Making the Semantics of ODRL and URM Explicit Using Web Ontologies

Andreas Kasten and Rüdiger Grimm

University of Koblenz-Landau,
Universitätsstr. 1, 56070 Koblenz, Germany
{andreas.kasten, ruediger.grimm}@uni-koblenz.de

Abstract. The Open Digital Rights Language (ODRL) aims at providing a foundation for interoperable DRM systems. Since its specification does not define any formal semantics, ODRL licenses are prone to misinterpretation. In order to address this issue, this paper presents a transformation of the ODRL vocabulary to the semantic web language OWL. By doing this, the semantics of an ODRL license is made explicit. Based on the resulting ontology, a second one is created for Usage Rights Management (URM). URM is a tool for informing users about their usage rights of their digital media files. URM is based on ODRL. The URM ontology therefore represents a possible application of the ODRL ontology.

Keywords: ODRL, URM, DRM, Semantic Web, Ontology

1 Introduction

The Open Digital Rights Language (ODRL) is a rights expression language for describing rights over physical or digital goods. ODRL aims at creating a foundation for more interoperability between DRM systems. It does not focus on a particular use case and can therefore be used within different applications. The current version of ODRL is 1.1, which is specified in [1]. The specification not only describes the language model of ODRL but also its XML serialization. ODRL is based on a license concept with each license being a separate XML document.

The ODRL vocabulary defines several entities for describing such licenses as well as their relations to each other. The syntax of these relations is described within an XML schema. However, only a small part of the semantics of these relations is encoded within this schema as well but most of it is described using a natural language. A formal semantics which enables an automatic processing of an ODRL license is not defined within the ODRL specification. Therefore, each ODRL license is open for interpretation concerning its actual meaning. Since ODRL aims at providing a foundation for more interoperability between different DRM systems, such an ambiguity should be avoided as much as possible.

ODRL follows an open design approach and is not restricted to any particular application. It can be used for the enforcement of DRM policies as well as for just describing what an end user is allowed to do with a particular good. An example of the latter use case is Usage Rights Management (URM) that is described in [2]. URM is based on ODRL and also aims at providing a general purpose system rather than a particular application.

This paper presents the first steps toward a formal semantics for ODRL using semantic web ontologies. This allows having both explicitly expressed semantics as well as an interoperable license format. Based on the resulting ODRL ontology, a second one is created for the URM system. The URM ontology also represents an example of how the ODRL ontology can be used.

The paper is organized as follows: Section 2 outlines similar approaches for obtaining a formal semantics for ODRL. Section 3 gives a short description of some selected semantic web technologies. The ontology for ODRL is then presented in section 4. Section 5 introduces a foundation model and an ontology for URM, and section 6 outlines two possible use cases for that ontology. Finally, section 7 gives a short conclusion.

2 Related Work

Pucella and Weissman [3] give a formal semantics for a fragment of ODRL based on many-sorted first-order logic with equality. The primary goal of their formal semantics is to create a tool for deciding whether or not a particular permission or prohibition follows from a set of ODRL licenses. They also describe an algorithm for performing such queries.

A formal semantics for a fragment of ODRL based on a finite-automata like structure is presented by Holzer et al. in [4]. The automaton models the actions that end users are allowed to perform on a particular asset at each point of time. Each automaton represents exactly one asset and arbitrary users and permissions. Users are represented as labels of states and permissions are represented as state transitions.

Both approaches map a part of the ODRL vocabulary to a more formal representation. In doing so, several possible ambiguities of the ODRL licenses can be eliminated. However, in both approaches the formal semantics is defined separately from an ODRL license. Therefore, the interpretation of an ODRL license requires several steps. Firstly, the validity of the ODRL license as an XML document has to be verified. Secondly, the license has to be transformed into another representation used by the formal semantics. Finally, the actual interpretation can be done based on the transformed license. In order to ease the process of the automatic interpretation of an ODRL license, a formal semantics should be included into the license. That way, the license can be directly interpreted and the transformation process becomes obsolete.

Delgado et al. [5] present a mapping of the ODRL model to the semantic web language OWL [6]. In doing so, the semantics of the ODRL model that is hidden within the ODRL specification as well as within the XML schema is made explicit. The actual mapping process is divided into three steps. The first step consists of an automatic transformation of the XML schema into an

OWL representation. Since the XML schema only contains very little semantic information, the result of the transformation is enriched with further semantics based on a manual interpretation of the ODRL specification. Finally, the resulting OWL representation is linked to the OWL model of the rights expression language IPROnto. IPROnto [7] has a much more expressive vocabulary than ODRL and has a formal semantics directly encoded within its OWL model. By linking the OWL version of ODRL to that of IPROnto, the formal semantics of IPROnto can also be used for interpreting an ODRL license.

However, the approach of Delgado et al. has two major drawbacks. Since the XML schema of ODRL only covers very little semantics, its direct use as a foundation for an ontology can easily create an unnecessary burden. The resulting ontology will be too closely related to the syntax of the XML schema rather than to the actual meaning of ODRL's language model. In order to circumvent this issue, a manual interpretation of the ODRL specification should be considered a first step.

Secondly, an interpretation of an ODRL license always requires the IPROnto ontology. As shown by Pucella and Weissman [3] as well as by Holzer et al. [4], most parts of the ODRL language model can be directly used for interpreting an ODRL license without the need of any additional ontologies. Therefore, it is desirable to have an ODRL ontology that enables such interpretations.

3 Semantic Web Basics

The semantic web aims at providing machine-readable metadata for arbitrary resources. A resource can be anything, ranging from web pages to MP3 files or even persons. One of the mostly used description languages for the semantic web is the Resource Description Framework (RDF) [8]. RDF describes resources using simple statements that consist of subjects, predicates, and objects. Subjects are the resources being described, predicates define the attributes of the resources and objects are the corresponding values. An object can either be an atomic value called a literal or another resource. Every resource in RDF is uniquely identified by means of a not necessarily dissolvable URI.

RDF can only describe the actual data. For extracting any meaning from the data, a corresponding schema model is needed. Such a schema model can be specified with the Resource Description Framework Schema (RDFS) [9] or the Web Ontology Language (OWL) [6]. RDFS allows the definition of classes and properties. Classes are general concepts that group together similar resources and can be used to create a simple taxonomy. Properties are the attributes of a class and further describe its characteristics. Resources always have at least one type that corresponds to a class. OWL is based on RDFS and has much more expressiveness. For example, it is possible to define a property to be symmetric or transitive.

A schema model defined in RDFS or OWL is called an ontology. An ontology describes classes and properties as well as their relations to each other. It can be used to conclude further information from the explicit given RDF data. This automatic process of concluding information is called reasoning. Reasoning is based on evaluating the characteristics of classes and properties. To be able to perform reasoning on RDF data, a machine needs to have access to the corresponding ontology. Many ontologies are freely available online. This not only eases the process of obtaining an ontology but it also allows reusing specific ontologies for similar use cases. This in turn allows different applications to better understand each other which can be considered a basis for an interoperable semantic web.

4 Putting More Semantics into ODRL

The ODRL vocabulary is divided into the ODRL expression language and the ODRL data dictionary. The expression language describes the main part of the ODRL vocabulary, namely the ODRL foundation model. This model defines the key semantics of ODRL. Its core entities are `Asset`, `Party`, and `Rights`. Every ODRL license must at least contain these three entities. `Asset` refers to the good that is described within the license; `Party` refers to end users and `Rights Holders`. `Rights` is the actual license that describes what a `Party` is allowed or is not allowed to do with the `Asset`.

The data dictionary describes several other models that further specify the semantics of the foundation model. One of these models is, for example, the permission model that contains the actual `Permissions` used within an ODRL license. However, an ODRL license does not have to use the vocabulary of ODRL's default data dictionary. It is also possible to define an own data dictionary and use it within a license.

4.1 Semantics of ODRL

ODRL follows an open design approach which allows it to be used for different applications. Each application can use a different implementation of an ODRL license based on its own data dictionary. This makes it impossible for the ODRL specification to cover the semantics of all possible ODRL licenses. Instead, the application itself has to specify the semantics of the concrete ODRL license it actually uses. This is done by writing an ODRL profile. Such a profile describes what entities are used within the license and what those entities mean.

The OWL ontology presented in this paper covers the key semantics of the ODRL foundation model as well as the semantics of the models described in the ODRL data dictionary. Since the latter can vary between different application-based implementations, it is restricted to the semantics given in the ODRL specification. Further semantics must be modeled using an additional ontology analog to an ODRL profile.

4.2 An Ontology for ODRL

The ODRL ontology presented in this paper is modeled in OWL [6] and based on a student's thesis [10] as well as on interpretations that could be drawn from reading both the ODRL specification and the XML schema. All entities described within the specification were modeled as OWL classes. This covers both the entities of the expression language and of the data dictionary. Possible relations between two entities were mapped to a corresponding OWL property. Each property was associated with a meaningful name that describes the semantics of the relation. Since neither the XML schema nor the ODRL specification contain an explicit label for every possible relation, the names of the OWL properties are solely based on an interpretation of the semantics of the ODRL models.

The following subsections outline only a small part of the resulting ODRL ontology because describing it in all its details goes beyond the scope of this paper. The described parts are the foundation model and the permission model, the latter one being an example of the ontology's conceptual structure. ODRL's constraint model and its requirement model are constructed similarly to the permission model.

The ODRL Foundation Model. Figure 1 shows an OWL transformation of the ODRL foundation model. All entities of ODRL that do not have any semantics were left out. This includes the security entities `Digital Signature` and `Encryption Digest/Key` as well as the entities `Context` and `Rights`.

`Digital Signature` and `Encryption Digest/Key` are only used when transferring a license and do not account for its actual content description. `Context` is used for describing other entities such as `Party` or `Asset` by means of further descriptive elements. These elements are, for example, `Name`, `URI`, or `Date`. Since they are only used for describing other entities, they can be interpreted as attributes of such and are therefore modeled as OWL properties.

`Rights` is just a container that groups together all those entities that are used for forming a specific `Offer`, an `Agreement`, or a `Revocation`. To highlight the actual meaning of `Rights`, a new OWL class `License` was introduced as a superclass of `Agreement` and `Offer`. All relevant ODRL entities for describing both `Agreements` and `Offers` were linked to `License` via corresponding OWL properties. The class `Revocation`[1] was defined to be separate from `License` and has its own properties.

The foundation model depicted in figure 1 can be read as follows: A `License` is about an `Asset` and grants `Permissions`. If the `License` is an `Agreement`, it is attested by one or more `Parties`. Since `Rights Holders` are also `Parties`, an `Agreement` can be attested by a `Rights Holder`, too. However, if the `License` is an `Offer`, it must be provided by a `Rights Holder`. `Parties` that are not `Rights Holders` cannot provide an `Offer`. `Revocations` can be performed by any `Party`. A `Revocation` can withdraw a complete `License` or only specific `Permissions`.

`Permissions` can be further restrained by `Constraints`, `Conditions`, and `Requirements`. However, the ODRL specification does not define any specific `Conditions`. Instead, `Constraints` shall be used for further describing a `Condition`.

[1] Note that the ODRL entity Revoke was renamed to the OWL class Revocation in order to only use nouns as class names.

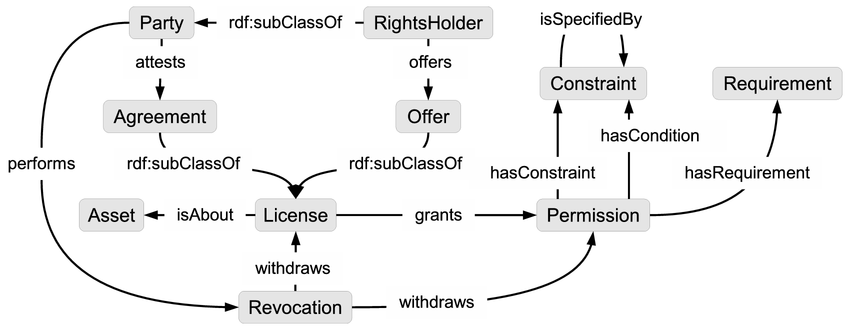

Fig. 1. The ODRL foundation model.

Since the ODRL entity `Condition` thus just links `Permissions` to `Constraints`, it was modeled as an OWL property rather than a separate OWL class.

The ODRL Permission Model. The permission model defines concrete permissions that can be used within an ODRL license. Figure 2 shows an excerpt of an OWL transformation of this model. The class `Permission` is the core of the model and defined as a superclass for all concrete permissions such as `Play` or `Print`. These concrete permissions are further grouped to additional classes according to their semantic similarity. For example, both the permissions `Play` and `Print` share the same superclass `Usage` which in turn is a direct subclass of `Permission`. Defining such intermediary classes like `Usage` enriches the semantics of the permission model. For readability, the property `rdf:subClassOf` was left out in figure 2. Each arrow represents such a relationship.

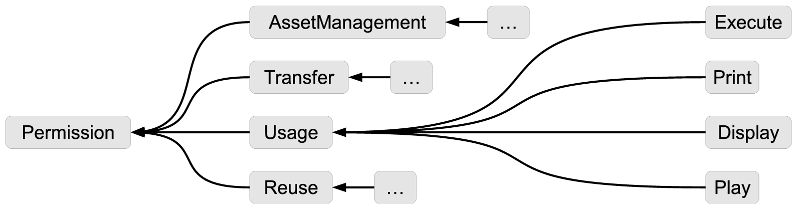

Fig. 2. The ODRL permission model.

Additional Ontologies. The ODRL specification primarily focuses on defining the main entities of a license but does not cover all the details that are required for actually describing such an entity. For example, possible attributes of a Party are not further given by ODRL itself. Instead, the specification suggests using additional vocabularies for specifying this entity.

Since the semantic web is meant to be an interoperable web of meaningful data, it is common to reuse already existing ontologies when creating a new one. By using the same ontology, different applications are able to better understand each other. The reused ontologies often define general purpose vocabularies like such for describing currencies or for describing persons and organizations. In order to achieve a broader interoperability with different applications, the ODRL ontology presented in this paper also imports some additional ontologies.

One imported ontology is the FOAF ontology[2] that was designed for describing persons, organizations, and groups of such. The FOAF ontology contains a class `Agent` which in turn has the subclasses `Person`, `Organization`, and `Group`. `Agent` is used for describing entities that are somehow actively involved in something. For example, the participants of a meeting can be considered to be its `Agents`. The ODRL vocabulary contains the entity `Party`, which is rather similar to `Agent`. A `Party` can also be a person, an organization, or a group of these. Furthermore, a `Party` is somehow associated with an ODRL license. Therefore, the entity `Agent` is used for describing `Parties`. However, if the expressiveness of the FOAF ontology is not sufficient for describing a particular `Party`, additional ontologies may also be used.

Another ontology used by the ODRL ontology is the Time Ontology in OWL[3]. This ontology is able to describe general time concepts such as `Instants`, `Intervals`, or `Durations`. `Instant` is a specific point in time, `Interval` represents a range in time and has a defined start and a defined end, and `Duration` constitutes a specific length of time without specifying any start or end point. These entities have a much more expressiveness than simple character strings that are conform to the ISO 8601 standard. Therefore, the ODRL ontology refers to an entity of the Time Ontology, whenever a temporal description is required.

5 A Semantic Model for URM

Usage Rights Management (URM) [2] is an ODRL-based DRM system especially designed for end users. Unlike ODRL, URM focuses solely on digital media files. Users often have several of these files from different origins stored on their computers. Some of them can be created through digitizing a physical object like a book or a CD; others can directly be bought via an online shop. Each of the media files has different usage rights attached to it, depending on how it was actually obtained. This often results in a confusing situation for the users, leaving them uncertain about what they are actually allowed to do with a particular media file.

To solve this issue, the usage rights of each media file should be made explicit and stored somewhere. Ideally, this is done by having a kind of license for each

[2] See http://www.foaf-project.org (last accessed: 30 May 2010).
[3] See http://www.w3.org/TR/owl-time/ (last accessed: 30 May 2010).

file that is signed by its origin. An online shop, for example, could provide such a license for each sold media file. With a signed license like this, users would have an indisputable proof of that they are allowed to use a particular media file in a specific way. However, a singed license is not always available for every media file.

URM is an informational system for helping users to manage their usage rights without having a signed license. The system assumes that users want to behave legally and do not want to use their media files in a way they are not allowed to. URM is based on unsigned licenses created by the users themselves. Each such license refers to a single media file and contains all the usage rights associated with that file. Furthermore, the URM license describes how the media file was obtained and names its origin. Such an origin can be anything that can be reviewed at a later date. The origin is assumed to be the basis for defining the usage rights of the media file. Users are allowed to use a particular media file if they have some kind of legal relationship to its origin. In that case, the origin of the media file is able to proof the legality of the usage rights.

Consider an MP3 file as an example. Such a file can be obtained by ripping a CD or by buying it directly from an online shop. Users who own a copy of the CD or bought the MP3 file at the shop themselves are undoubtedly allowed to play the file.[4] On the other hand, users that downloaded the file at an illegal peer-to-peer network won't have any usage rights at all.

Note that users can easily manipulate their URM licenses because they created the licenses themselves. However, this does neither affect the origin of the media files nor their usage rights. Therefore, a manipulation of an URM license does not benefit a user in any way.

5.1 The URM License

Since URM is based on ODRL, each URM license is also an ODRL license. More specifically, URM uses the ODRL entity `Party` for describing end users, `Permission` for describing the usage rights, `Asset` for describing the digital media file, and `pLocation` for referring to its origin. The current URM license definition does not provide any means for describing how the `Asset` was actually obtained from its origin. However, this mapping process is essential for URM. If an `Asset` cannot be traced back to its origin, it won't be associable to its usage rights that are bound to that origin. Consequently, users won't have any proof of that they are allowed to use the `Asset`.

Consider, for example, a PDF document as an `Asset` and a book as its origin. The PDF document can be created from the book by different means: the book can be digitized using an image scanner, it can be photographed using a digital camera, or it can be manually transcribed using a word processor. Each of these actions requires further steps to finally create a PDF document. Although all three resulting PDF documents are based on the same origin, they are very different in quality, file size, etc. Without any further information on

[4] This assumes that the ripping process of the CD itself is legal.

how each PDF document was actually created, it is almost impossible to trace it back to the original book. This also complicates associating the usage rights with the document. The PDF document may even be considered illegal if it is indistinguishable from a second PDF document of a dubious origin.

Therefore, it is desirable to describe the mapping process from an origin to a corresponding `Asset` as detailed as possible within the URM license. The more detailed this description is, the more plausible the origin is as a proof of the `Asset`'s usage rights.

5.2 An Ontology for URM

This section presents an ontology for URM that focuses on the description of the process of mapping an origin to an `Asset`. The ontology is designed in OWL and based on the ODRL ontology described in section 4 and the Event Ontology presented in [11]. Moreover, it makes use of several other ontologies like the BBC Programmes Ontology[5] for describing (web) radio and TV programs and the FOAF ontology[6] for describing persons. This section first outlines the basic concepts of the Event Ontology and then describes the different models of the URM ontology.

The Event Ontology. The Event Ontology is a general purpose ontology and part of the Music Ontology framework [11]. It is designed for describing events as happenings at a specific time and date at a specific location. The ontology defines six different classes with `Event` being the main class. Each `Event` is associated with a time and a date, represented by the class `TemporalThing`, locations, described with `SpatialThing`, active and passive participants, called `Agents` and `Factors`, respectively, and `Products` as the outcome of an `Event`.

`TemporalThing` can describe both instants and intervals. It is taken from the Time Ontology in OWL[7], which allows describing temporal information. `SpatialThing` is taken from the WGS84 Geo Positioning Ontology[8], which is designed for describing any kind of geospatial information. The class `Agent` is taken from the FOAF Ontology and can represent `Persons`, `Organizations`, or `Groups` of such. `Factors` are in some way required for an `Event` such as a particular tool, a specific material, or an abstract cause. Since a `Factor` can generally be anything, the Event Ontology does not further describe this class. Similarly, `Products` are not restrained to having any other characteristics than being the outcome of an `Event`.

The URM Foundation Model. The URM foundation model is the core of the URM ontology. It contains the ODRL entities `Asset` and `Permission` as well

[5] See http://www.bbc.co.uk/ontologies/programmes/ (last accessed: 30 May 2010).
[6] See http://www.foaf-project.org (last accessed: 30 May 2010).
[7] See http://www.w3.org/TR/owl-time/ (last accessed: 30 May 2010).
[8] See http://www.w3.org/2003/01/geo/wgs84_pos (last accessed: 30 May 2010).

as the entities `Process`, `Origin`, and `ExternalLicense`. According to section 5.1, `Permissions` are bound to the `Origin` and `Process` can be interpreted as a function, taking an `Origin` as input and producing an `Asset` as a result. Since the `Asset` not only depends on its `Origin` but rather on the `Process`, there is no direct relation between `Asset` and `Origin`.

`ExternalLicense` is used if an additional document or even a singed license exists that directly links an `Asset` to its `Permissions`. In that case, `ExternalLicense` refers to an undoubtedly proof of the user being allowed to use the `Asset`. The entity is only used as a link to an external document and is therefore not further described within the URM ontology.

Figure 3 shows the resulting foundation model of URM. It only depicts the properties of the URM ontology; all properties that are already defined within the ODRL ontology are not shown. Since these properties can be used in the URM ontology as well, there is no need for defining new properties with the same meaning.

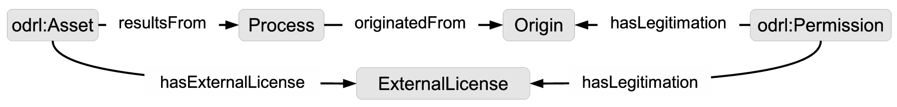

Fig. 3. The URM foundation model.

`Process` is modeled as a subclass of the entity `Event` from the Event Ontology. This allows a `Process` to be further specified using temporal and geospatial information as well as `Factors`, `Products`, and `Agents`. Following the core concept of the Event Ontology, `Origin` is defined as a `Factor` and `Asset` is defined as a `Product` of a `Process`. In order to make this interpretation more explicit, the properties of the URM foundation model are bound to corresponding properties of the Event Ontology. For example, the property `originatedFrom` is modeled as a subproperty of `hasFactor`, which links a `Factor` to an `Event`.

The URM foundation model only defines an abstract concept of how an `Asset` can be obtained from an `Origin`. It does neither cover any particular type of `Process` nor any specific type of `Origin`. For describing a certain `Process` or `Origin`, more specific entities must be used instead. The URM ontology currently considers nine different possibilities of obtaining an `Asset` from its `Origin`. These possibilities are shown in table 1. Each one is represented by a particular `Process` and a corresponding `Origin`. It is also possible to define additional `Processes` and `Origins`.

The URM Physical Media Object Model. `PhysicalMediaObject` represents a physical object of any media type such as books, CDs, video tapes, vinyl records, etc. An `Asset` is obtained from a `PhysicalMediaObject` via a

Process	Origin
`Digitization`	`PhysicalMediaObject`
`PersonToPersonCopy`	`PersonalProperty`
`FriendToFriendCopy`	`PersonalProperty`
`PeerToPeerDownload`	`PeerToPeerNetwork`
`WebPageDownload`	`WebPage`
`WebShopDownload`	`WebShop`
`Recording`	`Broadcast`
`Capture`	`LiveEvent`
`OriginalWork`	`OriginalWork`

Table 1. Processes and Origins defined in the URM ontology.

corresponding `Digitization`. For example, the `Digitization` of a book can be a scan, a photograph, or a manual transcription. Since the variety of possible media types is huge, a separate ontology should be used for describing these media types and their corresponding `Digitization`. As far as the authors know, such an ontology does not exist yet. Its creation will be further work. Until that ontology is available, the media type and the `Digitization` are described using simple properties of the entities `PhysicalMediaObject` and `Digitization`, respectively.

The URM Personal Property Model. In order to obtain an `Asset`, users may copy it directly from another `Person`. This assumes that the `Person` already has a copy of the `Asset`. This copy is described using `PersonalProperty`, which is just a subclass of `Asset`.

From the users' perspective, `PersonalProperty` is the `Origin` of their `Asset` and therefore described as such within their URM license. The `Person` that owns the `PersonalProperty` may have another URM license itself. This `Person` is actively involved in the copying process by allowing the users to perform the process. Since this makes the `Person` an `Agent` according to the Event Ontology, it is therefore modeled as such. For describing a particular `Person`, the FOAF Ontology is used. The resulting model is shown in figure 4.

The actual `Process` is described using `PersonToPersonCopy`. If the original owner is a close friend of the user, `FriendToFriendCopy` may be used instead. `FriendToFriendCopy` is a subclass of `PersonToPersonCopy` and especially interesting for the German copyright law, which allows a limited number of private copies of `Assets` between close friends.

The URM Download Model. The URM download model currently covers three different types of downloads and their corresponding `Origins`. The types of downloads are specified by `PeerToPeerDownload`, `WebPageDownload`, or `WebShopDownload`. `PeerToPeerDownload` describes a download from a `PeerTo-`

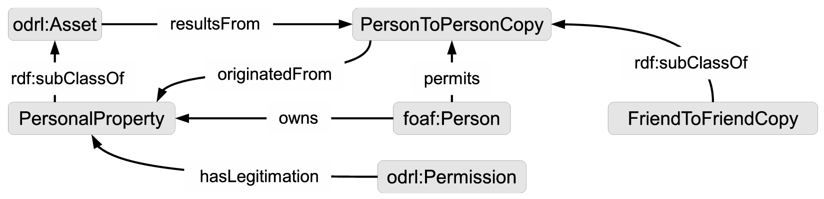

Fig. 4. The URM PersonalProperty model.

`PeerNetwork`. The entity may either be used for describing legal or illegal downloads. For example, Gnutella or the eDonkey network may be described as a `PeerToPeerNetwork` for explicitly stating that the usage of an `Asset` is illegal in any way.

`WebPageDownload` primarily describes free downloads and `WebShopDownload` specifies non-free downloads. The `Origin` of a `WebPageDownload` is a `WebPage`, which in turn is part of a `WebSite`. A `WebSite` can be a `PrivateHomePage`, a `WebPlatform`, or a `WebShop`. `PrivateHomePage` and `WebPlatform` are very similar. They both provide only free `WebPageDownloads` but differ in their trust level. A `PrivateHomePage` is rather dubious concerning the legality of its provided downloads, whereas a `WebPlatform` is considered to be more reputable and therefore provides a lower risk of possibly illegal downloads. An example of such a `WebPlatform` is flickr[9] for pictures.

A `WebShop` provides mainly non-free `WebShopDownloads` but can also provide free `WebPageDownloads`. Before a `WebShopDownload` can be performed, a user has to carry out a corresponding `Order`. This `Order` legitimates the actual `WebShopDownload`. Note that a `WebShopDownload` can also be a `WebPageDownload` at the same time and hence have a particular `WebPage` as well as a `WebShop` as its `Origin`. Figure 5 shows a fragment of the resulting download model.

Other URM Models. `Broadcast` describes a (web) radio or TV show that was aired at a specific date and time at a particular channel. The entity is taken from the BBC Programmes Ontology[10] and does not cover any kind of recording process. The actual recording process is described using `Recording`.

`LiveEvent` refers to an event, in which the user was directly involved somehow. For example, this can be a live concert of a rock band. `LiveEvent` is modeled as a subclass of the entity `Event` from the Event Ontology. An `Asset` can be created from a `LiveEvent` by capturing something that was present within that event. For instance, a visitor of a rock concert could record a part of the concert using some kind of recording device. The used device as well as its configuration at the time of the recording process is described using `Capture`. Since there are

[9] See http://www.flickr.com (last accessed: 30 May 2010).
[10] See http://www.bbc.co.uk/ontologies/programmes/ (last accessed: 30 May 2010).

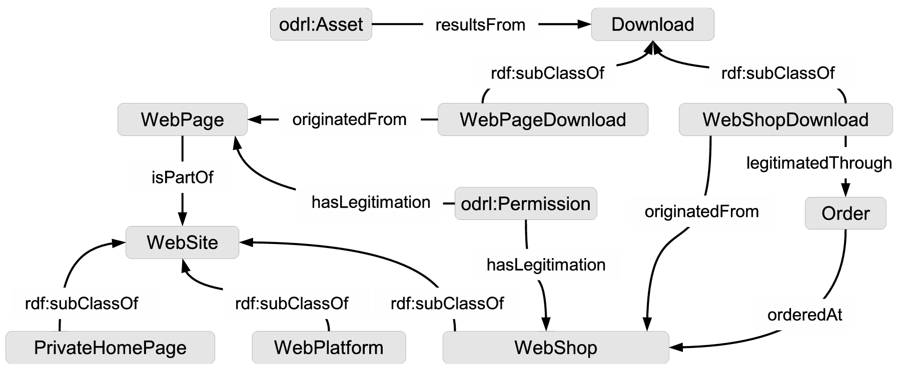

Fig. 5. A fragment of the URM download model.

many possibilities of the actual capturing process, `Capture` may be expanded by user-defined properties or subclasses.

`OriginalWork` refers to `Assets` that were completely created by users themselves. In this case, the `Asset` did not exist until it was created through a certain creation process. This makes it rather difficult to identify the actual `Origin` of the `Asset`, and to distinguish the `Origin` from the `Process`. For simplicity, `OriginalWork` is defined to be both the `Process` and the `Origin` of the `Asset`. This is done by making `OriginalWork` a subclass of `Process` and `Origin`[11]. `OriginalWork` currently considers only `Assets` that are completely new creations without using any previously existing works from other people. Describing such a re-use process is far too complex to be described within the URM ontology. Future work will address this issue.

6 Potential Use Cases

Both ODRL and URM follow an open design approach in order to support different applications. This section outlines two possible applications for the URM ontology described in the last section. Since URM is based on ODRL, the following applications describe also possible use cases of the ODRL ontology presented in section 4.

6.1 Semantic Queries Using Additional Ontologies

URM is designed for end users who want to manage the usage rights of their digital media files. Users often further annotate such files in order to easily recover them at a later date. For example, audio files containing music can be annotated with the artists' names, the title of the musical work, its genre, etc.

[11] Note that multi-inheritance is possible in both RDFS and OWL.

However, most of the metadata formats used for annotating media files are only used within particular applications or are only applicable for specific file formats. For example, ID3 tags are mostly used for MP3 files.

In order to provide an application-independent metadata vocabulary that can be used for annotating many different kinds of media file formats, a corresponding ontology can be used instead. Combining such a domain-specific ontology with the URM ontology allows complex semantic queries that cover both the metadata information and the legal aspects of the media file.

The Music Ontology [11] is an example for a domain-specific ontology for describing musical information. By combining this ontology with the URM ontology, queries of the form "List all music tracks of the genre Jazz that were recorded between 1951 and 1968 and which I am allowed to give to a close friend without losing any usage rights of the track myself" become possible . The first part of this query can be answered according to the annotations made with the Music Ontology, and the second part of the query relates to the URM licenses. In a similar way, other domain-specific ontologies can also be combined with the URM ontology.

6.2 Combining the URM Ontology with a Semantic Desktop

Similar to the above use case, the URM ontology can be integrated with a Social Semantic Desktop (SSD). The concept of the SSD aims at sharing information among several users, who form a collaboration network. The information is represented by files and file annotations which makes the sharing of these a core concept of the SSD. However, none of the existing implementations of a SSD currently consider the legal aspects of a sharing process. Users can easily share particular information with other users even if this sharing is explicitly forbidden. Therefore, a user may unintentionally behave illegally without actually realizing it.

Since the concept of the SSD is based on the semantic web, it makes heavy use of ontologies. An example for a framework of such ontologies is the NEPOMUK project [12]. By integrating the URM ontology with a SSD ontology, prohibited sharing of information can be detected beforehand. This allows protecting users from unintentionally behaving illegally.

7 Conclusion and Further Work

This paper presented the first steps toward an OWL ontology for the ODRL vocabulary. The ontology was created by manually interpreting the ODRL specification in order to cover as much of the semantics as possible. Some aspects could not accordingly be mapped yet and must be reconsidered in future work. The aim was to provide a formal semantics based on web ontologies that could be used for automatically interpreting an ODRL license. Future work has especially to deal with this issue and has to consider the limitations of such an automatic interpretation. However, the main structure of the ontology is already in a usable state.

Future work also has to consider how to use additional ontologies with the ODRL ontology. Such ontologies could be used for describing currencies and geospatial information. The presented ODRL ontology itself currently covers both ODRL's expression language and the data dictionary. Since the models of the data dictionary are not mandatory for an ODRL license, they could be sourced out to separate ontologies.

Based on the ODRL ontology, a second one was introduced for URM. URM is a tool that provides information about the usage rights of digital media files. This paper presented several models for describing how a media file was obtained from its origin. Expressing this information in an ontology allows combining it with other ontology-based metadata frameworks. Two use cases were presented that connect the URM ontology to such frameworks. Future work will address how to actually integrate the URM ontology with a Social Semantic Desktop application and how to combine it with a domain-specific ontology.

References

1. Iannella, R.: Open Digital Rights Language (ODRL) Version 1.1. World Wide Web Consortium (2002)
2. Hundacker, H., Pähler, D., Grimm, R.: URM - Usage Rights Management. In: Nützel, J. & Arnap, A. (eds.) Virtual Goods 2009 (2009)
3. Pucella, R., Weissman, V.: A formal foundation for ODRL. Workshop on Issues in the Theory of Security (2004)
4. Holzer, M., Katzenbeisser, S., Schallhart, C.: Towards a formal semantics for ODRL. Proceedings of the 1st International workshop on ODRL, pp. 137-148, (2004)
5. García, R., Gil, R., Gallego, I., Delgado, J.: Formalising ODRL Semantics Using Web Ontologies. Open Digital Rights Language Workshop 2005 (2005)
6. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., Stein, L. A.: OWL Web Ontology Language Reference. W3C Recommendation (2004)
7. Delgado, J., Gallego, I., Llorente, S., Garcia, R.: IPROnto: An Ontology for Digital Rights Management. 16th Conference on Legal Knowledge and Information Systems, JURIX. Frontiers in Artificial Intelligence and Applications, IOS Press (2003)
8. Manola, F. & Miller, E. (eds.): RDF Primer. W3C Recommendation (2004)
9. Brickley, D. & Guha, R. V. (eds.): RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation (2004)
10. Eckstein, U.: Entwicklung eines RDF-Schemas für ODRL. University of Koblenz-Landau (2010)
11. Raimond, Y., Abdallah, S., Sandler, M., Giasson, F.: The Music Ontology. Proceedings of the International Conference on Music Information Retrieval, pp. 417-422 (2007)
12. Groza, T., Handschuh, S., Moeller, K., Grimnes, G., Sauermann, L., Minack, E., Mesnage, C., Jazayeri, M., Reif, G., Gudjonsdottir, R.: The NEPOMUK Project - On the Way to the Social Semantic Desktop. Proceedings of I-Semantics, pp. 201-211 (2007)