An Ontological Approach for the Management of Rights Data Dictionaries

Roberto GARCÍA, Jaime DELGADO
Universitat Pompeu Fabra (UPF), Departament de Tecnologia,
Pg. Circumval·lació 8, E-08003 Barcelona, Spain
{roberto.garcia, jaime.delgado}@upf.edu

Abstract. We started some time ago the development of RDDOnto, which provides an ontological approach to the Rights Data Dictionary (RDD) part of MPEG-21, one of the main Intellectual Property Rights (IPR) Management standardisation efforts. In order to build the ontology, the terms defined in the RDD specification have been modelled using OWL, trying to capture the greatest part of its semantics. The ontology allows formalising a great part of the standard and simplifying its verification, consistency checking and implementation. During the RDDOnto construction, some integrity problems were detected, which even led to a standard corrigendum. Additional checks were possible using Description Logic reasoning in order to test the standard consistency. Moreover, RDDOnto is now helping on how new terms can be added to the RDD and to integrate the RDD with other parts of MPEG-21 also mapped to OWL. Finally, there are the implementation facilities provided by the ontology. They have been used to develop MPEG-21 licenses searching, validation and checking. Existing ontology-enabled tools as semantic query engines or logic reasoners facilitate this.

Keywords. Digital Rights Management, Intellectual Property, Ontology, Semantic Web

Introduction

The number of online marketplaces has grown in recent years and will continue to expand in the future. Content companies consider unauthorized use or reproduction of digital content a serious problem. The goal of a Digital Rights Management (DRM) system is to enforce licenses between a content provider and a consumer that define rules about authorized use of managed content.

The Moving Pictures Expert Group (MPEG) [1] is the ISO/IEC working group in charge of developing standards for the coded representation of digital audio and video. Among other standards, the group is working on the MPEG-21 standard [2] with the objective of developing a standardized multimedia framework. The fifth part of MPEG-21 specifies a Rights Expression Language (REL) [3] and the sixth one an associated Rights Data Dictionary (RDD) [4].

The rights language introduces ways to assign rights expression to digital goods or services and to control usage and access. The two constitutive factors in a rights language are its syntax and semantics. The term syntax refers to the grammar rules, which apply to the language's vocabulary, whereas the term semantics refers to the meaning of valid sentences in the language. Each rights expression language includes a rights vocabulary or dictionary, which defines the permitted words and their semantics.

There are other initiatives for rights expression languages, but there is just one that defines also a rights data dictionary. It is the ODRL initiative [5]. However, the ODRL Data Dictionary (DD), on the contrary to MPEG-21 RDD, was not developed as an ontology that provides the semantics of the REL terms. ODRL DD is a XML Schema, like MPEG-21 REL and ODRL REL, and it extends the generic elements defined in the ODRL REL but from a purely syntactic point of view.

Therefore, we have centred our research on applying an ontological approach to rights data dictionaries on MPEG-21 RDD. It is generic so it can be applied to other dictionaries similar to the MPEG-21 one. For dictionaries similar to ODRL we have explored other methodologies that also try to move them to an ontology space. The common part is that the final objective in all cases is to use web ontologies as an integration framework where all these initiatives and approaches can be connected, made interoperable and enriched from the new facilities provided by ontologies.

1. The Rights Data Dictionary Ontology

The objective of RDDOnto is to translate the RDD terms descriptions from its current textual representation in the dictionary to a machine processable representation using the semantic web paradigm.

The set of all predefined classes and properties are the building blocks provided by the OWL [6] and RDF/S (RDF plus RDFSchema) [7,8] frameworks. These building blocks are used to construct Semantic Web ontologies, i.e. sets of restrictions to the basic RDF elements. These restrictions can be automatically validated in order to test that a particular RDF description conforms to the semantics of the particular domain captured by the ontology.

In the next subsections, we will analyse RDD and then detail how first RDF/S and afterwards OWL frameworks can be used to capture RDD terms definitions and a great part of their semantics. RDF/S is capable of modelling only a fraction of the RDD semantics. This fraction is augmented when the constructs introduced by OWL are also used. Therefore, two versions of the ontology can be produced. The simpler one uses RDF/S and the more complex one uses OWL.

1.1. RDD Specification analysis

The RDD Specification [4] defines a set of terms, the "words" in the vocabulary. The RDD Specification is self contained so all the terms that it uses, even the relating terms, are defined in it. For each term, its description is composed by a set of attributes:

- **Headword**: the term name. It must appear in the term description.
- **Synonym**: some alternative names. It is not mandatory.
- **Definition**: a short text that defines the term.
- **MeaningType**: allowed values are: Original, PartlyDerived and Derived.
- **Comments**: extended textual information about the term. It is not mandatory.
- **Relationships**: this attribute lists the relationships, from a set of predefined ones, among this term an other terms. They are used to specify the term semantics from different points of views. The relations are classified in the following categories:
  - **Genealogy**: these relations give a semantic point of view similar to that from Semantic Networks [9], i.e. inheritance, relations domain and range, etc. Therefore, they are quite similar to RDF/S and OWL semantics. The relations are: *IsTypeOf*, *IsA*, *Is*, *IsEquivalentTo*, *IsOpposedTo*, *IsPartOf*, *IsAllowedValueOf*, *HasDomain*, *HasRange* and *IsReciprocalOf*.
  - **Types**: the types are enumerated using the *HasType* relationship. It is the reciprocal of *IsTypeOf*.
  - **Membership of Sets**: the relating term from members to sets, *IsMemberOf*. RDD sets are considered equivalent to RDF Containers and thus it can be related to RDF membership relations.
  - **Family**: these relationships connect an *ActType* and the terms that it begets through the application of the Context Model semantics. E.g. *BegetsAgentType* and more details in Fig. 1. Moreover, there are the interrelations inside the concrete act type context model among the begotten terms, the *ActionFamily* relational view. E.g. between *Place* and *Agent* for *Act ActType* there is *IsPlaceOfActingBy* relationship (see Fig. 2).

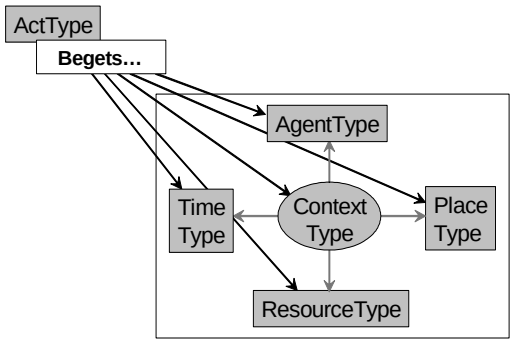

Fig. 1. The Begets relationships in *ActionFamily*, from RDD standard [17], section A.10.6.1

  - **ContextView**: the group of relationships describing the attributes of a specific *ContextType* using the Context Model semantics. They are: *isContextOfAgent*, *isContextOfResource*, *isContextOfTime*, *isContextOfPlace*, *HasValue* and their reciprocals *IsAgentInContext*, etc. More details in Fig. 3.

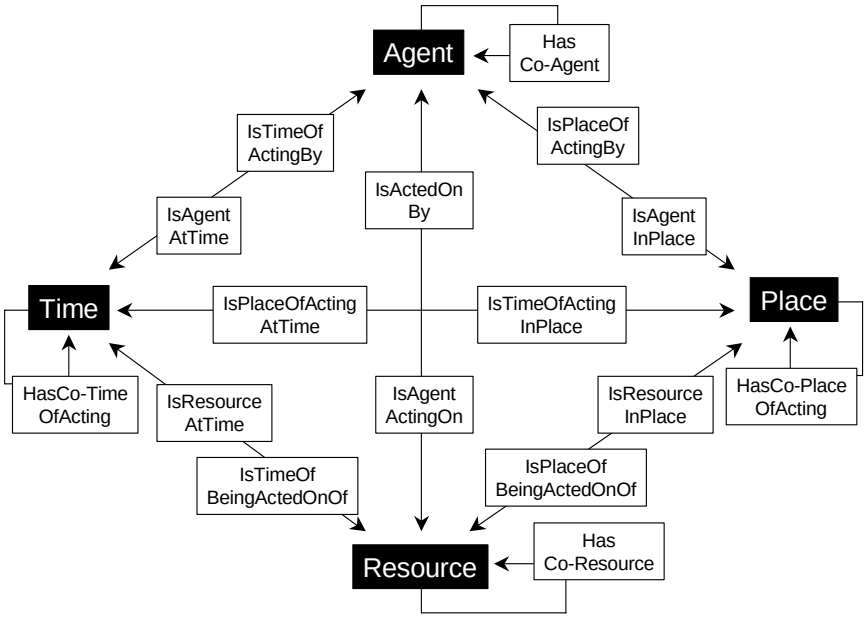

Fig. 2. The *Action**Family*** relational view for *Act*, from RDD standard [17], section A.10.6.2

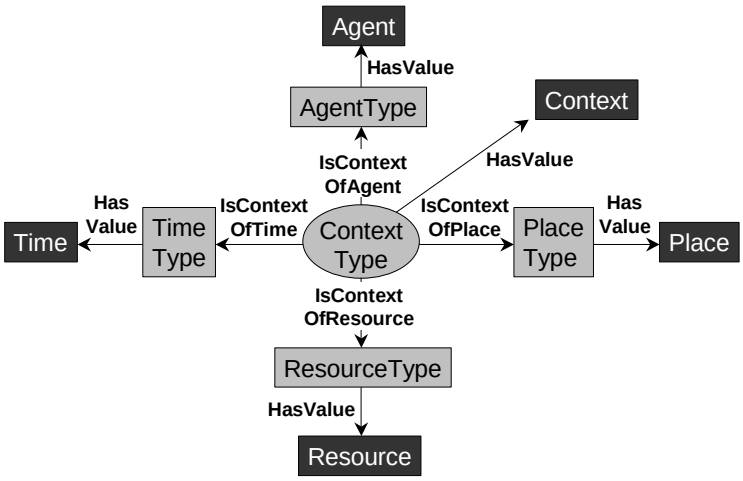

Fig. 3. RDD Context Model relationships, from RDD standard [17], section A.10.1

1.2. RDD to web ontology mappings

From the RDD Specification analysis two kinds of attributes can be detected. The first group is composed by those attributes with unstructured values, i.e. textual values. They can be easily mapped to predefined or new RDF properties with textual (literal) values.

The first option is to try to find predefined RDF properties that have the same meaning that the RDD term attributes that are being mapped. When this is not possible, the RDFS constructs will be used to define new RDF properties to which the corresponding attributes will be mapped. These properties are defined in the RDDOnto namespace, "rddo".

The mappings of this kind are shown in Table 1. Note that the Dublin Core [10] RDF Schema is also reused in RDDOnto. The Dublin Core (DC) metadata element set is a standard for cross-domain information resource description. The DC RDF Schema implements the Dublin Core standard.

Table 1. Mappings for the RDD attributes with text value

RDD Attribute	RDF Property	Kind of RDF property
Headword	rdf:ID	Predefined in RDF
Synonym	rddo:synonym	New property defined in RDDOnto
Definition	dc:description	Predefined in Dublin Core RDFS
MeaningType	rddo:meaningType	New property defined in RDDOnto
Comments	rdfs:comment	Predefined in RDFS Schema

The other kind of attribute is the *Relationships* one. Its value is not textual. Firstly, it is categorised into five groups: *Genealogy*, *Family*, *ContextView*, *Types* and *Membership of Sets*. Each of these groups is composed by a set of relations that can be used to describe a term related to other terms in the RDD specification.

As it has been shown in the previous section, these groups of relationships take different semantic points of view. The *Genealogy*, *Types* and *Membership of Sets* groups comprise relationships with semantics almost equivalent to RDF/S and OWL ones. The semantic equivalences have been deduced from RDD, RDF/S and OWL specifications.

The relations in this groups that can be mapped to RDF/S are presented in the upper part of Table 2. There is also a short description and the equivalent RDF property used to map them in RDDOnto. Only the RDD relations with an equivalent property in RDF/S are mapped at this level, i.e. *IsTypeOf*, *IsA*, *HasDomain* and *HasRange*. The other relations have associated semantics that do not have equivalence in RDF/S.

Therefore, if the mapping is restricted to the possibilities provided by RDF/S, then we get an incomplete ontology, i.e. it does not capture all the available semantics of RDD. However, on top of RDF/S, more advanced restriction building tools, like OWL, have been developed.

Using OWL ontology building blocks, some of the previously unmapped RDD relations can be mapped to the RDD ontology. In bottom part of Table 2 they are presented together with a short description and the equivalent OWL property used to map them in RDDOnto. With OWL almost all relationships can be mapped.

Only *Is* and *IsPartOf* relations do not have equivalents in OWL. Therefore, new properties in the RDDOnto namespace have been created to map them. Another alternative is to reuse other ontologies, as it has been done with Dublin Core. In this case mereological (*IsPartOf*) and quality (*Is*) notions are needed. For instance, they can be reused from the DOLCE [11] foundational ontology. For *IsPartOf* the equivalent is dolce:part-of and for *Is* it is dolce:has-quality. However, in the current RDDOnto version, the alignment with foundational ontologies like DOLCE has only been considered but not implemented.

Table 2. Mappings for relationships in the *Genealogy*, *Types* and *Membership of Sets* groups

RDD relation	Short description	RDF
IsTypeOf	Builds the hierarchy of term types	rdfs:subClassOf rdfs:subPropertyOf
IsA	Relates an instance term to its type	rdf:type
HasDomain	Defines the source term type for relations	rdf:domain
HasRange	Defines the target term type for relations	rdf:range
IsMemberOf	The RelatingTerm from Member to Set	rdfs:member

RDD relation	Short description	OWL
Is	Relates resources to ascribed qualities	rddo:hasQuality
IsEquivalentTo	Relates two equivalent terms	owl:equivalentClass owl:equivalentProperty owl:sameIndividualAs
IsOpposedTo	Relates two opposite terms	owl:disjointWith (owl:complementOf)
IsPartOf	Relates a terms that is part of another term	rddo:isPartOf
IsAllowedValueOf	Relates an instance terms that is allowed value of a type term	Inverse of owl:oneOf
HasType	The RelatingTerm from Archetype to Type	Inverse of rdfs:subClassOf rdfs:subPropertyOf
IsReciprocalOf	For relation terms defines the relation term that captures the inverse relation	owl:inverseOf

For the rest of the relationship groups, a part from *Genealogy*, there are no equivalent relations in the RDF/S plus OWL domain. This is due to the fact that these relationships are based on different kinds of semantics than those used in RDF/S and OWL. Therefore, the approach is to map them to new properties in the "rddo" namespace.

As has been said before, another alternative would be to reuse an ontology that captures the Context Model semantics that guide these relationships. However, we have not found one that directly fits so the creation of new properties is, for the moment, the alternative chosen.

Table 3 shows the mappings for the relationships in the *Family* group that generate the terms for an *ActType* following the Context Model, i.e. the begotten terms. There are also the reciprocal relations that are not shown but are mapped also mapped directly into the "rddo" namespace.

Table 3. Mappings for the generative relationships in *Family*, without considering reciprocals

RDD Attribute	OWL Property
BegetsContextType	rddo:BegetsContextType
BegetsAgentType	rddo:BegetsAgentType
BegetsResourceType	rddo:BegetsResourceType
BegetsTimeType	rddo:BegetsTimeType
BegetsPlaceType	rddo:BegetsPlaceType
BegetsRelatingTerm	rddo:BegetsRelatingTerm
BegetsQualityType	rddo:BegetsQualityType
BegetsActType	rddo:BegetsActType
BegetsStateType	rddo:BegetsStateType

Table 4 shows the mappings for some of the relationships that connect the begotten terms. The table shows only the relationships in the *Act Actype* context. The relationships among the terms for other *ActTypes* are mapped in the same way.

Table 4. Mappings for the *Family* relationships among the terms begotten from the *Act ActType*

RDD Attribute	OWL Property
HasCo-Agent	rddo:HasCo-Agent
IsAgentActingOn	rddo:IsAgentActingOn
IsAgentAtTime	rddo:IsAgentAtTime
IsAgentInPlace	rddo:IsAgentInPlace
IsResourceActedOnBy	rddo:IsResourceActedOnBy
HasCo-Resource	rddo:HasCo-Resource
IsResourceAtTime	rddo:IsResourceAtTime
IsResourceInPlace	rddo:IsResourceInPlace
IsTimeOfActingBy	rddo:IsTimeOfActingBy
IsTimeOfBeingActedOnOf	rddo:IsTimeOfBeingActedOnOf
HasCo-Time	rddo:HasCo-Time
IsTimeOfActingInPlace	rddo:IsTimeOfActingInPlace
IsPlaceOfActingBy	rddo:IsPlaceOfActingBy
IsPlaceOfBeingActedOnOf	rddo:IsPlaceOfBeingActedOnOf
IsPlaceOfActingAtTime	rddo:IsPlaceOfActingAtTime
HasCo-PlaceOfActing	rddo:HasCo-PlaceOfActing

Finally, there are the relationships that connect a concrete *ContextType* to the entities involved in this context. Table 5 shows them and their mappings.

Table 5. Mappings for the *ContextView* relating terms

RDD Attribute	OWL Property	RDD reciprocals	OWL Property
icoAgent	rddo:icoAgent	IsAgentInContext	rddo:IsAgentInContext
icoResource	rddo:icoResource	IsResourceInContext	rddo:IsResourceInContext
icoTime	rddo:icoTime	IsTimeInContext	rddo:IsTimeInContext
icoPlace	rddo:icoPlace	IsPlaceInContext	rddo:IsPlaceInContext
HasValue	rddo:HasValue	IsValueOf	rddo:IsValueOf

To conclude the mappings, it is also necessary to map RDD terms to web ontology concepts. The previous mappings only cover the attributes that relate them. This has been postponed until now because web ontology languages discern the RDD terms into three kinds: classes, properties and instances. The distinction is not made in RDD but it can be deduced from the term attributes.

If the term R*elationships* attribute includes *HasRange* or *HasRange* relationships, it is clear that this terms must be mapped to a rdf:Property. This is a necessary and sufficient condition because all terms referring to relations have at least one of this relationships.

Otherwise, the term is a class or an instance. It will be mapped to rdfs:Class if it has a *IsTypeOf* or if there is no *IsA* relationship. If there is an *IsA* relationship but not *IsTypeOf* relationship, then it will be mapped to an instance, i.e. rdf:Description. It can be noted that it is possible to have a term that has both *IsTypeOf* and *IsA* relationships that is mapped to rdfs:Class. Therefore, as specified in the OWL Overview [12], the concrete OWL ontology produced is an OWL Full one.

2. Implementation

The RDD to RDF/S and OWL mappings that have been established in Table 1 and Table 2 have been implemented in the RDDOntoParser [13]. It is a Java implementation of these mapping using regular expressions [14]. Regular expressions are used to define patterns that detect the RDD part of the mappings. When patterns match, the corresponding RDF is generated in order to build RDDOnto.

Finally, once attributes have been mapped, they are used to discern the processed term as a rdfs:Class, a rdf:Property or an instance, rdf:Description. The input of the RDDOntoParser is a plain text version of Table 3 - Standardized Terms of the RDD standard [17]. The output constitutes the RDDOnto web ontology [15]. For the other relationships a direct mapping to a new property with the same name in "rddo" namespace is implemented, as it has been defined in Table 3, Table 4 and Table 5.

However, these relationships do not remain isolated in the resulting ontology. As all RDD terms are defined using RDD, relating terms are defined using relationships in the *Genealogy* group. Therefore, RDDOnto includes information about domain and range restrictions, relationships hierarchical organisation, etc.

3. Checking RDD with RDDOnto

During the ontology development, ontology tools facilitated the detection of integrity problems in RDD. There were many references to undefined terms and inconsistencies between different parts of the standard. These problems were communicated to the MPEG-21 RDD working group [13] and the RDDOnto development process led then to a revision [16] of the then recently published RDD ISO/IEC standard [17].

First of all, there were some inconsistencies between the textual RDD terms definitions and a figure showing the hierarchy tree of RDD act types. These inconsistencies were detected by comparing the figure included in the standard with a drawing of the Act hierarchy generated automatically from RDDOnto using the Protégé [18] ontology editor and the OntoViz [19] ontology visualisation plug-in.

However, more important problems were related with the integrity issues of the standard. Some of the relationships and terms that were used in the terms definitions were not defined in it. They were added to RDDOnto. There were also some spelling errors. Table 6 summarises these integrity problems resolved in RDDOnto. The integrity checks were performed with the help of the OWL validator vOWLidator [20].

Table 6. Integrity problems corrected by RDDOnto

Relationships	**Terms**	**Spelling errors**
HasCoChangedResource	*ContextModelTermSet*	*PlaceOfCategorizeing*
icoInteractor	*TS_2*	*TimeOfCategorizeing*
IsInteractorInContext	*CategorizingEvent*	
IsInteractorWith	*Categorized*	
IsDescriptionOf	*RenderedAsFixation*	
IsInteractedWithBy		

However, the greatest problem, still not solved in the current standard, that has been found is about the particular approach to multiple inheritance that is explained in section A.11.1.2 of the RDD standard. This approach might be practical because it is based on object-oriented programming inheritance. On the other hand, it imposes a lot of work when RDD is put into practice. For instance, during license checking of rights, i.e. acts. In this case, a particular act is checked against available licenses to test if it is included, i.e. subsumed, by them.

When subsumption between acts is checked, all the parents are affected recursively. If the subsumption is between directly related acts it can be simple. When indirectly connected acts are checked, intermediate acts that are also affected by multiple inheritance make checking more complicated. Our proposed approach, also taken by most ontology languages, considers multiple inheritance as an intersection of sets (types). This approach makes checking easier because a type with multiple inheritance is subsumed by any of the super types.

Another testing facility once mapped to an OWL ontology is the consistency check provided by Description Logic (DL) [21] reasoners. OWL is a Description Logic so DL reasoners can be directly used in order to reason with OWL ontologies. The only limitation is that reasoners only deal with two of the three OWL sublanguages, i.e. OWL DL and OWL Lite but not OWL Full. As it has been said, RDDOnto is OWL Full so we have to change some of the mapped constraints that make it Full prior to feeding it into the DL reasoner. This has been done deactivating or changing some of the mappings in the RDDOntoParser and with the assistance of Protégé combined with the Racer DL reasoner. First, we have deactivated the "IsA" to "rdf:type" mapping in order to avoid OWL Classes or Properties that are instances of other classes as this is

not allowed in OWL DL and all terms are mapped to classes or properties. Second, in OWL there are not instance level relationships between classes. The RDD specific relationships, i.e. the ones shown summarized in Table 3, Table 4 and Table 5, are instance level relations so they must be ignored or converted into documentation properties in order to keep RDDOnto an OWL DL ontology. We have adopted the latter because the RDD relationships do not have any semantics as to OWL but they might be useful for documentation purposes.

Once we applied all these simplifications in order to make RDDOnto an OWL DL ontology, we applied automated consistency checks to RDDOnto obtaining 292 problems. All of them are due to inconsistencies between the classes and properties hierarchies that are accumulated along the hierarchy chains. The consequence is that many properties domains and ranges are inconsistent with the domains and ranges of the corresponding superproperties. For instance, the property *IsAgentActingOn* has domain *Agent*. The direct superproperty *IsRelativeOf* has domain *Relative* but *Relative* is not a superclass of *Agent* so there is an inconsistency in the *IsAgentActingOn* domain.

In order to resolve these inconsistencies in RDDOnto, a detailed analysis of them and the classes and properties hierarchies has been carried out. In many cases the necessary changes to harmonize the hierarchies has been to relax the domains of the following properties to owl:Thing, which subsumes all the other OWL classes: *isRelativeOf*, *IsBegottenBy*, *IsAscribedTo*, *Is* and *IsQualityOf*.

Moreover, other minor changes have been carried out in order to remove the remaining inconsistencies. First, it has been necessary to exchange the domain and range of the *IsAgentTypeBegottenBy* property. Then, we have changed the domain of *IsAClassFromSet* to *Instance*. Finally, the *IsSetWithClassOf* and *IsAClassFromSet* have been moved in the properties hierarchy in order to make them direct subproperties of *IsAscribedTo*. With all these changes we have made RDDOnto consistent from the point of view of the OWL semantics, which captures a great part of the original RDD semantics and thus corrects at least many of its inconsistencies. Nevertheless, we still need to analyse how these changes should be moved to the RDD standard. All the versions of RDDOnto are available at [15].

4. Using RDDOnto

As it has been introduced, to have RDD formalised as an OWL ontology provides many advantages. The following sections describe some of them. First there are the integration facilities provided by web ontologies that are used to integrate RDD in OWL form with other parts of MPEG-21, which are also mapped to OWL. Then, once in this integrated ontological framework, ontology-enabled tools like semantic query engines and DL reasoners facilitate the implementation of MPEG-21 tools.

4.1. Ontological framework for integration with MPEG-21 REL

The rights statements representation part of MPEG-21 is composed of the RDD, which defines the terms as it has been shown, but it also composed of the Rights Expression Language (REL). The easiest way of explaining this is through a simile: the RDD provides the definition of the words while the REL provides a language to put this words together in order to build statements.

However, this intended complementarity is difficult to put into practice from the MPEG-21 standard specifications of REL and RDD. While the RDD is defined as an ontology, although not using a formal ontology language as it has been shown, REL is defined on the basis of a set of XML Schemas. This makes the integration between them very tricky.

Our approach has been to take profit from the integration facilities provided by web ontologies. The REL XML Schemas have been also mapped to OWL and then easily integrated with RDDOnto using the OWL semantic relations for equivalence and inclusion: *subClassOf*, *subPropertyOf*, *equivalentClass*, *equivalentProperty*, *sameIndividualAs*, etc. In order to map the XML Schemas to OWL and XML instances to RDF the XSD2OWL and XML2RDF mappings [22] have been applied. The former is a generic mapping from XML Schemas to OWL ontologies, which has been also applied to map another REL called ODRL to OWL ontologies [23]. The later maps XML instances, e.g. MPEG-21 licenses, to RDF taking into account the previous mappings from the XML Schemas used by the instance to their corresponding OWL ontologies. Thus, we get RDF versions of the licenses that are semantics-aware, i.e. they are connected to the ontologies that formalise the terms they use.

4.2. Semantic Query

Once the REL and the RDD were integrated, it was possible to develop ontology-enabled applications that take profit from their formal semantics. This has been used to implement MPEG-21 licenses management tools. For instance, the acts taxonomy in MPEG-21 RDD can be seamlessly integrated in order to facilitate

license-checking implementation, see Fig. 4. Consider the scenario: we want to check if our set of licenses authorises us to uninstall a licensed program.

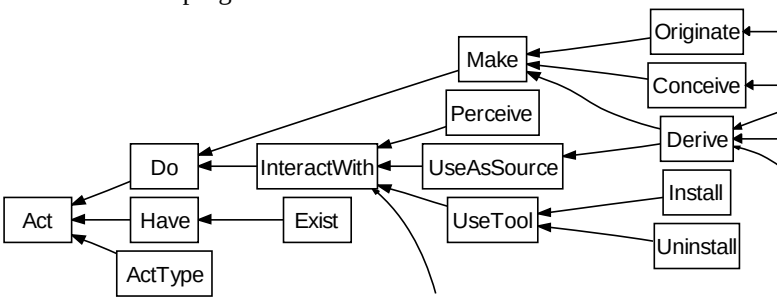

Fig. 4. Portion of the acts taxonomy in MPEG-21 RDD

If we use a purely syntactic approach like XPath over MPEG-21 XML licenses, there must be a path to look for licenses that grant the *uninstall* act, e.g. "//r:license/r:grant/mx:uninstall". Moreover, as it is shown in the taxonomy, the *usetool* act is a generalisation of the *uninstall* act. Therefore, we must also check for licenses that grant us *usetool*, e.g "//r:license/r:grant/mx:uninstall". And successively, we should check for *interactwith*, *do* and *act*. All this must be done programmatically, the XPath queries are generated after we check the RDD ontology.

However, if we use semantic queries, the existence of a license that grants any of the acts that generalise *uninstall* implies that the license also states that the *uninstall* act is also granted. This is so because, by inference, the presence of the fact that relates the license to the granted act implies all the facts that relate the license to all the acts that specialise this act.

Therefore, it would suffice to check the semantic query "//r:license/r:grant/mx:uninstall". If any of the more general acts were granted it would match. For instance, the XML fragment:

- /r:license/r:grant/dd:usetool

implies the fragments:

- /r:license/r:grant/dd:install
- /r:license/r:grant/dd:uninstall.

4.3. Usage against license checking using DL reasoners

There are other application development facilities more sophisticated than the semantic queries benefits shown before. One of the most promising tools is Description Logics. OWL is based on DL so it can be directly fed into DL classifiers. Classifiers are specialised logic reasoners that guarantee computable results. DL classifiers are used with RDDOnto in order to automatically check IP uses against the use patterns specified in IP agreements or offers. This facilitates checking if a particular use is allowed in the context of a set of licenses or finding an offer that enables it, once an agreement is reached.

DL classifiers can be directly reused so there is no need to develop ad-hoc applications to perform this function. In order to do that the following steps are followed:

1. First of all, the usage event that is going to be checked is modelled as instance data using RDDOnto and the REL ontology. For instance: "USER1 is trying to access a given video stream from a given streaming server at 9:30:10 UTC on 2005-04-10". The streaming server implements digital rights management so it inquires the license manager if the current usage instance is permitted. In order to do that, the streamer models this usage and sends it to the license manager, e.g. as a RDF/XML serialisation.
2. The license manager contains licenses also modelled using the RDD and REL ontologies. However, they are modelled as classes. These licenses define usage patterns and the conditions to be fulfilled in order to be authorised. When the pattern refers to a particular instance, e.g. the particular video stream on the previous example, the license class is defined by OWL *hasValue* constraints. This kind of constraints defines a class of instances that are related by a given property to a particular instance. When the constraint on the usage pattern is more general, e.g. a set of users of which the USER1 in the example is a member, OWL constraints like *allValuesFrom* or *someValuesFrom* are used. They are defined for the given property and to the corresponding class, e.g. an enumerated class containing USER1.
3. The license manager checks if there is any license that grants a usage pattern that subsumes the usage instance. This can be performed easily and efficiently using a DL classifier. However, there are some problems that should we resolved before. The usage patterns may define time intervals that should be tested against the usage time point. In order to check if the time point is included in the time interval, we must use a DL classifier capable of dealing with custom data types reasoning [24]. Then, the time

interval is translated to a real interval (pointInTime.≥[20050401] ^{real} ∧ ≤[20060401] ^{real}) and the time point to a real (pointInTime.=[20050410.093010] ^{real}).

4. After applying the previous adaptations, subsumption is computed. The usage might be classified in one or more usage patterns. In this case it is tested if the usage pattern is contained in a grant or and offer. In the first case, the condition is checked and if it is satisfied the license manager tells the streaming server that the use is authorised. Otherwise, the use is not authorised. In the second case, the offer that has been found may be accepted or negotiated in order to achieve the usage.

5. Conclusions

We have presented RDDOnto, an ontology for the MPEG-21 RDD. Its added value over other initiatives to implement rights data dictionaries is that it is based on applying an ontological approach. This is done by modelling the RDD standard using ontologies. Ontologies allow that a greater part of the standard is formalised and thus more easily available for implementation, verification, consistency checking, etc.

RDDOnto demonstrates the benefits of capturing the RDD semantics in a computer-aware formalisation. First of all it has been possible to check the standard integrity and consistency with the support of ontology-aware tools that facilitate this. Then, it has been possible to integrate RDD with another MPEG-21 standard part, the Rights Expression Language (REL) in a common ontological framework. This framework facilitates the implementation of MPEG-21 tools. We have shown our achievements using semantic query engines and Description Logic reasoners for license searching, validation and checking. The ontological approach has also made possible, as shown in previous work, the development of advanced Digital Rights Management systems that integrate this tools in order to build semantic information systems [25,26] and intelligent agents for assisted rights negotiation [27].

6. Future Work

We expect to take profit from the abstraction and integration facilities of formal ontologies in order to cope with the RDD standard criticisms that are lately arising. First of all, RDDOnto is being used in order to extend RDD in a consistent and more informed way. There are some communities that have detected unsatisfied requirements in the current RDD [28]. This is completely normal as it is impossible to cope with all the requirements of a community as big as the one that might be interested in the MPEG-21 standard.

The real problem in this kind of situations is the standard rigidity to changes. This is even more critical when the standardisation domain is as abstract and dynamic as the Digital Rights Management domain. The MPEG-21 RDD standard specifies mechanisms for standard extension. However, it is difficult to put these mechanisms into practise. The size of the standard makes it very complicate to people outside its standardisation process to manipulate and extend it in order to satisfy their particular needs. This is why we have started to use RDDOnto as an assistance mechanism for RDD testing of requirements. There is currently an initiative inside MPEG-21 that tries to carry out an experiment about how to make consistent the concept of “Work” into RDD [29].

“Work” is understood as it is common in the Intellectual Property domain: “Any distinct (unique) artistic or intellectual creation”. First of all, RDDOnto is used together with ontology rendering tools in order to navigate the RDD hierarchy of concepts, detect the part of it where the new concept might be situated and even produce a graphical drawing of it, as it is shown in Fig. 5.

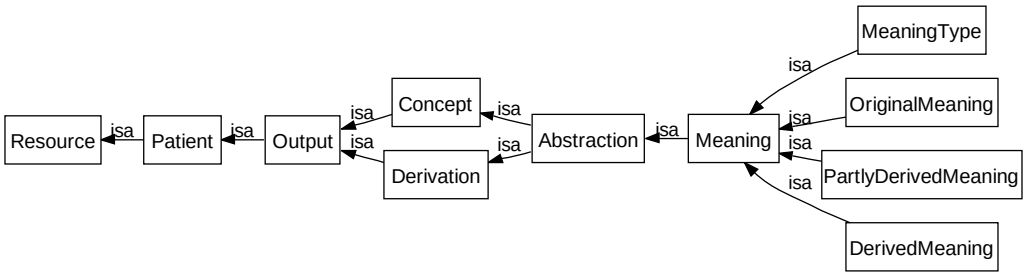

Fig. 5. Hierarchy of concepts related to "Work"

This might seem a quite trivial functionality of RDDOnto but we have proven its usefulness as some of the drawings generated have had great acceptance among the MPEG-21 development community. This is due to the lack of graphical views of the standard, although it is an ontology, and the difficulties to generate them because the ontology language used is not easily computer processable.

Additionally, it is possible to take profit from the RDDOnto formal semantics combined with reasoning engines. It is easier to detect where to put new concepts using DL classifiers. The new concept is defined independently from the rest of RDD concepts using OWL constraints. Then, the classifier is responsible for situating the concept in the hierarchy, taking into account any inconsistency. We are exploring these in order to face the mentioned experiment of RDD enhancement with the "Work" concept.

Another future line is to exploit the integration possibilities of OWL in order to connect RDDOnto to more general Intellectual Property ontologies, e.g. IPROnto [30], or rights data dictionaries of other rights expression languages like ODRL. The objective here is to build an ontology-based framework that allows integrating these initiatives, making them interoperable and enrich them with the possibilities offered by formal ontologies. This might lead to levels of interoperability that allow combining different RELs and RDDs in a totally uncoupled way.

Acknowledgements

This work has been partly supported by the Spanish administration (AgentWeb, TIC 2002-01336). The final part of the work presented was developed within VISNET, a European Network of Excellence (http://www.visnet-noe.org), funded under the European Commission IST FP6 program.

References

[1] Moving Picture Experts Group (MPEG) ISO/IEC/JTC1 SC29/WG11, http://www.chiariglione.org/mpeg/index.htm.
[2] Walle, R.v.d.: The MPEG-21 Book, John Wiley & Sons, Chichester, 2005.
[3] Wang, X.; DeMartini, T.; Wragg, B. and Paramasivam, M.: The MPEG-21 Rights Expression Language, IEEE Transactions on Multimedia 7(2), 2005.
[4] Rust, G. and Barlas, C.: The MPEG-21 Rights Data Dictionary, IEEE Transactions on Multimedia 7(2), 2005.
[5] Iannella, R.: Open Digital Rights Language (ODRL), Version 1.1. Technical report, World Wide Web Consortium, 2002
[6] Dean, M. and Schreiber, G. (eds.): OWL Web Ontology Language Reference, W3C Recommendation, 2004. http://www.w3.org/TR/owl-ref.
[7] Lassila, O. and Swick, R.R. (eds.): Resource Description Framework (RDF), Syntax Specification, W3C Recommendation, 2004. http://www.w3.org/TR/rdf-syntax-grammar.
[8] Brickley, D. and Guha, R.V. (eds.): RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, 2004. http://www.w3.org/TR/rdf-schema.
[9] Sowa, J.F. (ed.): Principles of Semantic Networks: Explorations in the Representation of Knowledge. Morgan Kaufmann Publishers, San Mateo, CA, 1991.
[10] Dublin Core Metadata Element Set, Version 1.1: Reference Description, http://dublincore.org/documents/dces.
[11] Masolo, C.; Borgo, S.; Gangemi, A.; Guarino, N. And Oltramari A.: Ontology Library, v1.0. IST Project WonderWeb, Deliverable 18, 2003. http://wonderweb.semanticweb.org/deliverables/D18.shtml.
[12] McGuinness, D. L. and van Harmelen, F.: OWL Web Ontology Language Overview. W3C Recommendation, 2004. http://www.w3.org/TR/owl-features.
[13] García, R.; Delgado, J.; Rodríguez, E.; Llorente, S. and Gallego I.: RDDOnto, Rights Data Dictionary Ontology Version 2, ISO/IECJTC1/SC29/WG11/M10423, December 2003.
[14] Java 2 Platform Std. v1.4.2, Package java.util.regex, http://java.sun.com/j2se/1.4.2/docs/api/java/util/regex/package-summary.html.
[15] MPEG-21 REL and RDD Ontologies, http://dmag.upf.edu/ontologies/mpeg21ontos.
[16] Barlas, C. & Rust, G.: Rights Data Dictionary, Technical Corrigendum 1, ISO/IEC JTC 1/SC 29/WG 11, 2005.
[17] ISO/IEC 21000-6:2004, MPEG-21 Part 6: Rights Data Dictionary (RDD), ISO/IEC JTC 1/SC 29, 2004.
[18] Protégé Project, http://protege.stanford.edu.
[19] OntoViz, Protégé ontology visualization plug-in, http://protege.stanford.edu/plugins/ontoviz/ontoviz.html.
[20] vOWLidator, http://projects.semwebcentral.org/projects/vowlidator
[21] Pan, J.Z.: Description Logics: Reasoning Support for the Semantic Web. PhD thesis, School of Computer Science, The University of Manchester, 2004.
[22] Gil, R.; García, R. and Jaime Delgado: An interoperable framework for IPR using web ontologies. Legal Ontologies and Artificial Intelligence Techniques, LOAIT'05. IAAIL Workshop Series, Wolf Legal Publishers, pp. 135-148, 2005.
[23] García, R.; Gil, R.; Gallego, I. and Jaime Delgado: Formalising ODRL Semantics using Web Ontologies. Open Digital Rights Language Workshop, 2005.
[24] Pan, J.Z. and Horrocks, I.: OWL-Eu: Adding Customised Datatypes into OWL. In Euzenat, J. and Gómez-Pérez, A. (ed.): Proc. of Second European Semantic Web Conference (ESWC 2005). Springer-Verlag, LNCS Vol. 3532, pp. 153-166, 2005.
[25] García, R.; Gil, R. and Delgado, J.: Intellectual Property Rights Management using a Semantic Web Information System. In R. Meersman and Z. Tari (ed.): OTM Confederated International Conferences, CoopIS, DOA, and ODBASE '04. Springer-Verlag, LNCS Vol. 3291, pp. 689-704, 2004.
[26] García, R. and Delgado, J.: Brokerage of Intellectual Property Rights in the Semantic Web. In Cruz, I.F.; Decker, S.; Euzenat, J. and McGuinness, D.L. (ed.): Proc. Semantic Web Working Symposium. Stanford University, California, pp. 245-260, 2001.
[27] Gil, R.; García, R. and Delgado, J.: Delivery context negotiated by mobile agents using CC/PP. In Mobile Agents for Telecom Applications, (MATA'03). Springer-Verlag, pp. 99-110, 2003.
[28] Gauvin, M.; Delgado, J.; García, R. and Rodríguez, E.: MPEG 21 RDD spec vs. MPEG 21 Requirements and Vision documents Review. ISO/IECJTC1/SC29/WG11/M11875, Busan, KR, 2005.
[29] Gauvin, M.: Workplan for Core Experiment on RDD Abstraction & Resource. ISO/IECJTC1/SC29/WG11, Poznan, Poland, 2005
[30] Delgado, J.; Gallego, I.; Llorente, S. and García, R.: IPROnto: An Ontology for Digital Rights Management. In Bourcier, D. (ed.): Legal Knowledge and Information Systems. IOS Press, Amsterdam, Frontiers in Artificial Intelligence and Applications Vol. 106, 2003.