D5.2 Policy-based Language Data Management

Author(s): Víctor Rodríguez-Doncel, Pablo Calleja
Date: 31 December 2020

This project received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825182. The information and views set out in this publication are those of the author(s) and do not necessarily reflect the official opinion of the European Union.

H2020-ICT-29b
Grant Agreement No. 825182
Prêt-à-LLOD - Ready-to-use Multilingual Linked Language Data for Knowledge Services across Sectors

D5.2. Policy-based Language Data Management

Deliverable Number: D5.2
Type: ORDP
Dissemination Level: PU
Delivery Date: 31 December 2020
Version: 1
Author(s): Víctor Rodríguez-Doncel, Pablo Calleja.

Document History

Version Date	Changes	Authors
September 2020	initial version	Víctor Rodríguez-Doncel, Pablo Calleja
20th December 2020	pre-final version	Víctor Rodríguez-Doncel, Pablo Calleja
29th December 2020	internal review	Thierry Declerck (DFKI)

Table of Contents

List of Tables

List of Figures

List of Acronyms

API	Application Program Interface
DMP	Data Management Plan
DOI	Document Object Identifier
ELG	European Language Grid
FAIR	Findable, Accessible, Interoperable and Reusable
GDPR	General Data Protection Regulation
IPFS	Interplanetary File System
IPTC	International Press Telecommunication Council
ODRL	Open Digital Rights Language
ODRL-LR	Open Digital Rights Language for Language Resources
OSRD	Ontology Specification Requirements Document
PDDM	Policy Driven Data Manager
REL	Rights Expression Language
RDF	Resource Description Framework
UPM	Universidad Politécnica de Madrid

D5.2 Policy-based Language Data Management

Executive summary

Language resources ready to be exploited need a proper licence and policy information management. This deliverable describes the data model, data assets and REST Services to manage the rights and provenance information related to the language resources. A portal has been published where all the information is presented (`https://pddm-pal.oeg.fi.upm.es/`). Source code has been made publicly available, as well as the mapping of about 200 licences into a machine-readable form based on the Open Digital Rights Language (ODRL). Integration with parts of Linghub shall be reported in D5.3 "Prêt-à-LLOD Language Resource Discover Portal", in June 2020 -Linghub is a portal providing harmonized access to language resource metadata, and a central resource of Prêt-à-LLOD [31].

1. Introduction

Prêt-à-LLOD aims at making language resources «ready to be used». In non-academic scenarios, this implies making language resources usable from a legal point of view, and granting that legal compliance is not going to be a hurdle. This deliverable describes the works done towards lowering the intellectual property-related barriers that might prevent language resources to be promptly consumed. Resources that are not properly authored and licenced cannot be consumed, and compliance related tools need to be supplied to ease a lawful resource consumption.

Having this raw idea in mind, a more systematic gathering of specific requirements was made and a collection of use cases was described. From these use cases, an ontology requirements specification document (OSRD) was drafted and specific functional requirements were collected (for an HTTP REST API).

This deliverable will describe **data models**, **data assets** and licence-related **services** towards for facilitating the automated consumption of language resources. All these results are presented online through a portal:

Policy-Driven Data Manager
https://pddm-pal.oeg.fi.upm.es/

Data models describe rights information (the intellectual property status, the rights-holders and the policies of uses and transactions). The provenance and rights information is represented using the W3C recommendations for provenance (PROV-O [9]) and policies (ODRL, Open Digital Rights Language [10-11]). The data models are realized through an ontology, the ODRL-LR, ODRL profile for language resources (http://purl.org/odrl-lr).

As data asset, this deliverable describes a collection of licences in a machine-readable form, a dataset that helps policies to be reused (fighting against the problem known as «licence proliferation») and favouring automated licence reasoning.

Finally, a collection of services has been implemented for the integration with other applications. The work presented here will serve other Prêt-à-LLOD technologies for discovering datasets and services with an explicit and automated treatment of legal constraints. Deliverable D5.3 will report on the search-by-license across repositories in T5.3, to be implemented by the Linghub portal.

2. Background

2.1. Related work

Representation of rights and policies

ODRL, specified first in 2000 as an XML language, has been adopted to declare rights and policies in different industries: mobile devices, ebooks, news and others. More recently, the specification of the 2.2 version has extended the language and now policies are represented in RDF, having as data model an ontology. ODRL has been published as a W3C Recommendation and it is in use for several domains, such as news, financial information or ebooks. ODRL can be used both as a rights expression language and as a policy language for language resources, declaring non-enforceable rights as well as driving a conditional access system [12].

In ODRL, the following sentences can be represented: 'Anyone can copy and derive my ontology, but not make commercial use of it' or 'Mary can access my RDF dataset before 2020'. Note that violations of the first rule cannot be prevented (nor automatically detected), but violations of the second rule can be made a bit harder to make: a computer system can serve the dataset only if Mary has authenticated and the access time is before 2020. ODRL had been suggested as a good candidate to declare the rights for Linked Data [13][14].

Datasets of policies

The work done in Prêt-â-LLOD is based on the work of Rodríguez-Doncel et al. [26], who published a first version of a dataset of licenses -but crafted for the previous version of the ODRL standard. Since then, other repositories with structured information have appeared.

The most widespread repository of licences is https://tldrlegal.com/, where there is a JSON structure for every licence. However, its representation is not in RDF and information is proprietary of the private company FOSSA Inc., not being the dataset publicly licenced[1].

[1] We read in https://tldrlegal.com/pages/legal: «Other than as expressly set forth in these Terms, you may not copy, modify, publish, transmit, upload, participate in the transfer or sale of, reproduce, create derivative works, distribute, perform, or display, any of the Content, or Sites in whole or in part without written permission from TLDR.»

Finally, the DALICC project has also published a collection of licences[2], but the structured version of the licences is not available.

Reasoning with policies

One of the earliest works towards calculating the compatibility of licences is the theoretical work based on defeasible deontic logic made by Governatori et al. [27] in 2013.

The use of ODRL has been made as well to calculate the arithmetic of composition of licences, as in the DALICC project [22]. DALICC is an Austrian project much related to our efforts. DALICC stands for Data Licences Clearing Center (FFG, https://dalicc.net/), and it has developed «a software framework that supports legal experts, innovation managers and application developers in the legally secure reutilization of third party data sources». Within the same project, query based access control has also been described by Pellegrini et al. [20].

One of the best described authorization system based on ODRL is that of Governatori [23], which translates the policies into a defeasible logic language called Formal Contract Language (FCL) [24]. It was applied in the area of Social Networks, and it was able to manage conflicts among rules, violations and temporal reasoning.

3. Scenarios

Usage scenarios were drafted in order to gather requirements. These scenarios were produced in a collaborative manner. The requirements derived from the first use case have been implemented within the timeframe of the task T5.2, however, the requirements for the second use case, related to integration with Linghub, could not be developed. The integration efforts will be reported in "D5.3 Prêt-à-LLOD Language Resource Discover Portal", due by June 2021.

3.1 Scenario #1 Management of rights information in Linghub

Alice has created a new language resource: a dictionary klingon-swahili. She wants the resource to be used under certain restrictions (e.g. non-commercial use but available for research). Bob is a klingon-freak and wants to use Alice's resource.

Use case 1.1 Registration of a resource rich in provenance

Alice publishes a resource. She wants her resource to be properly licensed and with provenance.

Steps

1. Alice creates her resource deriving her work from an existing work, and describes it using Prêt-à-LLOD vocabularies with the Linghub create-resource webpage.
2. Alice describes the provenance of the used resources.

Comment. This is the simplest use case and the most important to be supported by Prêt-à-LLOD: a resource is created and the provenance and licensing information is registered.

[2] https://dalicc.net/license-library

Use case 1.2. Simple request for a licensed resource

After Bob has made a search, he must accept the terms and conditions in Alice's resource in order to get access to it.

Steps

1. Bob searches a resource in Linghub and finds Alice's dictionary.
2. Bob is informed on the terms and conditions in human language, and he accepts.
3. Bob accepts and is redirected to the machine hosting the resource with an access token (with a legal proof that he accepted that terms).

Comment. The efforts of Alice by properly registering her resource are now paid off.

Use case 1.3. Integration of a licensed resource

Bob extends Alice's dictionary and uploads it to Linghub.

Steps

1. Alice creates her resource
2. Bob obtains the resource (UC1.1) and then uploads a new resource in Linghub. He declares he has used Alice's resource.
3. Linghub automatically limits the range of policies Bob can use (e.g. maybe Alice's policy had a share-alike).

Comment. Having a single interface point for the user is a nice to-have. Avoiding multiplicity of interfaces, Linghub acts as a single user interface point for the user.

Use case 1.4. Integration of multiple licensed resources

Alice publishes a resource and Bob integrates it together with Carol's resource.

Steps

1. Alice creates a resource in Linghub with licensing restrictions A.
2. Carol creates a resource in Linghub with licensing restrictions C.
3. Bob wants to use Alice's and Carol's resources and tells Linghub about it.
4. Linghub calculates (via Policy Server) whether both resources can be integrated and under which conditions, informing Bob. Continue in Use Case 1.1.

Comment. The Prêt-à-LLOD project «will provide tools to combine language services and resources into complex pipelines by use of semantic technologies». This automated combination of resources requires automated reasoning on the permissions and licences that are involved.

Use case 1.5. Smart contracts

Bob purchases an Alice to resource. The smart contract is stored and executed at Linghub.

Steps

1. Alice offers a resource in Linghub for €10.
2. Bob agrees with the price and makes the payment *out* of Linghub

3. The policy server keeps a copy of the contract and authorizes further access from Alice.

Comment. There are important assumptions: we use Linghub user interface as the UI with the user. This use case is pending to be sanctioned by existing pilots.

2.2 Scenario #2 Service integration

This scenario comprises two use cases:

Use case 2.1. Integration with Linghub

Dan searches by licence for a certain language resource: he is looking for dictionaries that he can extend and republish in a commercial environment.

Steps

1. Resources are registered in Linghub together with a licence, as in “Use Case 1.1”
2. Dan opens the Linghub portal and browses after having added a filter in the faceted navigation: “for commercial use” and “with modifications”.
   This scenario requires a service where search by licence can be made.

Use case 2.2. Preservation

Alice created and registered a very nice resource (Use Case 1.1) 10 years ago, declaring her authorship over the resource. After 10 years the Prêt-à-LLOD servers are down, Linghub does not respond and a usurper called Francis has appropriated Alice’s resource.

Steps

1. Alice finds out Francis has published her resource with his name.
2. Alice remembers about something Prêt-à-LLOD did, preserving authorship in the Interplanetary File System (IPFS) network. She keeps a token she got when registering her resource.
3. Alice uses an IPFS gateway to find the licence that proves she registered her resource 10 years ago. Because information to be registered is not massive, other IPFS nodes could easily replicate and echo this information.
4. A court admits the timestamp-signed document, whose hash matches her claim. The court fails in her favour.

Comment. Prêt-à-LLOD proves useful even after its lifetime by using IPFS. IPFS is a peeer to peer network, a «distributed system for storing and accessing files, websites, applications, and data», where its are identified by their hash. Two properties are of interest for Prêt-à-LLOD: first, as a P2P network, IPFS files are soon distributed in multiple computers and preservation is more likely than with a single server. Second, the hash makes impossible the content to be tampered for the same reference. These desired properties make licenses to preserve integrity, makes them public (unless encrypted), and makes them perpetual.

Use case 2.3. Integration with ELG and other external platform

Elena wants to sell her proudly created language resource. Because Prêt-à-LLOD servers are not much visited, she wished her resource to gain a broader audience.

Steps

1. Elena uploads a resource for sale in Linghub.
2. Linghub contacts other platforms and disseminates the offer.

Comment. The business models and the technologies are not yet defined -this use case is only sketched, but the need is manifested here (read more in "D5.3 Prêt-à-LLOD Language Resource Discover Portal").

4. Requirements

This section includes the requirements towards making an ontology to serve as data model and the requirements for the services.

4.1 Requirements for the policy-based data management services

To provide the functionalities that fulfil the scenarios and uses cases presented in the previous section, a set of requirements have been defined. There are two main categories for the classification of requirements: functional and non-functional.

Functional Requirements

- **R1. Authorship Register**. Content creators or rights holder MUST be able to assert their ownership of the rights of a certain resource. As in any other IP registry, this claim SHAN'T be verified. Derivative contents shall be able to be declared as such together with the original works they are based upon. This requirement is supported by Use Case 1.1 and 2.2
- **R2. Authorship Query**. Content creators or rights holders MUST be able to obtain a proof of the authorship registration (e.g. a signed timestamp). This requirement is supported by Use Case 1.2 and Use Case 2.2
- **R3. Licence Register**. Content creators, rights holders or authorized parties MUST be able to declare a licence or policy for a certain content item. This use case is supported by Use Case 1.1
- **R4. Licence Query**. Whenever public, anybody MUST be able to obtain the licence associated to a certain resource. Whenever non-public, only interested parties MUST get access to such licence. This use case is supported by Use Case 1.4 and 2.1.
- **R5. Licence combination arithmetic**. The data manager MUST be able to determine the subset of licences that can be used to licence content resulting from the aggregation of heterogeneously licenced works. This Use Case is supported by Use Case 1.4.
- **R6. Provenance Retrieval**. Whenever public, anybody MUST be able to determine the provenance chain. This Use Case is supported by Use Case 1.4.
- **R7. Preservation**. The link Authorship-ContentHash MAY be made pervasive, stable and immutable by its addition to an IPFS file. This requirement derives from Use Case 2.2.

Non-functional Requirements

- **NFR1. User identification and authentication.** The data manager SHOULD be identified/authenticated with WebID, OAUTH or any other authentication strategy - user/password suffices. Other platforms that use registration could be explored as user identification and authentication such as DataHub (datahub.io)
- **NFR2. Standards**. Policies MUST be represented using standard technologies (e.g. W3C ODRL Recommendations).

4.2 Ontology Specification Requirements Document

The specification of the ontology to serve as data model has been made using the OSRD document template from the NEON ontology development methodology guidelines [28].

Table 1. Ontology Specification Requirement Document for the ODRL-LR Ontology

	Ontology Requirements Specification Document ODRL Profile for Language Resources
	Purpose
	The purpose of this ontology is to serve as a data model for the policies to be used in the Prêt-à-LLOD project.
	Scope
	The scope of the ontology is given by the existing licences and policies applied to language resources: this ontology should represent common licences and policies, as those found in the MetaShare repository.
	Implementation Language
	OWL extending the ODRL as an ODRL profile.
	Intended End-Users
	This ontology will be used by the services of the PDDM and Linghub
	Intended Uses
	Define the permitted, prohibited, or obliged actions and restrictions associated to licence of language resources.
	Ontology Requirements
a.	**Non-Functional Requirements**
	● The ontology shall be identified with a permanent URI. ● The ontology must be online and publicly accessible without restrictions ● The ontology must be described using the standard practices (Peroni's

	LODE [29]) ● The ontology must be consistent, as there will be reasoning with it.
b.	**Functional Requirements: Lists or tables of requirements written as Competency Questions and sentences**
	CQ1. What is permitted, obligated, or prohibited? *CQ2 What are the most common actions described in language resource policies?* *CQ3. What are the most common restrictions present in policies for language resources?* *CQ4. Which licence prevails over which in case two policies apply to the same resource?* *CQ5. Which are the most common licences and policies associated to language resources?* *CQ6. Which types of licences are there?*
	Pre-Glossary of Terms (optional)
a.	**Terms from Competency Questions**
	Derived from CQ1: permission, obligation, prohibition **Derived from CQ2:** those found in ODRL vocab plus: localize, extractVectorModel, speech2text **Derived from CQ3:** those found in ODRL vocab plus: allowedLanguage, availabily, user nature **Derived from CQ5**: those found in the namespace http://purl.org/NET/ms-rights# **Derived from CQ6**: licenceType
b.	**Terms from Answers**
	Derived from CQ3: research, education, commercialInstitution, educationalInstitution **Derived from CQ6**: MetaShareLicence, CustomLicence, ClarinLicence
c.	**Objects**
	The ontology must represent the golden three of the deontic logic: prohibitions, obligations and permissions The ontology must define actions commonly associated to language resources The ontology must define restrictions commonly present in language resource policies.

5. Data Model

The Prêt-à-LLOD data model is materialized through the ontology ODRL-LR (ODRL profile for language resources), whose URI is: `http://purl.org/odrl-dl`

5.1 Introduction

The ODRL for Language Resources (ODRL-LR) is an ontology and data model for representing policies and licences usable for language resources

ODRL is a policy expression language with more than 20 years of existence. In its most recent form, and after being an XML standard for long, policies are represented as RDF documents supported by an ODRL Ontology. According to the spec, ODRL "provides a flexible and interoperable ODRL information model [11], ODRL vocabulary [10], and encoding mechanisms for representing statements about the usage of content and services" --indeed also for language resources. The ODRL policy language can be extended by means of profiles and this document specifies the profile for language resources, providing the appropriate vocabulary for representing commonly used licences and policies for language resources.

In the framework of the project, which aims at producing ready-to-use language resources, a number of licences and algorithms have been published for the easiest policy-driven data management. Prêt-à-LLOD is a project funded by the H2020 programme.

This ODRL for Language Resources (ODRL-LR) is a profile done in accordance to the ODRL Profile Best Practices, dated in December 2020, which basically prescribes the use of (a) the W3C Profile Vocabulary (which is a W3C Note) and (b) the list of elements that can be extended and how. Other profiles have been specified, such as the profile for IPTC[3] news and the financial Market Data Profile for ODRL published by the Rights Automation for Market Data W3C Community Group. The ODRL-LR presented in this document departs from the MetaShare Rights ontology published online [4].

The ODRL policies are represented as RDF documents, usually serialized as Turtle or as JSON-LD. This document provides everything necessary to learn how to use this data model through a Cookbook and the detailed description of the ontology.

Advantages of using machine-readable licences are:

- Search-by-license is enabled. Information systems can implement search algorithms where the user can specify the desired licence of the resource.
- Reasoning is enabled. Certain logic can be implemented to automatically determine whether two resources can be combined or not.
- Resource management is easier, as computer programs can remind about licences about to expire, etc.

5.2 Ontology description

Versioning, storage and publication

The ontology has been published with a permanent URI, `http://purl.org/odrl-lr`, and its development has been made manually and with Protégé 5.0[4].

[3] International Press Telecommunications Council (IPTC), https://iptc.org/

[4] https://protege.stanford.edu/

The development has been made transparent, as different versions are traceable with a source code control system[5]. The master serialization is Turtle, the other serializations are derived thereof.

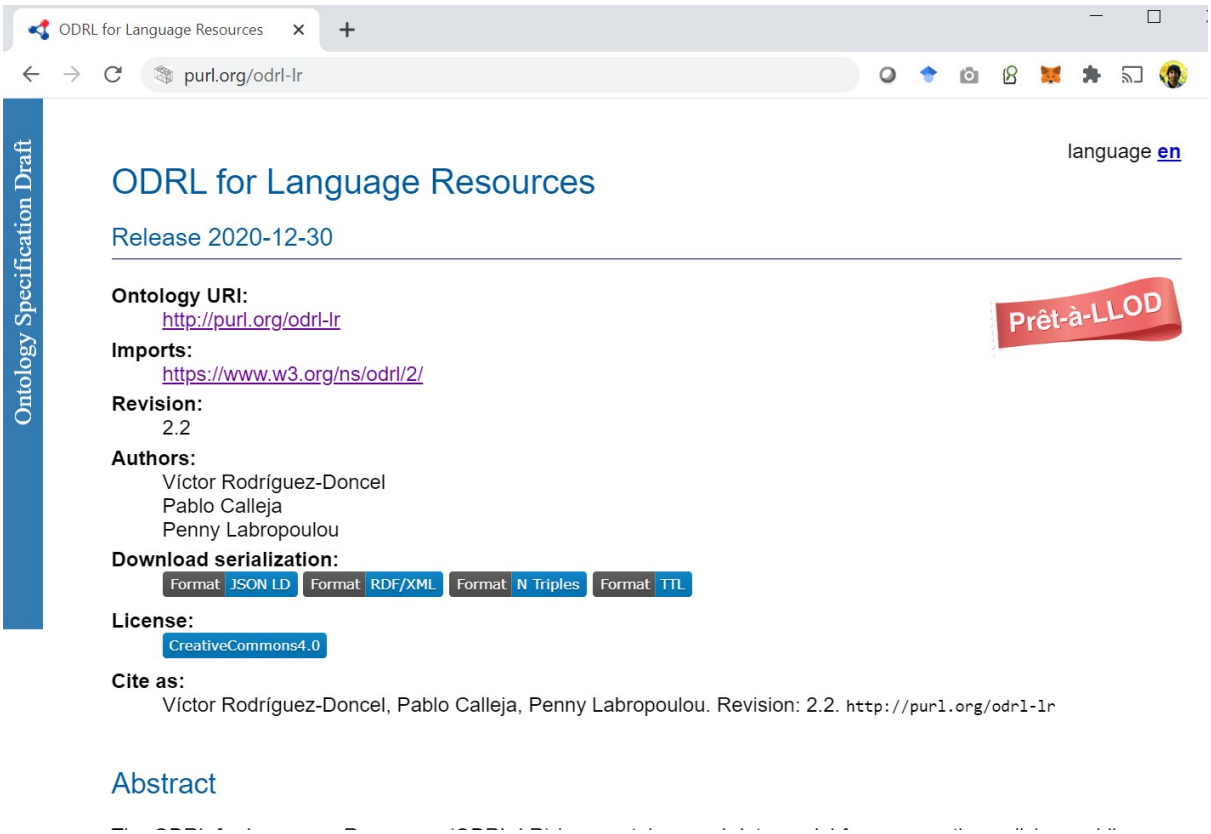

Figure 1. Screenshot of the ODRL for Language Resources ontology description

Statistics

Although the work is not given as finished, as of December 2020 the ontology consisted of 631 axioms, with 23 properties and 111 individuals, plus 25 annotation properties. The ontology is consistent and of very simple complexity (ALH(D)).

[5] https://gitlab.com/vroddon/www/-/blob/master/static/odrl-lr/odrl-lr.ttl

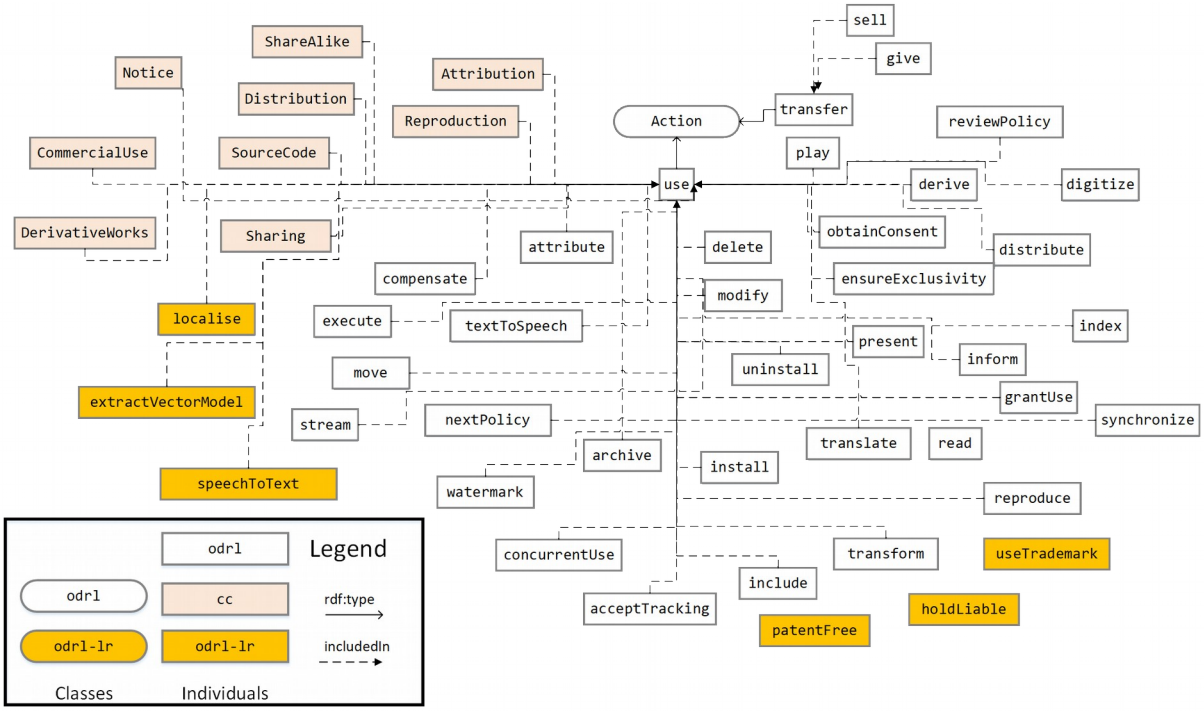

Figure 2. ODRL-LR Actions that can be used in the policies.

Quality
The ontology has been tested with Oops searching for pitfalls [30] –no major pitfall was found at the time of writing this report.

5.3. Cookbook

This section describes how to use the data model in practical stances. The same information has been reproduced in the online portal of the PDDM for the convenience of the user.

1. How to declare a licence in HTML in the simplest manner.
Without this specification, if you are describing your page in HTML you may simply write:

```
<p>My Dictionary is published under a  <a rel="license" href="http://www.meta-share.org/assets/pdf/META-SHARE_Commercial_NoRedistribution_v1.0.pdf">Metashare Commercial No Redistribution </a> license. </p>
```

Please note that we have micromarked (in red) that the link is a licence --this is the best way that can be done in a simple way.

2. How to represent a licence in a machine-readable form
A machine-readable licence is a data structure with the key information present in a licence so that machines can easily process them. An example of machine-readable licence can be the GNU GPL, which can be downloaded here: https://www.gnu.org/licenses/gpl-2.0.rdf:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<rdf:RDF
  xmlns:cc='http://creativecommons.org/ns#'
  xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
  xmlns:dcq='http://purl.org/dc/terms/'>
  <cc:License rdf:about="http://gnu.org/licenses/gpl-2.0.html">
    <dcq:hasVersion>2.0</dcq:hasVersion>
    <dcq:identifier>GNU GPL</dcq:identifier>
    <cc:requires rdf:resource="http://creativecommons.org/ns#Copyleft"/>
    <cc:requires rdf:resource="http://creativecommons.org/ns#SourceCode"/>
    <cc:requires rdf:resource="http://creativecommons.org/ns#Notice"/>
    <cc:permits rdf:resource="http://creativecommons.org/ns#Distribution"/>
    <cc:permits rdf:resource="http://creativecommons.org/ns#Reproduction"/>
    <cc:permits rdf:resource="http://creativecommons.org/ns#DerivativeWorks"/>
    <cc:legalcode rdf:resource="http://gnu.org/licenses/gpl-2.0.html"/>
    <dcq:title>GNU General Public License</dcq:title>
    <dcq:creator rdf:resource="http://fsf.org/"/>
  </cc:License>
</rdf:RDF>
```

Please note that the structure above is an RDF document, serialized as RDF/XML, self-published by the GNU Foundation, and using two vocabularies: DublinCore and the Creative Commons REL language. Similar machine-readable licences are available by other publishers such as the UK Open Government licence or the Creative Commons foundation. However, not every licence publisher gives the machine-readable version. Or maybe you want to define yours! -- The examples below will show you how.

3. How to use an existing machine-readable licence in HTML
Quite simple if the licence already exists. See the example below in HTML

```
    <p>My Dictionary is published under a  <a rel="license"
href="https://www.gnu.org/licenses/gpl-2.0">GPL 2.0</a> license. </p>
Nothing that any HTML developer wouldn't do. Because the RDF version is embedded
as RDFa in the GPL URI (and sometimes content negotiation is in place) you do not
even need to write the ".rdf"
```

4. How to use an existing machine-readable licence in RDF
Only one RDF triple suffices:

```
<http://mycompany/resource/001> <http://purl.org/dc/terms/license>
<https://www.gnu.org/licenses/gpl-2.0> .
```

5. How to use and discover machine-readable licences in RDF
As a result of the Prêt-à-LLOD project, a number of existing licences have been translated into ODRL 2.2 so that you can directly use them.

6. How to create a ODRL policy
In most cases, existing licences or policies will suffice. However, you may want to declare your own policy or licence with your own terms. In this case, you have to declare your own ODRL policy.

There is a full introductory document online with ODRL best practices. A sample ODRL policy for language resources can be as simple as follows, simply stating that a certain language resource 001 can be reproduced. The example uses the Turtle serialization of RDF for clarity. In the example below, the resource 001 is granted a permission to be reproduced.

```
@prefix odrl: <http://www.w3.org/ns/odrl/2/> .
<http://yourcompany.com/policies/001>
    a odrl:Policy;
    odrl:permission [
        a odrl:Permission ;
        odrl:target <http://yourcompany.com/resources/001> ;
        odrl:action odrl:reproduce
    ] .
```

6. How to create a ODRL-LR policy

The vocabulary provided by ODRL is quite generic, and you may want to have elements specific to the language domain -in such a case you need an ODRL-LR policy. An ODRL-LR policy is an ODRL policy using the vocabulary provided in this spec.

A sample ODRL-LR policy for language resources can be as simple as this.

```
@prefix odrl: <http://www.w3.org/ns/odrl/2/> .
@prefix odrl-lr: <http://purl.org/odrl-lr#> .
<http://yourcompany.com/policies/001>
    a odrl:Policy;
    odrl:permission [
        a odrl:Permission ;
        odrl:target <http://yourcompany.com/resources/001> ;
        odrl:action odrl-lr:localise .
    ] .
```

5.4. Formalization

The RDF version of a policy is per se a formalization, either visible as a collection of axioms in a Description Logic, or mappable into a First Order Logic (FOL) system. Still, a reduced formalization can be of help for the implementation.

> **Definition (policy, request, context)**. An enforceable *policy* P is a tuple $\langle u,g,a,C, \pm \rangle$ where the sign $\pm$ denotes permission or prohibition, $u \in U$ a user within a set of users, g a Language Data resource, $a \in A$ an action among the possible A actions, and C a set of boolean conditions $\{c0, c1, ...\}, ci \in \{F,T\}$, whose value depends on some variables $x_i(t) \in X$ which may change with the time —the set of external variables X(t) being called *context*. A *request* q is a tuple $\langle u,g,a \rangle$, issued at a certain time.

Please note that both policies and requests are considered `odrl:Policy` (the first one being of type `odrl:Set`, the second one being of type `odrl:Request`). Alternatively, policies can be

matched to `odrl:Rule` –please note that permissions can be prohibitions, obligations or permissions.

> **Definition (policy match)**. A request q = $\langle u_q, g_q, a_q \rangle$ matches an enforceable policy $P = \langle u_p, g_p, a_p, C, \pm \rangle$ iff $((u_p = u_q \vee u_p = \emptyset) \wedge (a_p = a_q \vee a_p = \emptyset) \wedge (\forall c_i \in C, c_i = T)$.

This definition does not consider actions subsuming other actions, but declares that the absence of users or actions in the policy assumes generality.

> **Definition (authorization decision)**. Given a request Q, a context $X(t)$, and a set of enforceable policies P, the authorization decision is the calculus that determines the subset of G that is permitted and the subset of G that is prohibited, from the requested g q and determined by the graphs g p of the matching policies for a given context.

Let us illustrate the definitions above with one example. Let p be a policy saying «Academic individuals can play my dictionary». $p = \langle u \,/\, \text{academic}(u), \text{my\_dictionary}, \text{play} \rangle$. In a First Order Logic language we may define these language elements:

my_dictionary	≡ individual which represents a certain resource
play	≡ individual wich represents using a certain action type
academic(u)	≡ u is academic
can(a)	≡ the action a is permitted
actionType(a,at)	≡ the action a is of type at
exectedBy(a,u)	≡ the action a is executed by user u
exectedOver(a,g)	≡ the action a is executed over resource g
Action(play)	means that play is an invididual of the Action class
User(bob)	means that bob is an invididual of the User class
ActionType(at)	means that the action a is of type play

We have defind exactly five monadic predicates, three dyadic predicates, and three individuals
The policy is represented in FOL as logic

$$p_1 \equiv \forall u(\text{academic}(u) \rightarrow \text{can}(\text{executedBy}(a,u) \wedge \text{executedOver}(a,\text{my\_dictionary}) \wedge \text{actionType}(a,\text{play}))$$

The definitions above are ilustrated by the Figure 3:

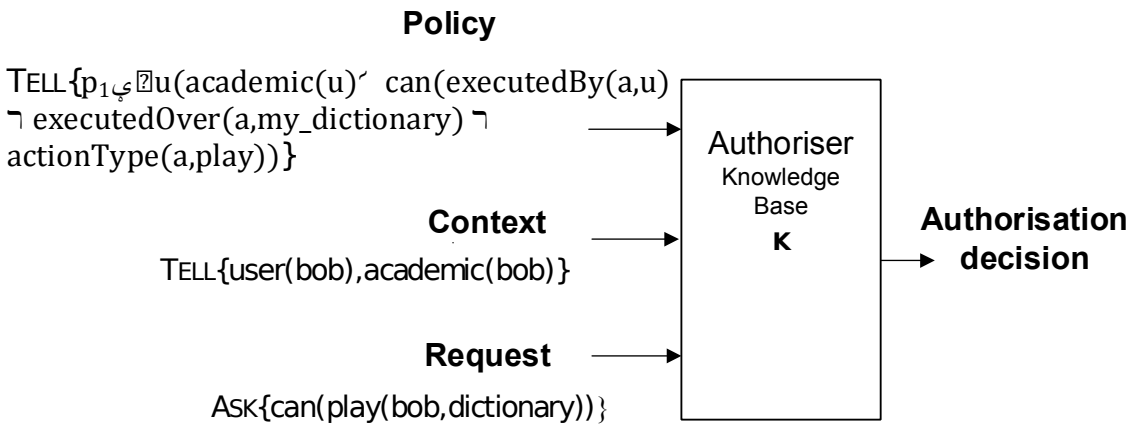

Figure 3. Policy, Context, request and authorisation decision

The authorisation and compatibility decision can be done in two steps: a first one determining which policies match, and a second one (in Figure 3) calculating the permitted and prohibited resources. This particular approach is shown in the Figure 4: a Policy Matcher decides which policies match a request, and the Graph Matcher decides which parts of the graph G are accessible for a request or not.

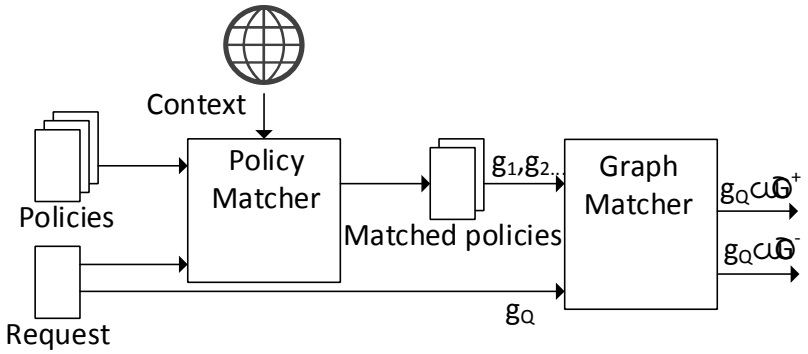

Figure 4. Role of the policy matcher and the graph matcher

Policy matcher

The policy matcher receives the set of all declared policies P_i and a request q and it returns the set of graphs g i present in the matching policies as defined before. The different procedures to decide whether a policy and request match (as described above) are various: (1) executing rules in a rule engine for Horn rules like such as SWRL etc. (2) making an inference with a DL reasoner (subsumption etc.) (3) declaring the rule in a declarative language and using logic programming (PROLOG etc.) (4) executing an algorithm with an imperative language (Java, etc.).

The Algorithm 1 determines the bag of accesible and non-accesible graphs by simply matching users, actions and contexts in the policies. In order to model the commercial strategy of selling a per-triple language resource access (such as a dictionary entry), special semantics are needed, as one policy for each triple would be inefficient. In this case, the policy can specify that the price in the duty applies for each accessed triple within the bigger subset (such as `gr:UnitOfMeasurement`, in the namespace of the GoodRelations vocabulary).

Algorithm 1 Calculates the bags of permitted and prohibited graphs

```
1: Input: a request q < u,g,a >, a set of policies P < u,g,a,C,± >, a context X
2: Output: a bag of graphs g + and a bag of graphs g −
3: procedure PolicyMatcher
4:     g+,g−,← []
5:     for all P_i do
6:         for all c_j inP_i do
7:         if Evaluate(c_j) == false then
8:             continue
9:         if (P_i .u = q.u&&P_i .a = q.a&&p.s = +) then
10:            add(g+,P i .g)
11:        if (P i .u = q.u&&P i .a = q.a&&p.s = −) then
12:            add(g−,P i .g)
13:        output ← g+, g−
```

Graph matcher

The graph matcher receives a set of language resources (or graphs patterns) g_p present in zero or more policies and the graph pattern in a request g_q , and it returns the permitted and prohibited graphs.

Given a request q and a set of policies P, the accesible graph and the non-accesible graph are defined as: $G^+ := (\cup g_i^+ - \cup g_i^-) \cap g_q$ and $G^+ := (\cup g_i^- - \cup g_i^+) \cap g_q$ respectively, where each gi is the graph in a policy $P < u_p,g_p,a_p,C,\pm >$ matching the request Q. The *conflictive graph* $G^\pm = (\cup g^+_i \cap \cup g^-_i) \cap g_q$ covers the language resources (RDF triples) for which the action a_q is both expressly permitted and prohibitted, and the undefined graph spans the triples not affected by any matching policy, $G^0 = G - G^+ - G^-$. The Venn diagram is represented in the figure below The default behaviour of the Graph Matcher in case of conflict may decide to assign $G^\pm$ to G^+ or to G^- , the default behaviour in case of indefinition may assign the undefined graph to the accessible or to the non-accesible graphs —ultimately the Graph Matcher decides whether a resource (RDF triple) is authorised or not. Because language resources are generally protected by copyright, the default policy is to use the negative assumption.

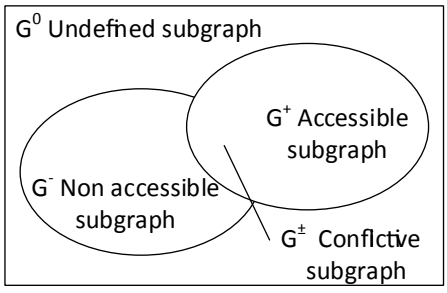

Figure 5. Policy conflicts: language resources with explicit permissions and prohibitions

Each of the graphs g p can be given as a simple IRI referring to an existing named graph, or they can be defined by means of a sparql construct query (graphs g p in the policies). In the first case, the query Q can be made over the allowed named graphs to obtain G + , being G − obtained as the set difference between g q and G + . In the second case, the request

query might be made on the unions over the graph created by the sparql construct (also called *views*).
However, nested queries are not supported by SPARQL and this would imply making two consecutive queries. This suggests two strategies: (i) an static approach, where for each newly created policy, a new named graph is created. Upon request, only authorized graphs are queried (ii) a dynamic approach, where no precomputed information whatsoever exists: queries in the request are rewritten so that beffore attacking the actual PDDM server, restricted triples are filtered.

Practical aspects

Considering the ODRL-LR data model elements and the authorisation framework, the following mapping principles are suggested:

- An enforceable policy is represented by a `odrl:Rule`. ODRL policies possibly comprise several rules, hence a single ODRL policy (instance of `odrl:Set`, subclass of `odrl:Policy`) can be decomposed in several enforceable policies as defined before.
- The kind of rule (±) is determined by the predicate introducing the rule (`odrl:Permission`, `odrl:Duty` or `odrl:Prohibition`).
- The user u is the IRI given with the predicate `odrl:assignee`, of the class `odrl:Party` The action a is the IRI given with the predicate `odrl:actions`: those in ODRL-LR.
- The graph g is defined by a sparql construct query, given as a text string predicated by `odrl:target`
- The default behaviour for the conflictive graph can be defined in ODRL with the term `odrl:conflict` , the default behaviour for the undefined resources with the ODRL term odrl:undefined — leading to four different scenarios. By default, language resources are protected by copyright and we shall be conservative.

6. Dataset

In order to initialize the server of the service with real data, a set of 183 licences have been acquired, transformed into ODRL v2.2 and published into the Fuseki server. The collection of licences is publicly available in a folder in the GitHub project PDDM in which each licence is stored in a different Turtle RDF file. One example of license follows.

```
@prefix odrl:  <http://www.w3.org/ns/odrl/2/> .
@prefix odrl-lr:   <http://purl.org/odrl-lr#> .
@prefix cc:    <http://creativecommons.org/ns#> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .
@prefix skos:  <http://www.w3.org/2004/02/skos/core#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dct:   <http://purl.org/dc/terms/> .
@prefix prov: <http://purl.org/net/provenance/ns#> .

:APACHE1.0  a odrl:Policy ;
  rdfs:label       "Apache License" ;
  prov:wasDerivedFrom <http://www.apache.org/licenses/LICENSE-1.0.txt> ;
```

```
  rdfs:seeAlso <https://tldrlegal.com/api/license/559ec47447cdaa075e00023f> ;
  cc:legalcode     "Copyright (c) 1995-1999 The Apache Group.  All rights
reserved. Redistribution [... deleted text...]"@en ;
  dct:hasVersion   "1.0" ;
  dct:language     <http://www.lexvo.org/page/iso639-3/eng> ;
  dct:publisher    "The Apache Group" ;
  dct:source       <http://www.apache.org/licenses/LICENSE-1.0> ;
  owl:sameAs       <https://licensedb.org/id/Apache-1> ;
  odrl:prohibition [
    a odrl:Prohibition ;
    odrl:action odrl-lr:holdLiable, odrl-lr:useTrademark
    ] ;
  odrl:permission  [
    a odrl:Permission ;
    odrl:action  cc:Reproduction , cc:DerivativeWorks , cc:Distribution;
    odrl:duty [
      a odrl:Duty ;
      odrl:action  cc:ShareAlike , cc:Attribution
    ]
] .
```

The community can use these licenses or propose new ones in a collaborative way. Moreover, the Fuseki server has been initialized with the 183 licences. The list of the 183 licences are listed in the Table below.

https://github.com/Pret-a-LLOD/pddm/tree/develop/data/licenses

Table 2. List of transformed licences

Licenses			
afl3.0	cc-by-nd3.0fr	cc-by2.5dk	gpl1.0
againstdrm2.0	cc-by-nd3.0gr	cc-by2.5hu	gpl2.0
agpl3.0	cc-by-nd3.0ie	cc-by2.5il	gpl3.0
allrightsreserved	cc-by-nd3.0it	cc-by2.5in	IBM1.0
APACHE1.0	cc-by-nd3.0nl	cc-by2.5mx	iodl1.0
APACHE2.0	cc-by-nd3.0pt	cc-by2.5pe	lgpl2.0
ARTISTIC2.0	cc-by-nd3.0ro	cc-by2.5pt	lgpl2.1
BOOST1.0	cc-by-nd3.0ve	cc-by2.5scotland	lgpl3.0
BSD2.0	cc-by-nd4.0	cc-by2.5se	MICROSOFT1.0
BSD3.0	cc-by-sa2.0	cc-by3.0	MIT1.0
BSD4.0	cc-by-sa3.0	cc-by3.0at	MOZILLA2.0
cc-by-nc-nd2.0	cc-by-sa3.0at	cc-by3.0au	ms-c-nored-ff
cc-by-nc-nd3.0	cc-by-sa3.0au	cc-by3.0br	ms-c-nored
cc-by-nc-nd3.0at	cc-by-sa3.0br	cc-by3.0ch	ms-commons-byncnd
cc-by-nc-nd3.0au	cc-by-sa3.0ch	cc-by3.0cl	NDL1.0
cc-by-nc-nd3.0br	cc-by-sa3.0cl	cc-by3.0cn	odbc-by1.0
cc-by-nc-nd3.0ch	cc-by-sa3.0cn	cc-by3.0de	odbc-pddl1.0
cc-by-nc-nd3.0cl	cc-by-sa3.0de	cc-by3.0ec	odbl1.0
cc-by-nc-nd3.0cn	cc-by-sa3.0ec	cc-by3.0eg	OGCDocument1.0
cc-by-nc-nd3.0de	cc-by-sa3.0es	cc-by3.0es	OGCSoftware1.0
cc-by-nc-nd3.0ec	cc-by-sa3.0fr	cc-by3.0fr	ogl-nc1.0
cc-by-nc-nd3.0es	cc-by-sa3.0gr	cc-by3.0gr	OGL1.0
cc-by-nc-nd3.0fr	cc-by-sa3.0ie	cc-by3.0ie	OL1.0
cc-by-nc-nd3.0gr	cc-by-sa3.0it	cc-by3.0it	ORACLE1.0
cc-by-nc-nd3.0ie	cc-by-sa3.0nl	cc-by3.0lu	OS3.0
cc-by-nc-nd3.0it	cc-by-sa3.0pt	cc-by3.0nl	PDM1.0
cc-by-nc-nd3.0nl	cc-by-sa3.0ro	cc-by3.0nz	publicdomain

cc-by-nc-nd3.0pt	cc-by-sa3.0ve	cc-by3.0pl	simple2.0
cc-by-nc-nd3.0ro	cc-by-sa4.0	cc-by3.0pt	ukogl-nc2.0
cc-by-nc-nd3.0ve	cc-by1.0	cc-by3.0ro	ukogl1.0
cc-by-nc-nd4.0	cc-by2.0	cc-by3.0th	ukogl2.0
cc-by-nc-sa2.0	cc-by2.0at	cc-by3.0us	ukogl3.0
cc-by-nc-sa3.0	cc-by2.0au	cc-by3.0ve	W3C1.0
cc-by-nc-sa4.0	cc-by2.0be	cc-by4.0	
cc-by-nc2.0	cc-by2.0br	cc-zero1.0	
cc-by-nc3.0	cc-by2.0ca	CDDL1.0	
cc-by-nc4.0	cc-by2.0cl	clarin_aca_by	
cc-by-nd2.0	cc-by2.0de	COLORIURIS1.0	
cc-by-nd2.0be	cc-by2.0es	COMMON1.0	
cc-by-nd2.5dk	cc-by2.0fr	CRYPTIX1.0	
cc-by-nd3.0	cc-by2.0it	ECLIPSE1.0	
cc-by-nd3.0at	cc-by2.0jp	elra-end-user	
cc-by-nd3.0au	cc-by2.0kr	elra-var	
cc-by-nd3.0br	cc-by2.0nl	EUC1.0	
cc-by-nd3.0ch	cc-by2.0uk	EUPL1.1	
cc-by-nd3.0cl	cc-by2.0za	fal1.3	
cc-by-nd3.0cn	cc-by2.5ar	FREEBSD1.0	
cc-by-nd3.0de	cc-by2.5bg	gfdl1.1	
cc-by-nd3.0ec	cc-by2.5ch	gfdl1.3	
cc-by-nd3.0es	cc-by2.5cz	GOVTRACK1.0	

7. Services

This section describes the Policy-Driven Data Manager (PDDM), a platform providing data. This section comprises the design of the PDDM architecture, the main components, and the most relevant REST API methods.

7.1. Design and Architecture

This section describes the technical aspects of the implemented service. The project is named **pddm** and is allocated in GitHub with public access and open licence. The overall architecture is presented in the Figure below. The project comprises 4 main modules: the odrl-api java library, the REST-API module, the project **pddm** and the Fuseki server.

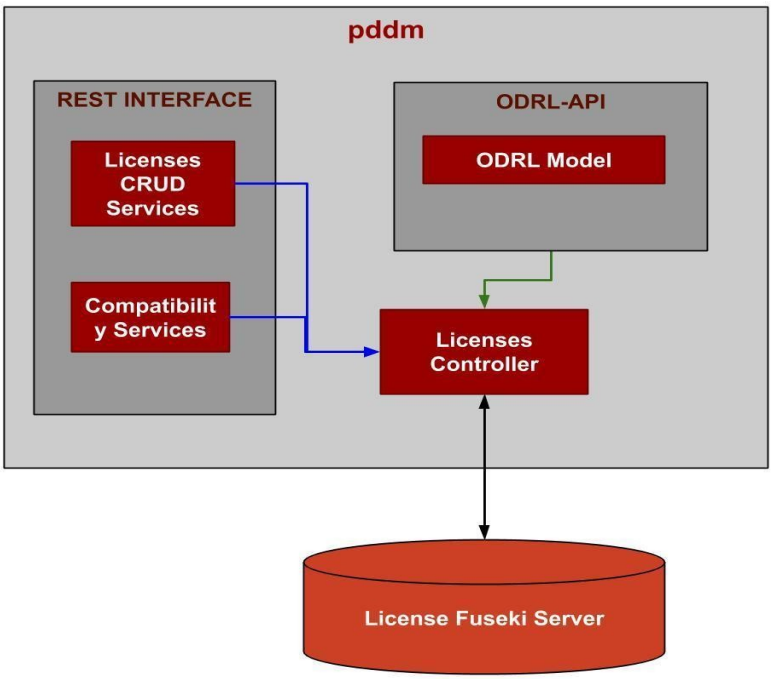

Figure 6. General architecture of the PDDM project

Implementation details

The projects are implemented in Java language, using maven and living in an open repo online. Deployment is automated through a proper githook: changes in the master branch trigger git pull, compilation and deployment. The HTTP REST service runs as a Linux service with a dedicated daemon in a Virtual Machine running over a dedicated server with backup and professional maintenance.

Servers are hosted in Madrid (at Universidad Politécnica de Madrid) and no data is leaked out of the European Union –although no personal data is for the moment stored.

7.2 Architecture components

ODRL-API Library

The odrl java api is the core library in which the model of ODRL is defined as a Java POJO class with all the possible attributes defined in the Data Model Section. Moreover, some functionalities have been included in the library for ODRL validation and conversion. This library was created before Prêt-à-LLOD but this project has updated its features to the latest ODRL version, with important changes.

The library is allocated in GitHub: https://github.com/oeg-upm/licensius/tree/master/jodrlapi

PDDM project

The Java project has been implemented with the framework Spring as a Spring Boot Application. The project is in charge of the deployment of the REST services, uses the ODRL-API library as the model of the ODRL objects and controls them to provide the required functionalities, as the classical Model-View-Controller pattern. Moreover, the project presents a presentation web page in which information regarding this WP is presented as well as the important links to the code and services.

The library is allocated in GitHub: https://github.com/Pret-a-LLOD/pddm

REST-API module

The REST-API module has been developed following the specifications of Spring Framework and enriched with the Swagger library, which is in charge of providing a user-friendly web page to show all the provided REST functions of the project. The REST-API is the interface to the logic of the pddm project and to the RDF server. The functions declared in REST-API are CRUD services for licences (Create, Retrieve, Update and Delete) and functions to get the compatibility between two licences.

Fuseki Server

For the developed project, an Apache Jena Fuseki server has been used and deployed in an UPM's server to store and retrieve the licences that are used by the project and its services.

7.3. Service description of the REST API module

This section describes a collection of HTTP REST API services that facilitates licence definition, compliance checking and licence compatibility. The services of the Policy Driven Data Manager (PDDM) are divided into two main groups: services for licences and services for compatibilities.

1 The PDDM as a register of licences
CRUD operations on licenses-templates. Users can create a licence, upload, read or delete it. Also, a method to retrieve all of them is provided. Licences are identified (id) by their name.

```
GET /license/   → returns complete list of ids (URLs)
GET /license/{licensid}
POST /license/{licensid}
DELTE /license/{licensid}
```

2 The PDDM providing smart functionality around licences
Simple licence composition method.

```
GET /isCompatible(lic url1, lic url2)
GET compatibility/{licenseid1}/{lincenseid2}
GET compatibility?license1={lincenseid1}&license2={lincenseid2}
```

Advanced licence composition method. For example, if licence 1 is CC-BY-ND and licence 2 is CC-BY-NC, the results should have AT LEAST these two constraints (e.g. CC-BY-ND-NC

is possible, but additional constrains are possible as well). The result is a licence with the minimal restrictions that must exist.

```
POST /calculateMinimumLicense(lic url1, lic url2)
GET minimumCompatibilityLicense?license1={lincenseid1}&license2={lincenseid2}
```

The API-REST has been published with Swagger[6] as the Figure below shows.

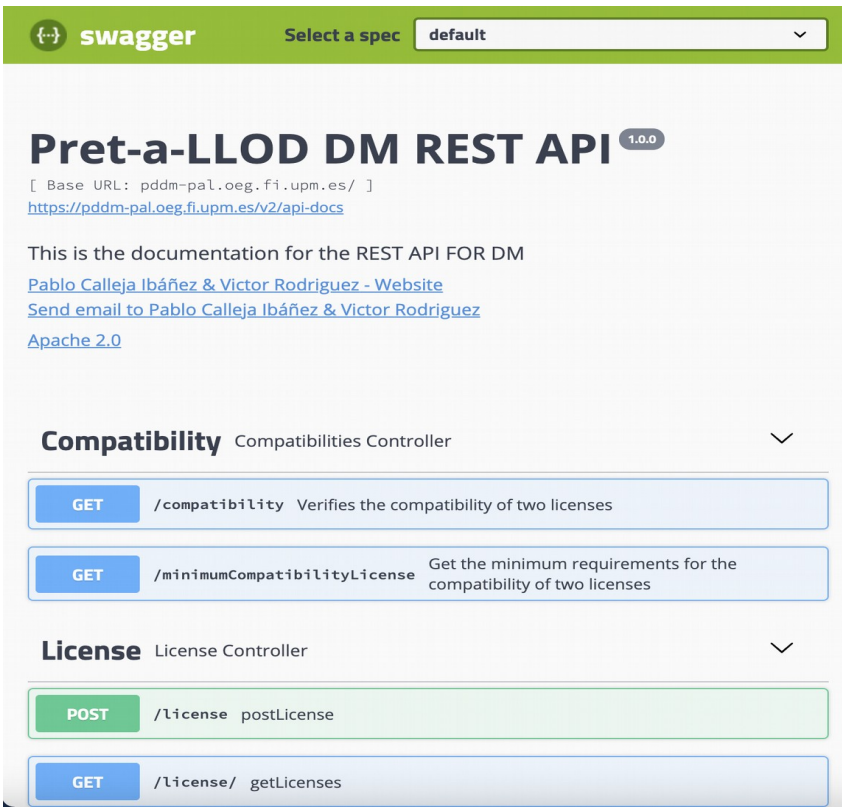

Figure 7. OpenAPI description of the PDDM REST API

Moreover, a web portal[7] has been developed in order to present the contributions of the WP with the information of this document and links to the project in GitHub and to the Swagger API (OpenAPI). Figure below is a screenshot of the main page.

[6] https://pddm-pal.oeg.fi.upm.es/swagger-ui.html#
[7] https://pddm-pal.oeg.fi.upm.es/

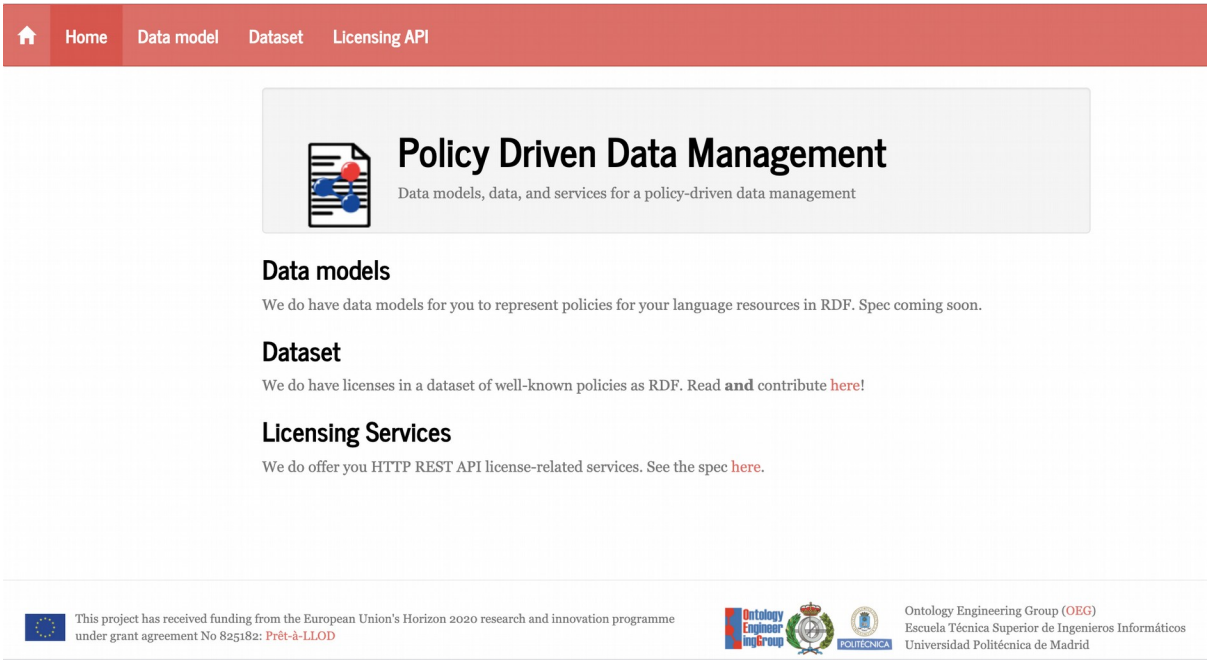

Figure 8. Screenshot of the PDDM web portal as of November 2020

8. References

[1] H2020 Programme Guidelines on FAIR Data Management in Horizon 2020 (2016), Version 3, Directorate-General for Research & Innovation, European Commission.

[2] Noussias, A. (2019) D7.1 Ethics Requirements I. Prêt-à-LLOD deliverable.

[3] McCrae, J. P., & Cimiano, P. (2015). Linghub: a Linked Data based portal supporting the discovery of language resources. SEMANTiCS (Posters & Demos), 1481, 88-91.

[4] Rodríguez Doncel, V., Gómez-Pérez, A., & Villata, S. (2014). A dataset of RDF licenses. in Proc. of the 27th Int. Conf. on Legal Knowledge and Information System (JURIX), R. Hoekstra (Ed.), ISBN 978-1-61499-467-1, pp. 187-189, IOS Press

[5] Radulovic, F., Poveda-Villalón, M., Vila-Suero, D., Rodríguez-Doncel, V., García-Castro, R., & Gómez-Pérez, A. (2015). Guidelines for Linked Data generation and publication: An example in building energy consumption. Automation in Construction, 57, 178-187.

[6] McCrae, J. P., Labropoulou, P., Gracia, J., Villegas, M., Rodríguez-Doncel, V., & Cimiano, P. (2015, May). One ontology to bind them all: The META-SHARE OWL ontology for the interoperability of linguistic datasets on the Web. In European Semantic Web Conference (pp. 271-282). Springer, Cham.

[7] McCrae, J. P. (2019) D7.4 Ethics Requirements 4. Prêt-à-LLOD deliverable

[8] McCrae, J. P., Chiarcos, C., Bond, F., Cimiano, P., Declerck, T., de Melo, G. & Osenova, P. (2016). The Open Linguistics Working Group: Developing the Linguistic Linked Open Data Cloud. In Proc. of the Tenth Int. Conf. on Language Resources and Evaluation, 2435-2441.

[9] Lebo, Timothy, et al. (2013) "Prov-o: The prov ontology." W3C recommendation 30 (2013).

[10] Iannella, R., & Villata, S. (2018). Odrl information model 2.2. W3C Recommendation.
[11] Iannella, R., Steidl, M., Myles, S., & Rodriguez-Doncel, V. (2018). ODRL Vocabulary & Expression 2.2: W3C Recommendation, 15 February 2018.
[12] Rodriguez-Doncel, V., Labropoulou, P. (2015). Digital Representation of Rights for Language Resources, in Proc. of the 4th W. on Linked Data in Linguistics: Resources and Applications, pp. 49-58, ISBN 9781941643570, ACL-IJCNPL (Eds.)
[13] Steyskal, S., & Polleres, A. (2014). Defining expressive access policies for linked data using the ODRL ontology 2.0. In Proceedings of the 10th International Conference on Semantic Systems (pp. 20-23).
[14] V. Rodríguez-Doncel, M.C. Suárez Figueroa, A. Gómez-Pérez and M. Poveda Villalón (201) Linked Data Rights 2.0 - Extension of ODRL for Licensing Linked Data, in Vocarnival Workshop within 10th Int. Conf. on Semantic Systems (SEMANTiCS).
[15] Kagal, L., Finin, T., Joshi, A., 2003. A Policy Based Approach to Security for the Semantic Web, in: Fensel, D., Sycara, K., Mylopoulos, J. (Eds.), The Semantic Web - ISWC 2003, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 402–418.
[16] Tonti, G., Bradshaw, J., Jeffers, R., Montanari, R., Suri, N., Uszok, A., 2003. Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder, in: Fensel, D., Sycara, K., Mylopoulos, J. (Eds.), The Semantic Web- ISWC 2003, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 419–437.
[17] Abel, F., Coi, J., Henze, N., Koesling, A., Krause, D., Olmedilla, D., 2007. Enabling Advanced and Context-Dependent Access Control in RDF Stores, in: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudr´ e-Mauroux, P. (Eds.), The Semantic Web, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 1–14.
[18] Carroll, J.J., Bizer, C., Hayes, P., Stickler, P., 2005. Named Graphs, Provenance and Trust, in: Proceedings of the 14th International Conference on World Wide Web, WWW '05. ACM, New York, NY, USA, pp. 613–622. doi:10.1145/1060745.1060835
[19] J. Carroll and P. Stickler. TriX: RDF Triples in XML. In Proceedings of Extreme Markup Languages, 2004
[20] Pellegrini, T., Havur, G., Steyskal, S., Panasiuk, O., Fensel, A., Mireles, V., ... & Schönhofer, A. (2019). DALICC: A License Management Framework for Digital Assets. Proceedings of the Internationales Rechtsinformatik Symposion (IRIS), 10.
[21] Kirrane, S., Mileo, A., Polleres, A., & Decker, S. (2020). Query Based Access Control for Linked Data. arXiv preprint arXiv:2007.00461.
[22] Havur, G., Steyskal, S., Panasiuk, O., Fensel, A., Mireles, V., Pellegrini, T., ... & Kirrane, S. (2019). Automatic License Compatibility Checking. In SEMANTICS Posters&Demos.
[23] Governatori, G., Iannella, R., 2011. A modelling and reasoning framework for social networks policies. Enterprise IS 5, 145–167.
[24] Governatori, G., 2005. Representing Business Contracts in RuleML. Int. J. of Cooperative Information Systems, 14 (2-3), 181–216.

[25] Steyskal, S., & Polleres, A. (2014, September). Defining expressive access policies for linked data using the ODRL ontology 2.0. In Proceedings of the 10th International Conference on Semantic Systems (pp. 20-23).
[26] Rodriguez-Doncel, V., Villata, S., & Gómez-Pérez, A. (2014, December). A dataset of RDF licenses. In JURIX (pp. 187-188).
[27] Governatori, G., Rotolo, A., Villata, S., & Gandon, F. (2013, October). One license to compose them all. In International semantic web conference (pp. 151-166). Springer, Berlin, Heidelberg.
[28] Suárez-Figueroa, M. C. D5. 4.1. NeOn Methodology for Building Contextualized Ontology Networks.
[29] Peroni, S., Shotton, D., & Vitali, F. (2012, October). The live OWL documentation environment: a tool for the automatic generation of ontology documentation. In International Conference on Knowledge Engineering and Knowledge Management (pp. 398-412). Springer, Berlin, Heidelberg.
[30] Poveda-Villalón, M., Suárez-Figueroa, M. C., & Gómez-Pérez, A. (2012, October). Validating ontologies with oops!. In International conference on knowledge engineering and knowledge management (pp. 267-281). Springer, Berlin, Heidelberg.
[31] McCrae, J. P. and Cimiano, P. (2015). Linghub: a Linked Data based portal supporting the discovery of language resources. In Proceedings of the 11th International Conference on Semantic Systems, pp. 88–91.