to the need to change the `range` of several classes to consider the `Resource` class as opposed to the `Asset` class. Additionally, we were often forced to define new vocabulary rather than reuse the existing ODRL vocabulary as the `skos:definition`, where skos denotes the Simple Knowledge Organization System ontology, was too specific/limited for our use. As for the modelling of the text of the GDPR using the proposed ODRL Regulatory Compliance Profile, rather than opting for a one to one modelling of the text as RDF, we chose instead to only model things that can be checked automatically. For instance, to enable companies to attest that certain provisions and safeguards exist, rather than actually carrying out, as part of the compliance checking process, the verification that such things exist. Here we assume we are dealing with companies that want to demonstrate compliance, and are using the compliance checking as a form of guidance with respect to their legal obligations, or as a means to verify that changes to business processes are still legally compliant. In this paper we assessed the suitability of the proposed ODRL Regulatory Compliance Profile using two (key) fragments of the General Data Protect Regulation, namely Articles 6 (Lawfulness of processing) and Articles 46 (Transfers subject to appropriate safeguards). Considering the extensible nature of RDF, the profile can easily be extended with additional vocabulary and constraints in order to extend this work to not only include other articles of the GDPR but also to model other legislation.

The second issue is the adequacy of the mapping from ODRL to InstAL. As illustrated in section 4, we took a direct approach that effectively replicates the data in the ODRL model as three element terms, while turning most of the internal nodes of the document tree into non-inertial fluents whose value is determined by the corresponding child nodes. This strategy has the benefits of (i) being able to associate `supports` and `lacks` rules with internal nodes where it is desirable to gather data about the justification or otherwise of the compliance result (ii) making the code generation highly

localized in terms of dependence on data in the source document. The model contains just the one action, which is used to invoke a compliance check on a given `Subject`, such as `bp_transfer` in the example. As we note in the next aspect of the discussion on performance, the translation could be differently structured to reduce grounding costs at the expense of verbosity and legibility. There are four other elements of the ODRL information model for which to account, namely the different types of rules (permission, prohibition, obligation and dispensation). The working example of Article 46 contains only permission and obligation, so we discuss those first. The `permission` translates to the inertial fluent that is initiated as a result of compliance with the `article46` rule, while the `prohibition` generation is an artefact of the operationalization of the compliance check to indicate the reporting of compliance failure. The obligation element of the ODRL information model provides syntactic structure and semantic annotation, to indicate that its subterms need to be checked, aligning with the legal interpretation of the notion of obligation, but has no actionable semantics of itself in respect of compliance checking, hence the translation skips over obligation to process its children. The same is effectively true for prohibition and dispensation, except that the former introduces a negation and the latter a side-condition on the child terms of the respective nodes.

The third issue we consider is the performance of the policy model. Answer Set Programming operates in two stages: grounding i.e. replacing variables with grounded terms, and solving, i.e. computing the answer sets. Both have high complexity in general that can often be tamed in practice. The solve step in this case is essentially polynomial because there is only one answer set and the program is stratified (see [16] for details). The grounding cost is, generally speaking, dependent on the number of distinct variables appearing in each rule and the number of alternative values a given variable might take. The encoding strategy used here is, as noted earlier, simplistic, and in consequence the state space size is a function of the number of combinations arising from the number of subject, predicate and object values, but since there is only one subject in each case (the example identifier, e.g. `bp_transfer`), this reduces to the product of the number of predicate and object values. The typing used in InstAL and the smart grounding processes used by the *gringo* grounder [14], reduces the search space significantly. This could be reduced further if the predicate could carry type information and hence define the range, restricting the values that the object might take. For example, instead of writing `triple(Subject,Predicate,Object)`, the predicate can be encoded in the term: `predicate(Subject,X)`, where `X` is the type denoting the set of object values in the range of `predicate`, which would be quite manageable in the context of a more sophisticated translation, although the resulting code might not be so human legible. As it stands, the grounding (and solving) costs are negligible, but could benefit from reconsideration given larger state spaces. Alternatively, we could consider enhancing InstAL's type system to support subsumption, which would allow the pre-grounding of most rules with the abstract types. The use of ASP allows us to guarantee the soundness and completeness of our approach (under the assumption that the modelling was correct). In other words, the system is always able to say if the business process is compliant or not and the answer is correct with respect to the modelling.

6 Conclusion

In this paper, we introduced our ODRL Regulatory Compliance Profile, which can be used to model both regulatory requirements via nested permissions, prohibitions, obligations, and dispensations, and business policies via discrete permissions that are needed to execute a compliant business process. We subsequently demonstrated how such policies can be translated into Answer Set Programming such that it is possible to automatically check compliance, and provide, if necessary, a basic explanation why compliance is not achieved.

We are currently working together with our industry partners to extend the proposed ODRL Regulatory Compliance Profile to cater for the representation of more detailed business policies, that specify which data are processed, for what purpose, where the processing takes places, for how long the data will be stored, and with whom the data is shared. Such information is needed in order to check compliance with a broader set of Articles from the GDPR. One of the primary challenges involves bridging the gap between very abstract legal requirements and the very detailed business policies. Here we plan to exploit class and property hierarchies by expanding our modelling and compliance checking to support subsumption based reasoning.

Further future work includes: (i) demonstrating how a broader set of articles can be modelled using our ODRL Regulatory Compliance Profile and automatically translated from ODRL policies into InstAL rules and Vice versa. (ii) providing a formal semantics for our ODRL Regulatory Compliance Profile, and adapting both our ontology and compliance checking to cater for legislative opening clauses that require reasoning over multiple pieces of legislation; and (iii) exploring how the ODRL policy description might form part of a description for a policy reasoning service [28] and thereby facilitate (semantic) discovery and use of such services.

Acknowledgements. This work was supported in part by the European Union's Horizon 2020 research and innovation programme under grant 731601 and by JSPS Grant-in-Aid for Scientific Research(S), Grant Number 17H06103. We would like to thank the SPECIAL project consortium for their feedback on the proposed profile.

References

1. S. Agarwal, S. Steyskal, F. Antunovic, and S. Kirrane. Legislative compliance assessment: Framework, model and GDPR instantiation. In *Proceedings of the Annual Privacy Forum (APF 2018)*, 2018.
2. T. Athan, H. Boley, G. Governatori, M. Palmirani, A. Paschke, and A. Z. Wyner. Oasis LegalRuleML. In *ICAIL*, volume 13, pages 3–12, 2013.
3. C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. CUP, 2003.
4. C. Bartolini, R. Muthuri, and C. Santos. Using ontologies to model data protection requirements in workflows. In *JSAI International Symposium on Artificial Intelligence*, 2015.

5. A. Boer, R. Hoekstra, R. Winkels, T. Van Engers, and F. Willaert. Metalex: Legislation in XML. *Legal Knowledge and Information Systems (Jurix 2002)*, pages 1–10, 2002.
6. A. Boer, R. Winkels, and F. Vitali. Metalex XML and the legal knowledge interchange format. In *Computable models of the law*, pages 21–41. Springer, 2008.
7. H. Boley, A. Paschke, and O. Shafiq. Ruleml 1.0: the overarching specification of web rules. In *International Workshop on Rules and Rule Markup Languages for the Semantic Web*, pages 162–178. Springer, 2010.
8. P. A. Bonatti and D. Olmedilla. Rule-based policy representation and reasoning for the semantic web. In *Proceedings of the Third International Summer School Conference on Reasoning Web*, 2007.
9. J. M. Bradshaw. *Software agents*. MIT press, 1997.
10. O. Cliffe, M. De Vos, and J. Padget. Answer set programming for representing and reasoning about virtual institutions. In K. Inoue, K. Satoh, and F. Toni, editors, *CLIMA VII*, volume 4371 of *Lecture Notes in Computer Science*, pages 60–79. Springer, 2006. ISBN 978-3-540-69618-6.
11. J. Dimyadi, P. Pauwels, and R. Amor. Modelling and accessing regulatory knowledge for computer-assisted compliance audit. 2016.
12. N. Fornara and M. Colombetti. Operational semantics of an extension of ODRL able to express obligations. In F. Belardinelli and E. Argente, editors, *Multi-Agent Systems and Agreement Technologies - 15th European Conference, EUMAS 2017, and 5th International Conference, AT 2017, Évry, France, December 14-15, 2017, Revised Selected Papers*, volume 10767 of *Lecture Notes in Computer Science*, pages 172–186. Springer, 2017. ISBN 978-3-030-01712-5. https://doi.org/10.1007/978-3-030-01713-2_13.
13. N. Fornara, A. Chiappa, and M. Colombetti. Using semantic web technologies and production rules for reasoning on obligations and permissions. In M. Lujak, editor, *Agreement Technologies - 6th International Conference, AT 2018, Bergen, Norway, December 6-7, 2018, Revised Selected Papers*, volume 11327 of *Lecture Notes in Computer Science*, pages 49–63. Springer, 2018. ISBN 978-3-030-17293-0. https://doi.org/10.1007/978-3-030-17294-7_4.
14. M. Gebser, R. Kaminski, A. König, and T. Schaub. Advances in gringo series 3. In *LPNMR*, volume 6645 of *Lecture Notes in Computer Science*, pages 345–351. Springer, 2011.
15. M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub. Clingo = ASP + control: Preliminary report. *CoRR*, abs/1405.3694, 2014.
16. M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In R. A. Kowalski and K. A. Bowen, editors, *Logic Programming, Proceedings of the Fifth International Conference and Symposium, Seattle, Washington, USA, August 15-19, 1988 (2 Volumes)*, pages 1070–1080. MIT Press, 1988. ISBN 0-262-61056-6.
17. M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3-4):365–386, 1991.
18. 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.

19. Information Commissioner's Office (ICO) UK. Getting ready for the GDPR, 2017. https://ico.org.uk/for-organisations/resources-and-support/data-protection-self-assessment/getting-ready-for-the-gdpr, visited on 1/5/2019.
20. A. Jones and M. Sergot. A formal characterisation of institutionalised power. *Logic Journal of IGPL*, 4(3):427–443, 1996.
21. L. Kagal and T. Finin. A policy language for a pervasive computing environment. In *Proceedings POLICY 2003. IEEE 4th International Workshop on Policies for Distributed Systems and Networks*, 2003.
22. H.-P. Lam and M. Hashmi. Enabling reasoning with LegalRuleML. *Theory and Practice of Logic Programming*, 19(1):1–26, 2019.
23. T. Li, T. Balke, M. D. Vos, J. A. Padget, and K. Satoh. A model-based approach to the automatic revision of secondary legislation. In E. Francesconi and B. Verheij, editors, *International Conference on Artificial Intelligence and Law, ICAIL '13, Rome, Italy, June 10-14, 2013*, pages 202–206. ACM, 2013. ISBN 978-1-4503-2080-1. https://doi.org/10.1145/2514601.2514627.
24. Microsoft Trust Center. Detailed GDPR Assessment, 2017. http://aka.ms/gdprdetailedassessment, visited on 1/5/2019.
25. B. Motik, I. Horrocks, R. Rosati, and U. Sattler. Can owl and logic programming live together happily ever after? In *International semantic web conference*, pages 501–514. Springer, 2006.
26. Nymity. GDPR Compliance Toolkit. https://www.nymity.com/gdpr-toolkit.aspx, visited on 1/5/2019.
27. J. Padget, E. ElDeen Elakehal, T. Li, and M. De Vos. *InstAL: An Institutional Action Language*, pages 101–124. Springer, 2016. ISBN 978-3-319-33570-4. https://doi.org/10.1007/978-3-319-33570-4_6.
28. J. Padget, M. D. Vos, and C. A. Page. Deontic sensors. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 475–481. International Joint Conferences on Artificial Intelligence Organization, 7 2018. https://doi.org/10.24963/ijcai.2018/66.
29. M. Palmirani, G. Governatori, A. Rotolo, S. Tabet, H. Boley, and A. Paschke. LegalRuleML: XML-based rules and norms. In *International Workshop on Rules and Rule Markup Languages for the Semantic Web*, pages 298–312. Springer, 2011.
30. O. Panasiuk, S. Steyskal, G. Havur, A. Fensel, and S. Kirrane. Modeling and reasoning over data licenses. In *European Semantic Web Conference*, pages 218–222. Springer, 2018.
31. H. J. Pandit, K. Fatema, D. O'Sullivan, and D. Lewis. Gdprtext-gdpr as a linked data resource. In *European Semantic Web Conference*, pages 481–495. Springer, 2018.
32. S. Steyskal and S. Kirrane. If you can't enforce it, contract it: Enforceability in policy-driven (linked) data markets. In *SEMANTiCS (Posters & Demos)*, 2015.
33. S. Steyskal and A. Polleres. Defining expressive access policies for linked data using the ODRL ontology 2.0. In *Proceedings of the 10th International Conference on Semantic Systems*, 2014.
34. G. von Wright. Deontic logic. *Mind*, 60(237):1–15, 1951. ISSN 00264423, 14602113.