KUNGL
TEKNISKA
HÖGSKOLAN

CID-243 • ISSN 1403-0721 • Department of Numerical Analysis and Computer Science • KTH

The LOM RDF binding - principles and implementation

Nilsson, M., Palmér, M., Brase, J.
**Proceedings of the 3rd annual ARIADNE conference,
Katholieke Universiteit Leuven, Belgium**

CID, CENTRE FOR USER ORIENTED IT DESIGN

Nilsson, M., Palmér, M., Brase, J.

The LOM RDF binding - principles and implementation
Proceedings of the 3rd annual ARIADNE conference, Katholieke Universiteit Leuven, Belgium
Report number: CID-243
ISSN number: ISSN 1403 - 0721 (print) 1403 - 073 X (Web/PDF)
Publication date: November 2003
E-mail of author: mini@nada.kth.se, matthias@nada.kth.se

Reports can be ordered from:

CID, Centre for User Oriented IT Design
NADA, Deptartment of Numerical Analysis and Computer Science
KTH (Royal Institute of Technology)
SE- 100 44 Stockhom, Sweden
Telephone: + 46 (0)8 790 91 00
Fax: + 46 (0)8 790 90 99
E-mail: cid@nada.kth.se
URL: http://cid.nada.kth.se

The LOM RDF binding - principles and implementation

Mikael Nilsson[†] Matthias Palmér[†] Jan Brase[‡]

[†] Knowledge Management Research Group,
Centre for User Oriented IT design, KTH, Stockholm, Sweden
http://kmr.nada.kth.se

[‡] Information System Institute, University of Hannover, Germany
http://www.kbs.uni-hannover.de

Abstract

This paper will discuss some of the advantages and complexities in using the Resource Description Framework, RDF, to express learning object metadata following the IEEE LOM standard. We will describe some details of the current draft for a complete RDF binding for LOM and discuss some of the constructs used in that binding.

We will then present a so-called SHAME Query Model of this binding that can be used to specify and visualize application profile constraints when using this binding. A metadata editor for RDF-based LOM metadata, which was built with the help of this Query model, will be briefly introduced.

1 Introduction

In June 2002, IEEE approved the first version of the Learning Object Metadata (LOM) standard. LOM is gradually becoming the reference standard for educational systems managing learning objects of many kinds.

The LOM datamodel standard, or IEEE LTSC 1484.12.1, is only the first part of a multi-part standard. This first part contains an abstract model of the descriptors, or *elements* that are used to describe learning objects, and does not deal with the technical realisation of these elements.

The LOM elements will be managed in many different formats, including SQL tables, text files, HTML meta tags, and so on. Such a technical realization of the abstract model in a specific format is called a "binding". Work is currently underway within the Learning Technology Standards Committee (LTSC) of the IEEE to produce standards for two bindings of the LOM abstract data model[1]:

- XML, eXtensible Markup Language (P1484.12.3), and
- RDF, Resource Description Framework (P1484.12.4)

RDF is a metadata framework being developed by the W3C with the purpose of being used to annotate resources referenced by URIs in the context of the World Wide Web. The RDF specifications provide a simple and lightweight, yet sophisticated framework for exchanging ontology-based knowledge, containing facilities for combining resource descriptions using different vocabularies and from different sources. RDF can be seen as a metadata grammar, where terms from standards and community/application-specific vocabularies can coexist.

This paper documents parts of the effort to produce an RDF binding of LOM. This work was initiated in 2000 within the context of the IMS Global Learning Consortium [7], and a first draft was released as an appendix to version 1.2 of their popular metadata standard [9, 8], which was based on earlier drafts of the LOM standard. The effort was subsequently transferred to LTSC, and the current draft which is being prepared for ballot can be found at [13].

In this paper we will describe some of the challenges and problems encountered in the process of producing an RDF binding for LOM. We will briefly discuss the metamodels of the XML and RDF frameworks, modeling semantics, extensions, and then try to describe some of the main features of the LOM RDF binding. The paper will close with a presentation of a LOM RDF editor that has been constructed to match this binding.

2 Using RDF for Metadata as Compared to XML

There are significant differences in the metadata modeling approaches used in the LOM XML binding (currently in ballot) and in the RDF binding, resulting from both the differences in the design of the respective frameworks and their different typical usage scenarios. In this section, we will discuss these differences in some detail.

[1] a third binding to ISO/IEC 11404, Language Independent Datatypes (P1484.12.2) has recently been abandoned

2.1 Metadata metamodels

In order to understand the intricacies of binding an abstract metadata model to a specific technical expression, it is necessary to understand the concept of metadata "metamodels". These are the conceptual schemas we use to describe our metadata models such as LOM.

For example, one of the most common metadata schemas on the web today is the Dublin Core Schema (DC) by the DCMI. The "simple" version of the schema consists of a set of 15 independent elements, including for example: Title, Identifier, Language, Description (see [5]). "Qualified" Dublin Core employs additional qualifiers to further refine the description of a resource.

The *metamodel* for Dublin Core defines the semantics of the DC elements and their qualifiers, such as: "An element is a property of the resource being descibed", "An element refinement is a property of a resource that shares the meaning of a particular DCMI element but with narrower semantics", "An encoding scheme provides contextual information or parsing rules that aid in the interpretation of a value string". There is work underway to make the DC metamodel explicit. See [6]. It should be noted that the Dublin Core metamodel is deliberately designed to be compatible with RDF.

LOM, by contrast, uses a completely different metamodel. LOM describes resources using a set of more than 70 attributes, divided into these nine categories:

1. General
2. Lifecycle
3. Meta-Metadata
4. Technical
5. Educational
6. Rights
7. Relation
8. Annotation
9. Classification

The descriptors are organized in a tree-like structure under these categories. This tree makes it possible to organize the information in a consistent way, grouping information into related pieces. The LOM metamodel is thus based on a recursive container model.

However, it can be easily seen that this metamodel is not compatible with the DC metamodel. As a simple example, the **2.3.3 Date** element is not a property of the resource being described, but can be seen to be a property of the "Contribution" it belongs to. Similarly, the elements in the "Meta-metadata" category are not properties of the resource being described, but of the metadata document itself.

The container-based metamodel used by LOM is thus not compatible with the metamodel used by Dublin Core. When does this matter? Binding LOM to RDF is the obvious example in this context, as the metamodel of RDF is based on a property-value model and not containment. In general, it leads to difficulties when trying to combine terms from two metadata standards into the same system. When the metamodels are compatible, such a combination or mapping can be realized by simply translating the metamodel contructs. If the metamodels are incompatible, the translation must be done on an idiosyncratic, element-by-element basis.

This metamodel incompatibility is the main source of the challenges in binding LOM to RDF, as described in this paper.

2.2 Semantic modeling

In an XML binding such as the LOM XML binding, the structure of the XML instance is the result of choosing the most convenient syntax, creating the element hierarchy that best matches the structure of the LOM data model. The XML metamodel is also containment-based, and is therefore easily adapted to LOM.

Where XML data has no no other semantics than just a tree, RDF data has the semantics of an object-oriented system, and can therefore be viewed as objects having properties that relate them to other objects. The *type* of an object or of a property defines its interpretation, and is thus not simply a syntactic marker.

In the XML binding of LOM each LOM element is represented by an XML element. In RDF, the semantics of each LOM element decides its representation. If it is a property applying to a resource, use an RDF property. If it is a resource having certain properties, use an object with a specific type. If it is just a container with no object or property semantics (Such as "General"), one might consider using a namespace for the contained properties and object types. And the choice matters, as those constructs have fundamentally different semantics, i.e., they will be processed differently by applications. All of these constructs are used in the current binding draft.

Thus, a considerable amount of effort is needed to extract the desired semantic quality of each LOM element in order to be able to represent it appropriately. If this reinterpretation is not done, you risk losing not only clarity for the human consumer, but you risk more serious damage to the usefulness of the model. Much of the effort that has gone into the LOM RDF binding has focused on creating such a well-formed (i.e., machine-interpretable) semantics of the model.

We therefore expect to see much richer structures on many levels in an RDF representation than in the corresponding XML binding instance. The RDF binding thus adds semantics to the LOM data model, in that it adds interpretations to the elements that are not explicit in the LOM data model.

2.3 Metadata Frameworks: Documents vs. statements

The fundamental unit in RDF is the *statement*, that expresses the value of one property of one resource. Such statements can be arbitrarily combined, separated and recombined.

Thus, the meta-data for one resource need not be contained in a single RDF document. Translations might be administrated separately, and different categories of meta-data might be separated. This dramatically strengthens the incentive both to reuse identical structures that are used repeatedly, as well as to create decentralized descriptions of resources. Both of these phenomena naturally lead to a fundamentally different approach to meta-data modelling than that found in

XML-based metadata. While XML describes the structure of a complete metadata instance, RDF describes the structure of single metadata statement. The RDF binding must therefore be designed one element at a time.

As a consequence of this, we cannot expect the RDF binding to fulfill the same purpose as the XML binding. The XML binding defines an exchange format for meta-data. The meta-data might be contained in a database and an XML representation generated on demand, for export to other tools and environments. Thus, an XML meta-data record is a self-contained entity with a well-defined structure. In RDF, the metadata for a resource is not always self-contained, but rather forms part of a global network of information, where anyone has the capability of adding any kind of meta-data to any resource. This is further elaborated in [18].

2.4 Semantic and Structural Extensions

Another aspect is that of compatibility. In the XML binding of LOM, there is no standard way to reuse other meta-data standards. The reason for this is the monolithical nature of an XML document – there is no canonical way of combining information from two documents into one.

The statement-centric design of RDF leads to naturally reuseable constructs. Metadata elements can be extended both structurally (by adding more information), and semantically (by adding refinements of elements). This binding has been designed to be directly compatible with Dublin Core (including the DC Qualifiers, DC Type and DC Education vocabularies) and with the vCard RDF binding [14]. However, this compatibility comes at the price of modeling freedom – some modeling restrictions are imposed on us. Fortunately, much of this compatibility comes for free when using the RDF metamodel.

Finally, as RDF is intended to be processed by software, and in many cases software with no explicit knowledge of LOM, it is important to use explicit data typing, i.e. self-describing data. This will be seen below in the representation of languages and dates, which are strings tagged with their encoding scheme. Thus, a goal of this binding has been to define a set of RDF constructs that facilitates introduction of LOM meta-data into the semantic web in the most semantically complete and useful way.

For more information on the importance of extensions and metadata frameworks, see [12] and [17].

3 The RDF binding of LOM

We will now turn to discussing some of the main features of the LOM RDF binding. There is no need to explain in detail the binding of each and every LOM element, as that is covered by the binding draft. However, there are a number of modeling constructs that are of more general interest, and we will now discuss them. For details on these constructs and the rest of the binding, we refer to the binding itself ([13])

3.1 Using RDF schemas

The binding makes use of RDF schema to express some of the semantics of the RDF constructs. In contrast to XML schemas, which are used for validation of XML records, RDF schemas are used to define the semantics of RDF classes and properties. This includes specifying the value range of properties, their relationship to other properties (such as being a refinement in the Dublin Core sense), and so on.

Therefore, RDF schemas do not support the level of validation offered by XML schemas, but are used as support in (machine-) interpretation of the instance data.

3.2 Relationship to Dublin Core

Some of the LOM elements are semantically similar to Dublin Core elements, and Appendix B of the LOM standard contains a translation between these elements and the corresponding Dublin Core elements.

Our RDF representation of LOM relies heavily on the Dublin Core meta-data element set ([5]), and its representation in RDF. LOM elements are modeled in a way similar to the representation of Dublin Core Qualifiers, give in [4] in RDF. Where applicable, LOM elements are described as *rdfs:subClassOf* or *rdfs:subPropertyOf* the corresponding DC/Qualified DC elements. In this sense, parts of LOM can be viewed as a proper extensions to qualified Dublin Core.

Our RDF representation of LOM is therefore almost fully Dublin Core RDF compatible, in the sense that most Dublin Core meta-data constructed according to this binding can be directly understood by Dublin Core-aware software. Most of the elements of the LOM Dublin Core mapping (in Appendix B of [11]) are compatibly represented, allowing the use of all the Dublin Core constructs in a way compatible with both [4] and this binding. It is, however, not always possible to map a pure Dublin Core construct (constructed without reference to this binding) to a LOM element without adding information, as LOM requires a more specific structure in many elements. The guiding principle has instead been that using the *dumb-down* algorithm described in [4] on LOM metadata should result in useful Dublin Core metadata. It should be noted that this results in a metadata structure that closely conforms to the requirements of the IEEE-DCMI MoU [10].

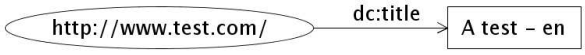
3.3 Langstring

In the LOM standard, many of the entries are either of the Datatype *Langstring* or *Vocabulary*. The first one can be easily realized in RDF. When encoding a string in a specific language, we use the language tag for RDF literal. In the XML serialization, this corresponds to

the xml:lang attribute, as described in [15] and [4]. An example of a language-tagged string follows:

```
<rdf:Description rdf:about="http://www.test.com/">
  <dc:title xml:lang="en">A test</dc:title>
</rdf:Description>
```

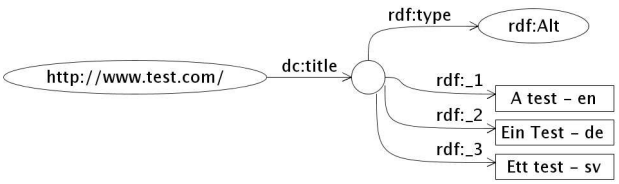
Viewed as a graph, it will look like



Here "en" is a language code conforming to RCF1766 (see [16]). In order to encode strings in several languages, which is needed for the LangString construct, we use the rdf:Alt construct:

```
<rdf:Description rdf:about="http://www.test.com/">
  <dc:title>
   <rdf:Alt>
    <rdf:li xml:lang="en">A test</rdf:li>
    <rdf:li xml:lang="de">Ein Test</rdf:li>
    <rdf:li xml:lang="se">En test</rdf:li>
   </rdf:Alt>
  </dc:title>
</rdf:Description>
```

which will look like

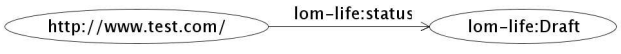


This technique allows us to separate the original title from translations, as the first title is the default (according to the semantics of rdf:Alt). It also allows Dublin Core-only RDF parsers to understand what the title is, via the "dumb-down" algorithm. Finally, it allows us to add translations in separate RDF documents. A necessary prerequisite for this is that rdf:Alt instances are given a URI so that it can be referenced.

3.4 Vocabularies

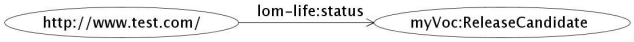
Vocabularies are represented in several different ways in this binding. The fundamental idea is that the (source, value) construct in LOM is best represented in RDF using the (namespace, value) construct that is naturally contained in a resource URI in RDF. Thus, vocabulary values are resources, and the source of a vocabulary is implicit in the URI of a resource.

This binding provides RDF resources for all the restricted vocabulary terms defined in LOM. These resources can be used directly as values of the corresponding property, for example:



These resources are in turn described in the LOM RDF schemas, which give them their official label and description. In the case of the "Draft" term, it is described as being of type "Status", as are all the terms in the LOM "status" vocabulary.

Users of the binding are free to define their own RDF resources for use as values in vocabularies, for example:



In the RDF schema describing this vocabulary, the "ReleaseCandidate" resource would also be described as an instance of the "Status" class. In this way, extending the LOM vocabularies is as simple as defining new instances of the relevant RDF schema classes.

Thus, vocabularies will need to be explicitly translated to RDF. This convention leads to some difficulties when interfacing with the XML binding, where vocabularies are not explicitly defined in this way. Further development in this area will be necessary.

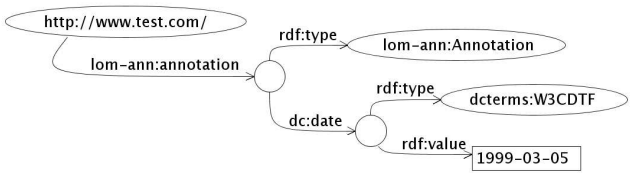
3.5 Using vocabularies for Properties

In several cases, the LOM vocabulary item is not to be used as the object of an RDF Statement, but rather as the predicate in the statement. This is the case with element *7.1 Relation.Kind*. An example could look like:

Here the Relation is of Relation.Kind "hasPart". LOM defines twelve terms for this vocabulary, and each of them corresponds to a separate property. Defining new vocabularies for this element is as simple as for the "Status" example above. The only difference is that in this case, instead of defining new instances of the "Status" class, one would need to define new sub-properties of the property *dc:relation*. This new property is an RDF resource, and thus the same remarks apply: explicit translation of vocabularies to RDF is necessary, the terms can be described in an RDF schema, and care must be taken when interfacing with the XML binding.

3.6 Element encodings

There are many places in the LOM standard where string literals that are not intended to be human-language text are used as values, such as dates, whole numbers or ranges of numbers, or language tags. When encoding such values, the LOM RDF binding takes the approach of tagging the value with a data type. Describing the date of an annotation (with no actual annotation text), for example, looks like:



This construct is used to indicate that the string "1999-03-05" is encoded using the W3C Date and Time format. Dublin Core defines several useful such element encodings such as W3CDTF for dates and RFC1766 for language tags. In some other cases, the LOM RDF binding defines new datatypes for similar fields. Using this technique, the RDF data becomes self-describing in a very useful way.

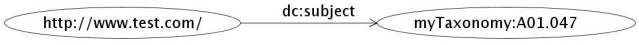
3.7 Using Metametadata

Generally, the metametadata category is obsoleted in RDF, as RDF itself comes with good support for metametadata. Two ways to describe such information are provided by RDF, and both rely on reusing the usual metadata properties from LOM and Dublin Core. These properties are applied to either:

- the URI representing the RDF document containing the metadata
- a set of RDF statements (using the RDF reification mechanism)

3.8 Classifications

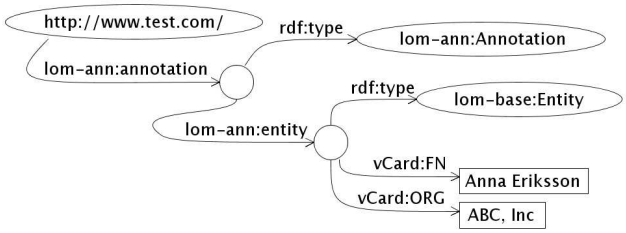
This is the most complex category of all in LOM. Instead of describing the full path to the describing "taxon" element in each metadata instance, the RDF binding allows taxonomies to be described separately from each metadata instance. The idea is to represent a hierarchical taxonomy separately, and then point into nodes in this hierarchy when classifying resources. At the same time, it is possible to reuse the subject classifications from Dublin Core Qualifiers. Using this will then look like:



In this example, the value is an element in a subject classification. This "taxon" can be described in a separate RDF document, and annotated using ordinary RDF metadata. For detailed information on the use of Taxonomies for the Classifications category, we refer to [1].

3.9 VCards

Another common LOM value type is the VCard, which is used in several places to describe a person or other entity. In the XML binding, VCards are inserted literally, without XML markup. In the RDF binding, the VCard is made into a resource, with properties such as *vCard:FN*, *vCard:ORG* being used to describe the VCard properties of that entity. The RDF properties are taken from the VCard RDF binding ([14]). Describing the entity that made an annotation in LOM (with no actual annotation text) could for example look like



4 The LOM RDF Query model

So far we have discussed how the LOM RDF binding is expressed in RDF using plain english. The RDF schemas is a technical description of some of the constraints for the LOM properties and classes that can be applied to any usage of the LOM elements.

In the next step we want a machine-processabel description of a metadata record using LOM, aggregating a number of LOM properties and choosing a set of vocabularies to use, etc. For this we need to have a more strict description of the details of a metadata record. We have used an approach from the SHAME project [19] called Query Models to specify metadata records using LOM RDF constructs.

As mentioned above, the RDF Schemas are used to define classes, properties, and whenever possible, range and domain restrictions on some of these properties. While the schema restrains the use of the properties on a global level, the Query Model specifies their usage in the specific context of a complete metadata record, when seen as an application profile containing a certain set of properties, vocabularies, etc. Thus, the SHAME Query Model is context dependent, and is used here to bridge the gap between the closed, record-oriented LOM datamodel and the open, RDF Schema based, statement-oriented model.

A Query Model can therefore be seen a formal description of an RDF metadata record, and can be visualized as a tree, rooted in the resource being described. Each arc corresponds to an RDF property, and each node corresponds to an RDF resource[2]. This tree is a generic mirror, or template, of how a full metadata record would be constructed and hence is also very suitable as a visualization of the metadata profile.

We designed a metadata record that includes exactly the set of properties contained in LOM, using the standard vocabularies given by LOM. The part of the Query Model that describes the Technical category of LOM can

[2]This is expressed using RDF *reifications*

be seen in Figure 1). The LOM Query Model was constructed with the help of the general purpose RDF editing and visualization tool Conzilla [2]. For more information about Query models we refer to [19].

5 A LOM editor

The LOM Query Model also allows us to construct tools for validation, querying, presentation or editing of LOM records. Based on the LOM RDF Query model, we can create various LOM editors via a complementary Form Model (which specifies how the Query Model should be presented in a form). The full LOM editor in Figure 2 makes use of the entire LOM Query Model. Provided with a URL for vocabularies, we can present the different choices in a drop down menu, as for the MIME types in 4.1 Format in our example. Alternative editors using e.g. a subset of LOM can be created from the same Query Model. Additionally, the vocabularies used in LOM can easily be extended or changed by including them in the Form Model.

6 Conclusion

There are many challenges in designing a LOM RDF binding, stemming from both incompatibilities in the models used to describe data in LOM and RDF, as well as incomplete semantics in parts of LOM. However, it has been shown that these difficulties are mostly surmountable, and that the LOM RDF binding can be successfully used to annotate resources.

In addition, application of the SHAME metadata editing framework has shown that the flexibility (in terms of being able to mix and match metadata standards) and extensibility (in terms of being able to plug in new vocabularies and refine existing) promised by the use of RDF, can be realized in a system using the LOM RDF binding.

Bibliography

References

[1] Brase J. & Nejdl W.
"Ontologies and Metadata for eLearning"
in *Handbook on Ontologies* (Springer-Verlag 2003)

[2] Nilsson M. & Palmér M.
Conzilla - Towards a Concept Browser,
(CID-53), KTH, 1999

[3] The Dublin Core Metadata Initiative
http://dublincore.org/

[4] Expressing Qualified Dublin Core in RDF / XML (Proposed Recommendation)
http://dublincore.org/documents/2002/04/14/dcq-rdf-xml/

[5] Dublin Core Metadata Element Set, Version 1.1: Reference Description
http://dublincore.org/documents/1999/07/02/dces/

[6] Dublin Core Abstract Model, Working Draft
http://www.ukoln.ac.uk/metadata/dcmi/abstract-model/

[7] IMS Global Learning Consortium
http://www.imsproject.org

[8] IMS Resource Description Framework(RDF) Bindings
http://www.imsproject.org/rdf/

[9] IMS Learning Resource Meta-data Specification
http://www.imsproject.org/metadata/index.cfm

[10] Memorandum of Understanding between the Dublin Core Metadata Initiative and the IEEE Learning Technology Standards Committee
http://dublincore.org/documents/2000/12/06/dcmi-ieee-mou/

[11] Learning Technology Standards Comittee of the IEEE: *Draft Standard for Learning Objects Metadata IEEE P1484.12.1/D6.4*12. June 2002).
http://ltsc.ieee.org/

[12] Duval, E., Hodgins, W., Sutton, S., Weibel, S. L., *Metadata Principles and Practicalities*, D-Lib Magazine, April 2002,
http://www.dlib.org/dlib/april02/weibel/04weibel.html

[13] The RDF binding of LOM
http://kmr.nada.kth.se/el/ims/metadata.html

[14] The vCard RDF binding http://www.w3.org/2001/vcard-rdf/3.0

[15] Refactoring RDF/XML Syntax (Working Draft)
http://www.w3.org/TR/rdf-syntax-grammar/

[16] Tags for the Identification of Languages
http://www.ietf.org/rfc/rfc1766.txt

[17] Nilsson, M., *The semantic web: How RDF will change learning technology standards*, CETIS Feature article, Sept 27, 2001.
http://www.cetis.ac.uk/content/20010927172953/viewArticle

[18] Nilsson, M., Palmér, M., Naeve, A., *Semantic Web Meta-data for e-Learning - Some Architectural Guidelines*, Proceedings of the 11th World Wide Web Conference (WWW2002), Hawaii, USA.
http://kmr.nada.kth.se/papers/SemanticWeb/p744-nilsson.pdf

[19] *SHAME - Standardized Hyper Adaptable Metadata Editor*,
http://kmr.nada.kth.se/shame

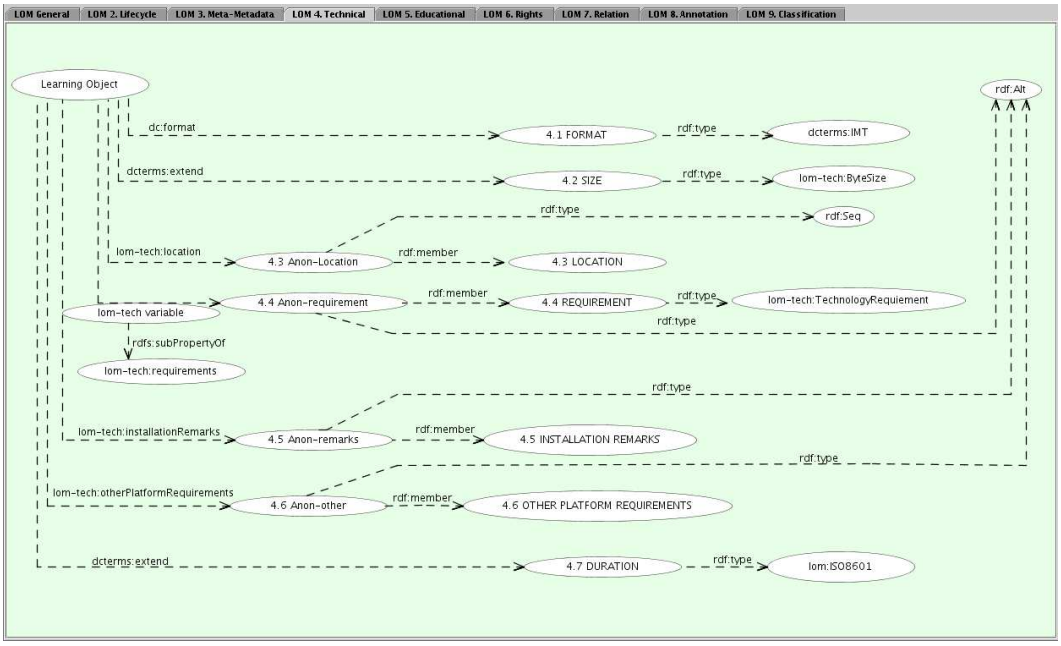


Figure 1: The RDF-Query model for LOM category 4.Technical

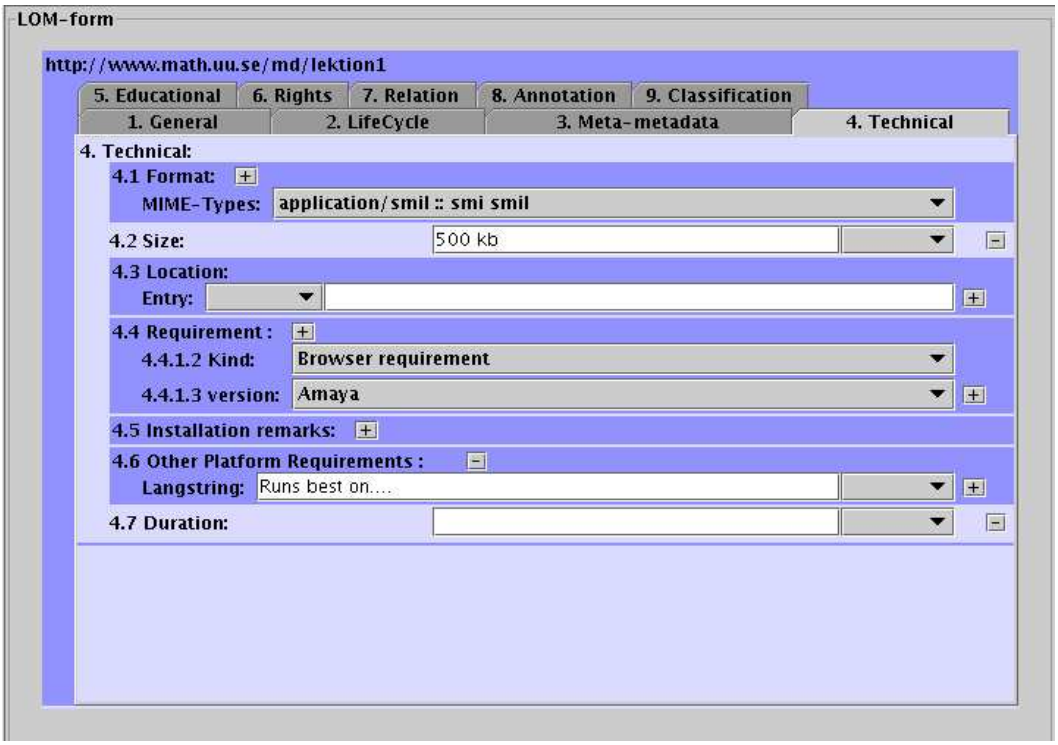


Figure 2: The LOM Category 4.Technical in the RDF Editor