Issues in Expressing Metadata Application Profiles with Description Logics and OWL 2

Dimitrios A. Koutsomitropoulos   Georgia D. Solomou

High Performance Information Systems Laboratory
Computer Engineering and Informatics Dpt.
University of Patras, Building B
26500, Patras-Rio, Greece
+302610996900

{kotsomit, solomou}@hpclab.ceid.upatras.gr

ABSTRACT

This paper gives an account of how traditional metadata application profiles are related to Web ontologies and Description Logics. It is shown that metadata profiles can be formalized into Description Logics; oddly enough though, OWL 2, with its current expressivity, is shown to be inadequate for this purpose. First, we give an overview of the recent proposal for representing metadata profiles using the notion of Description Set Profiles and XML Schema. We point out the necessity of an additional representation scheme, using also ontological languages. Following, we introduce a possible translation of Description Set Profiles to Description Logics and identify the required expressivity characteristics that are essential for such a translation. The relationship of this representation to OWL Integrity Constraints is also examined. Finally, we discuss what expense do these characteristics put on to the complexity of reasoning.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services – *web-based services.*

H.3.7 [**Information Storage and Retrieval**]: Digital Libraries – *standards.*

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods – *representation languages.*

General Terms

Performance, Design, Standardization, Languages, Theory.

Keywords

Semantic Web, Metadata, Interoperability, Ontologies, Integrity Constraints, XML.

1. INTRODUCTION

Metadata application profiles are a means to describe resources with a fine-grained analysis that a single metadata schema cannot provide. The most common way to express an application profile is to "mix & match" metadata elements coming from various schemata. Consider for example the following fragment:

```
<?xml version="1.0"?>

<record
xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:dspace="http://repository.upatras.gr/dspace/"
xmlns:lom="http://ltsc.ieee.org/xsd/LOM/"
>
 <dspace:Item ID="1987/143">
   <dcterms:title>Good Practices</dcterms:title>
   <dspace:author>David Dawson</dspace:author>
   <dcterms:language>en-US</dcterms:language>
   <lom:status>Revised</lom:status>
 </dspace:Item>
</record>
```

Figure 1. En example metadata record in XML format.

In the example above, encoded in XML format, we notice how elements provided by separate namespaces can be combined into a single metadata record. In fact, this example is a real metadata record from the University of Patras institutional repository (http://repository.upatras.gr/dspace/) that employs DSpace[1].

Recently attention has been drawn to the development of a standardized way to define, construct and represent application profiles [9]. This standardization is important because it attempts to offer a means to clearly represent the semantic as well as the syntactic constraints posed implicitly or explicitly by an application profile. On the other hand, the Semantic Web and its related languages (with OWL 2 as the most prominent) become gradually the *de facto* standard for managing and representing the semantics of metadata and the knowledge they inherently carry, especially in the e-learning and the cultural heritage domain [3].

The question then naturally arises as to whether and to what extent is it possible to use Semantic Web ontology languages as bearers of metadata application profiles. In particular, now that this paradigm is being standardized within the Dublin Core community, it would be worth investigating its correspondence to OWL and OWL 2. The path from application profiles to OWL passes through Description Logics (DLs), as the latter serve naturally as the logical foundation of Semantic Web languages.

[1] http://www.dspace.org

If such a correspondence is possible, then all the benefits of ontologies and the Semantic Web become readily available for application profiles as well. For example, automated reasoning, knowledge acquisition and semantic interoperability can seriously boost any metadata-intensive application, such as a digital library, which usually depends on one or more metadata application profiles for describing its resources.

Therefore, in this paper we see how metadata application profiles can be formalized into Description Logics and present a relevant procedure for this purpose. In this process we identify certain expressivity characteristics missing from OWL and OWL 2, which are necessary for expressing Description Set Profiles (DSPs) in these languages. The recent efforts for adding integrity constraints to OWL [6], in a sense similar to relational databases, can also be relevant to this formalization, since DSPs mainly deal with constraints. Consequently, we investigate their potential applicability in this context and see how they compare to our DSP-to-DL translation.

Section 2 presents the background behind the notion of DSPs and gives particular attention to its motivation. Then, section 3 describes the mechanics of DSPs and presents a concrete example. The formalization procedure from DSPs to Description Logics is given in section 4, where the alternative of integrity constraints is also considered. OWL 2 is not adequate for such a formalization and section 5 discusses how the "missing" characteristics may harden the reasoning process. Finally, section 6 outlines the conclusions and implications of this work.

2. BACKGROUND

The need for a consistent framework for developing application profiles has also been recognized from within the Dublin Core Metadata Initiative (DCMI). After the DC 2007 conference, the so-called "Singapore Framework" for developing application profiles based on Dublin Core (DC) was introduced in [8]. Building upon notions of the DCAM [12], a major part of the Singapore Framework is the development of a constraint language, just like an XML schema, that would formally specify what kind of (possibly originating from different standards) properties the particular application profile involves and what kind of values are appropriate for these properties, possibly constrained by specific syntax and vocabulary encoding schemes.

The purpose of such a *Description Set Profile* (DSP) [7] is to identify metadata records that are matching (or conforming) to this DSP. This in turn means that the DSP language may be primarily used for expressing *structural* and *syntactic* constraints that underline the application profile, leaving out of scope *semantic* interoperability. Besides, as mentioned in the Singapore Framework:

"It is important to realize that the semantics of the terms used in application profiles is carried by their definitions, which are independent of any application profile. [...]. As semantic interoperability is provided by a correct use of terms defined in one or more vocabularies, application profiles are about providing high-level syntactic or structural interoperability in addition to the semantic interoperability".

The above argument is not valid in the case of *semantic application profiles*, introduced in [3], where the notion of semantic profiling further refines the semantics of terms and drives towards semantic interoperability; in fact, application profiles are as much about structure as they are about semantics. Moreover, and despite the implementation of the DSP language in an XML Schema[2], we believe that its expression in RDF would be more appropriate, having in mind the recent implementations of DC in RDF(S) [10] as well as the DC ontology in OWL [4]. Finally, this approach can have a desirable effect in Semantic Web applications, by allowing them to draw reasoning-based conclusions about metadata, rather than strictly enforcing a set of syntactic constraints only.

3. DESCRIPTION SET PROFILES

The main stated purpose of a DSP is to define the structure of metadata records, adhering to a specific DC application profile [9]. A metadata record describes a resource by defining a series of properties and the values the resource is related with, through these properties. A DSP then formalizes exactly the restrictions that these properties and values should conform to.

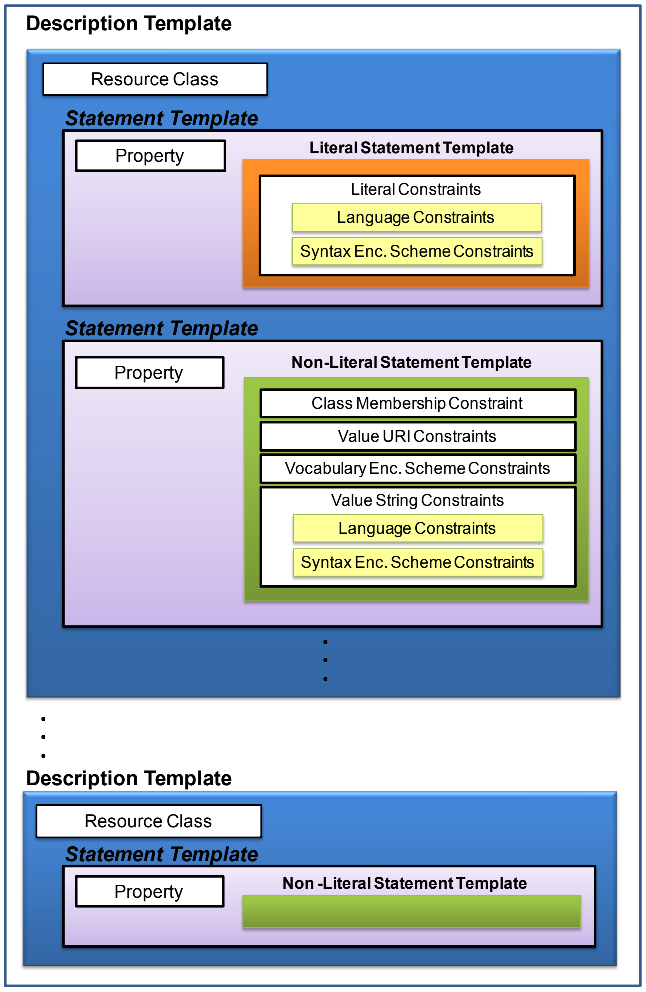

Figure 2. The structure of a Description Set Profile.

The basic structural element of a DSP is a *description template*. A DSP may include any number of such templates. A description

[2]http://dublincore.org/architecturewiki/DescriptionSetProfile?action=AttachFile&do=get&target=dcmi-dsp.xsd

```
<?xml version="1.0" ?>

<DescriptionSetTemplate xmlns="http://dublincore.org/xml/dc-dsp/2008/03/31">
 <DescriptionTemplate ID="item" standalone="yes">

   <ResourceClass>http://repository.upatras.gr/dspace/Item</ResourceClass>

   <StatementTemplate ID="title" minOccurs="1" type="literal">
    <Property>http://purl.org/dc/terms/title</Property>
   </StatementTemplate>

   <StatementTemplate ID="author" minOccurs="1" type="nonliteral">
    <Property> http://repository.upatras.gr/dspace/author</Property>
    <NonLiteralConstraint>
     <ValueClass>http://purl.org/dc/terms/Agent</ValueClass>
    </NonLiteralConstraint>
   </StatementTemplate>

   <StatementTemplate ID="language" minOccurs="0" maxOccurs="1" type="literal">
    <Property>http://purl.org/dc/terms/language</Property>
    <LiteralConstraint>
     <SyntaxEncodingSchemeOccurrence>mandatory</SyntaxEncodingSchemeOccurrence>
     <SyntaxEncodingScheme>http://www.w3.org/2001/XMLSchema#language</SyntaxEncodingScheme>
    </LiteralConstraint>
  </StatementTemplate>

  <StatementTemplate ID="role" minOccurs="0" maxOccurs="1" type="nonliteral">
   <Property>http://ltsc.ieee.org/xsd/LOM/status</Property>
   <NonLiteralConstraint>
    <ValueClass>http://ltsc.ieee.org/xsd/LOM/Status</ValueClass>
    <ValueURIOccurrence>mandatory</ValueURIOccurrence>
    <ValueURI>http://ltsc.ieee.org/xsd/LOM/Draft</ValueURI>
    <ValueURI>http://ltsc.ieee.org/xsd/LOM/Final</ValueURI>
    <ValueURI>http://ltsc.ieee.org/xsd/LOM/Revised</ValueURI>
   </NonLiteralConstraint>
  </StatementTemplate>

 </DescriptionTemplate>
</DescriptionSetTemplate>
```

Figure 3. An example DSP in XML syntax.

template corresponds to the description of resources of a specific type (e.g., items, persons, …), specified within a *Resource Class Constraint,* and defines restrictions on the set of properties that are relevant to the specific resource type (i.e., it is in their domain). Restrictions on a property are posed by a *statement template* and thus a description template can have any number of statement templates (see Figure 2).

Depending on whether the described property has a literal or a non-literal range, a *literal statement template* or a *non-literal statement template* can be optionally used inside a statement template, in order to express relevant constraints. For example, a common constraint imposed on non-literal values is expressed by a *Class Membership Constraint,* which requires all values to be members of a specific class. Moreover, a *Vocabulary Encoding Scheme Constraint* declares what controlled vocabularies the property values must originate from, if any. The *Literal Constraints* concern limitations on the values' language (expressed by *Language Constraints*) or their syntactic form. For the syntactic form, a certain syntax encoding scheme may be followed, indicated by a *Syntax Encoding Scheme Constraint*.

A metadata record in the University of Patras institutional repository adheres to a specific application profile that brings together metadata terms from various schemas. This application profile has its basis on the metadata schema of the DSpace system, where the repository is based upon. In particular, it involves some of the DCMI metadata terms (such as *title* and *subject*) and some DSpace-specific elements (such as *author* and *sponsorship*) and is further complemented by some educational metadata originating from the IEEE LOM specification (such as *status* and *interactivity type*). For DSpace-specific elements we have defined the namespace `http://repository.upatras.gr/dspace/`, which is assigned the `dspace:` prefix.

In order to formalize this application profile, a DSP can be used. For simplicity, in our example we refer to only a part of such a DSP, to which only a fragment of a metadata record should match. This fragment is assumed to be comprised of the mandatory fields `dcterms:title` and `dspace:author` and the optional fields `dcterms:language` and `lom:status`. `dcterms:title` and `dcterms:language` accept literal values; on the other hand the values for `dspace:author` and `lom:status` come from the `dcterms:Agent` and `lom:Status` classes respectively (i.e., they accept non-literal values). A conforming metadata record should look like the one in Figure 1.

A description template can then be used in order to specify the descriptions of items, containing a series of statement templates, one for each property characterizing the item. The XML representation of this template is given in Figure 3.

4. THE FORMALIZATION PROCEDURE

In this section we see how the constraints of a DSP can be formalized in OWL through a translation to Description Logics. In addition, we consider the notion of integrity constraints for OWL, and show how it can be related to this purpose.

4.1 Translation to DLs

First, notice that structural constraints, such as values permitted and typing of resources, have their counterpart in RDFS domain and range restrictions. Also the notion of "allowed properties" can be accommodated as in the following:

A description template can be seen as a single property on its own; a property that is partitioned by the set of allowed properties (i.e., an *n-ary* property)[3]. An approach to define such properties can be found in [11]. We can define a class *Description_ID* for each description template, thus actually reifying the *n*-ary relation. Then, for each allowed property $P_1, \ldots, P_n$:

$$Description\_ID \sqsubseteq \exists P_1.range_1 \sqcap \ldots \sqcap \exists P_n.range_n \quad (1)$$

This expresses the constraint that every instance of *Description_ID* has at least one relation, through $P_1, \ldots, P_n$ with an instance from the appropriate range. To express the constraint that $P_1, \ldots, P_n$ can relate *Description_ID* instances *only* to the appropriate ranges, we can use universal quantification:

$$Description\_ID \sqsubseteq \forall P_1.range_1 \sqcap \ldots \sqcap \forall P_n.range_n \quad (2)$$

Restrictions on the number of allowed fillers for each property can be modeled in the same manner, using qualified number restrictions, thus replacing the (more general) existential restrictions.

To see that the above expressions are also sufficient (not only necessary), suppose that $x \in Description\_ID$ and $P_k(x, y)$, where $1 \leq k \leq n$ and y does not belong to the allowed range. Then, due to (2) the ontology becomes inconsistent, since y must be an instance of the appropriate P_k range. Also, due to (1) *x must* have (although undeclared yet) n-1 other relations, through allowed properties.

To express the fact that $P_1, \ldots, P_n$ are the *only* properties allowed requires some more elaboration. In fact, what we need is the ability to express a role-disjunction axiom. OWL 2 provides for disjoint roles, but not for role disjunctions in general. Let U be the *universal role*, i.e. the parent of all roles. It holds that:

$$\exists P_1.range_1 \sqcup \ldots \sqcup \exists P_n.range_n \sqsubseteq \exists U.\top$$

We want to express that $\exists P_m.range_m$, where m other than $1 \ldots n$ (actually $P_m \sqcap P_i \equiv \emptyset$, for each $1 \leq i \leq n$), is not allowed in *Description_ID*. In the presence of a role-disjunction axiom and role complement a concise representation of the "non-allowed properties" set can be expressed as:

$$U \sqcap \neg(P_1 \sqcup \ldots \sqcup P_n)$$

i.e. the maximal property that is disjoint with every allowed property. *Description_ID* should not include instances that are related to others through non-allowed properties. That is:

$$Description\_ID \sqcap \exists(U \sqcap \neg(P_1 \sqcup \ldots \sqcup P_n)).\top \equiv \emptyset$$

Finally, a description template can be imposed over the resource class it is related to, by making this class a subclass of *Description_ID*.

For reasons of clarity, let's examine how the DSP example of section 3 can be expressed in OWL 2 following the above procedure. In OWL 2 *Manchester Syntax*[4] this would be expressed as in Table 1. Note however that this is now an incomplete translation due to the Description Logics expressivity characteristics missing in OWL 2.

Table 1. The example DSP in Manchester Syntax.

```
Class: Description_item
  SubClassOf: dcterms:title some string and
  dspace:author some dcterms:Agent and
  dcterms:language max 1 xsd:language and
  lom:status max 1 lom:Status
  SubClassOf: dcterms:title only string and
  dspace:author only dcterms:Agent and
  dcterms:language only xsd:language and
  lom:status only lom:Status
Class: lom:Status
  EquivalentTo: {lom:Draft, lom:Final,
                 lom:Revised}
Class: dspace:Item
    SubClassOf: Description_item
```

Notice that, unless we could express role disjunction and complement in OWL 2, nothing can stop us from stating that *item 1987/143* has also a property other than the allowed ones. E.g. with:

```
Individual: item_1987/143
  Types: dspace:Item, Description_item
  Facts: dcterms:type dspace:book
```

item 1987/143 can still be an instance of the Description_item class without problems.

Notice also that as it stands, the example above would not match the DSP for an additional reason: Not all properties are included in the *Facts* and the DSP requires that *all* its (mandatory) property statements need to be matched.

[3] Assuming that each statement template corresponds to a single property. In the case that *a list* of properties is allowed for a statement, this can be similarly modeled in a recursive fashion.

[4] http://www.w3.org/TR/owl2-manchester-syntax/

This highlights another difference between mere syntactic constraints and the OWL world: the *Open World Assumption* (OWA). In OWL, the example above would be perfectly satisfiable in accordance to the ontology statements in Table 1, even though the item has no stated facts about mandatory properties such as `dspace:author`. Rather, it would be *inferred* that these facts hold, i.e. it would be inferred for example that *item 1987/143* has an author that is a member of the Agent class, but we do not know yet exactly who.

Therefore, translating DSP constraints to OWL allows the application profile to be more *conclusive* - by allowing safe, metadata-based conclusions to be drawn - in contrast to *preclusive*, as it would be for example in an XML schema, and this behavior can be desirable in many applications where semantic interoperability plays a major role.

An opposite track can be followed by considering integrity constraints, where the OWA no longer holds, as discussed in the next section.

4.2 Relationship with Integrity Constraints

A subset of DSP constraints may be modeled using *Integrity Constraints* (ICs). Indeed, there is a bunch of approaches for augmenting OWL with ICs (e.g. see [15] for an overview). These efforts mainly focus on specifying the semantics that these ICs should have, often in accordance to the usual OWL constructs, and then on devising suitable algorithms for interpreting these semantics.

A most straightforward implementation of ICs for OWL can be found in the Pellet ICV tool[5]. There, an IC is expressed in the form of a normal OWL axiom, using for example an existential restriction, a subsumption relation, a property disjointness axiom and so on. These axioms however are treated differently than normal KB axioms: they are converted to a corresponding SPARQL query and, if this query succeeds in returning any results, then the IC is considered to be violated.

It is easy to see that this approach can work for some of the cases described above, by interpreting our subsumption axioms as ICs (complementary to their standard OWL semantics). For example, (1) can generate a series of SPARQL queries, one for each allowed property P_k, like in the following (using SPARQL 1.1):

```
ASK WHERE {
    ?x rdf:type Description_ID .
    NOT EXISTS {?x Pk ?y .
                   ?y rdf:type rangek .}}
```

If at least one of these queries succeeds, we can deduce that the corresponding constraint is violated. The difference now is that, in fact, a *Closed World Assumption* is enforced [14], and (1) is virtually interpreted as a check rather than an inference rule.

However, a possible IC for the “non-allowed properties” rule would still require the presence of a role disjunction axiom, as there appears to be no other way to bound the set of non-allowed properties, which are, in principle, unknown beforehand. If this was not the case, then a corresponding set of SPARQL queries could be easily constructed (one query for each non-allowed property) and validated against the instance set (ABox) of the ontology, thus successfully enforcing the required constraint.

[5] http://clarkparsia.com/pellet/icv/

5. EXPRESSIVITY CHARACTERISTICS AND COMPLEXITY

We therefore come to the conclusion that, in order to express DSPs it is necessary to have the three logical operators (union, intersection, complement) available for role expressions or, at least, role names only. Of the above operators, no one is available in OWL 2 (or OWL, for that matter). Additionally, we also need qualified number restrictions, which are not included in OWL, while the universal role can be safely eliminated [2].

Lutz and Sattler [5] show that the Description Logic *ALC* ($\sqcup$, $\sqcap$, $\neg$) that is, the basic logic *ALC* augmented with logical operators on roles, is in NEXP. Moreover, in [16], it is shown that the Description Logic *ALCIQ* ($\sqcup$, $\sqcap$, $\neg$) that is, the logic mentioned previously augmented with qualified number restrictions and role inverses, which apparently is adequate for expressing DSPs, but does not correspond to any ontology language, is NEXP-complete.

Under the condition that role expressions, transformed to *Disjunctive Normal Form* (DNF, i.e., union of intersections), should have at least one non-negated role in each disjunct (aka *safe Boolean combinations*), this logic becomes EXP-complete. For example, the expression $\neg(P_1 \sqcup \ldots \sqcup P_n)$ is written in DNF as $\neg P_1 \sqcap \ldots \sqcap \neg P_n$, which consists of only one disjunct that includes no non-negative parts, unless the universal role is used.

In addition, Schmidt and Tishkovsky in [13] show that the Description Logic *ALBO* that is, the basic logic *ALC* augmented with role union, role complement, inverse roles and nominals, is NEXP-complete (role intersection is not mentioned but included, because $P \sqcap R \equiv \neg (\neg P \sqcup \neg R)$, where P, R role names) and implement a corresponding tableau algorithm. They also mention that applying complement on role chains leads to undecidability, a fact that also holds when applying intersection [1].

Consequently, we argue that the RDF(S) and OWL semantics are not adequate for expressing the structural and syntactic limitations of DC application profiles. In addition, we see that, although DSPs intend to tackle with syntactic constraints only, it turns out that these have considerable semantic implications.

6. CONCLUSIONS

Counter-intuitively, the expression of classic metadata application profiles, as they are recently attempted to be standardized around DC with the Singapore Framework, cannot be achieved with the current expressivity of OWL 2. We have shown exactly what expressivity characteristics are missing from the last specification of this language and are necessary for the formalization of such profiles. Nevertheless, we have shown that the syntactic constraints posed by a typical application profile can be represented as semantic restrictions inside an ontology. Also, this can mean a shift towards more “intelligent” manipulation of application profiles by applications.

The hardness -even undecidability- of reasoning under the addition of the missing characteristics should not be seen as weakness though: First, XML Schema can easily be used in order to enforce and verify such syntactic restrictions, at the cost however of confining metadata to the XML realm. Alternatively, extending OWL 2 with constraints or adopting a Closed World Assumption for reasoning may also be used in certain cases; on the other hand, the counterpart of these restrictions in Description Logics is not necessary to simultaneously include all the rest of the OWL 2 expressivity characteristics, except for only a few.

7. REFERENCES

[1] Calvanese, D. and Giacomo, G. d. 2007. Expressive Description Logics. In Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. F. (Eds.), *The Description Logic Handbook* (2nd ed.). Cambridge University Press.

[2] Horrocks, I., Kutz, O., and Sattler, U. 2006. The Even More Irresistible SROIQ. In *Proceedings of the 10th International Conference on Principles of Knowledge Representation and Reasoning* (Lake District, UK, June 02-05, 2006). AAAI Press, 57-67.

[3] Koutsomitropoulos, D., Paloukis, G. , and Papatheodorou, T. S. 2009. Semantic Application Profiles: A Means to Enhance Knowledge Discovery in Domain Metadata Models. In Sicilia, M. A., and Lytras, M. (Eds.), Metadata and Semantics, Springer, 23-34.

[4] Koutsomitropoulos, D. A., Solomou, G. D., and Papatheodorou, T. S. 2008. Semantic Interoperability of Dublin Core Metadata in Digital Repositories. In *Proceedings of the 5th International Conference on Innovations in Information Technology* (Al Ain, UAE, December 16-18, 2008). IEEE, 233-237.

[5] Lutz, C., and Sattler, U. 2000. Mary likes all Cats. In *Proceedings of the 2000 International Workshop in Description Logics* (RWTH Aachen, Germany, August 17-19, 2000). Vol 33 in CEUR-WS, 213-226.

[6] Motik, B., Horrocks, I., and Sattler, U. 2009. Bridging the gap between OWL and relational databases. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*. 7(2) (April 2009), 74-89.

[7] Nilsson, M. 2008. Description Set Profiles: A Constraint Language for Dublin Core Application Profiles. DCMI Working Draft. http://dublincore.org/documents/dc-dsp/

[8] Nilsson, M., Baker, T., and Johnston, P. 2008. The Singapore Framework for Dublin Core Application Profiles. http://www.dublincore.org/documents/singapore-framework/

[9] Nilsson, M., Miles, A., Johnston, P., and Enoksson, F. 2007. Formalizing Dublin Core Application Profiles - Description Set Profiles and Graph Constraints. Sicilia, M. A., and Lytras M. (Eds.), Metadata and Semantics, Springer, 101-112.

[10] Nilsson, M., Powell, A., Johnston, P., and Naeve, A. 2008. Expressing Dublin Core metadata using the Resource Description Framework (RDF). DCMI Recommendation. http://dublincore.org/documents/dc-rdf/

[11] Noy, N., and Rector, A. (Eds.). 2006. Defining N-ary Relations on the Semantic Web. Semantic Web Best Practices and Deployment Working Group Note. http://www.w3.org/TR/swbp-n-aryRelations/

[12] Powell, A., Nilsson, M., Naeve, A., Johnston, P., and Baker, T. 2007. DCMI Abstract Model. DCMI Recommendation. http://dublincore.org/documents/abstract-model/

[13] Schmidt, R. A., and Tishkovsky, D. 2007. Using Tableau to Decide Expressive Description Logics with Role Negation. In *Proceedings of the 6th International Semantic Web Conference* (Busan, Korea, November 11-15, 2007). Springer, 438-451.

[14] Sirin, E., Smith, M., and Wallace, E. 2008. Opening, Closing Worlds - On Integrity Constraints. In *Proceedings of the 5th OWL Experiences and Directions Workshop* (Karlsruhe, Germany, October 26-27, 2008). Vol 432 in CEUR-WS.

[15] Tao, J., Sirin, E., Bao, J., and McGuinness, D. L. 2010. Extending OWL with Integrity Constraints. In *Proceedings of the 23rd International Workshop on Description Logics* (Waterloo, Canada, May 04-07, 2010). Vol 573 in CEUR-WS.

[16] Tobies, S. 2001. Complexity results and practical algorithms for logics in Knowledge Representation. Doctoral Thesis. LuFG Theoretical Computer Science, RWTH-Aachen.