

Intelligently Authoring Metadata for a Semantic Web Peer-to-Peer Environment

Long version

Jan Brase¹, Wolfgang Nejdl², Mark Painter³, Michael Sintek⁴, and Uwe Thaden²

¹ Information System Institute, University of Hannover

² Learning Lab Lower Saxony, Hannover

³ Institute for Communications Technology, Braunschweig Technical University

⁴ DFKI GmbH, Kaiserslautern

Abstract. Learning Objects Metadata (LOM) aims at describing educational resources in order to allow better reusability and retrieval. Unfortunately, annotating complete courses thoroughly with LOM metadata can be a tedious task. In this paper we show how additional inference rules can make this task easier, and allows us to derive additional metadata from existing ones. Additionally, using these rules as integrity constraints helps us to define the constraints on LOM elements, thus taking an important step toward a complete axiomatization of LOM metadata (with the goal of transforming the LOM definitions from a simple syntactical description into a complete ontology). In this paper we will use RDF metadata descriptions and an inference language explicitly developed for RDF (TRIPLE) to represent metadata and axioms. We show how these rules can be applied for the extensions of course metadata using an existing test bed with several courses. Based on the Edutella peer-to-peer architecture we can easily make RDF metadata accessible to a whole community using Edutella peers that manage RDF metadata. By processing inference rules we can achieve better search results. In the appendix you find the complete LOM table extended with inference rules.

1 Motivation

1.1 Metadata

In the vision of the Semantic Web the vast amount of information (i.e. resources) already available via the Web to date will be described and annotated comprehensively, allowing intelligent applications and retrieval. One of the cornerstones for the Semantic Web is the Resource Description Framework RDF. RDF enables the creation and exchange of resource metadata as normal Web data.

At universities an increasing amount of electronic supplemental resources are available or under development for the various curricular activities. These can be small modular resources like animations or video sequences or complete courses in a more complex form. Instructive resources from universities are usually available for free. One yet unsolved problem is the precise access to electronic resources for educational purposes. To solve this problem metadata standards are available that allow to describe

educational resources. The Dublin Core (DC) element set defines a set of basic metadata elements such as Title, Creator, Subject or Description, which are the most relevant for any search and retrieval functionality[1]. Besides Dublin Core the most important one in the domain of educational metadata is Learning Objects Metadata (LOM) [2], which is an official IEEE standard since July 2002. LOM comprises 46 elements that are categorized into the nine categories General, Life Cycle, Meta-Metadata, Technical, Educational, Rights, Relation, Annotation and Classification. For several metadata elements LOM provides vocabulary sets with recommended values. One example is the description of the kind of relationship between one resource and another, where the IEEE LOM vocabulary includes the elements {IsPartOf, HasPart, IsVersionOf, HasVersion, IsFormatOf, HasFormat, References, IsReferencedBy, IsBasedOn, IsBasisFor, Requires, IsRequiredBy}, which has been directly adopted from Dublin Core [3].

At the Institute for Communications Technology, Braunschweig Technical University, and the Information Systems Institute, Hannover University, several electronic courses have been developed in previous projects (e.g. Signal Transmission II [4] and Artificial Intelligence I [5]). To allow sophisticated access to parts of these courses following a user query the content of the courses needs to be described comprehensively using the described standards. For the annotation of the courses *Signal Transmission II* and *Artificial Intelligence* a subset of both Dublin Core and LOM metadata has been selected. The following specifications have been used:

- Dublin Core
- LOM
- Dublin Core Qualifiers
- LOM Classification
- vCard

Sophisticated tools to describe or annotate learning resources using RDF (Resource Description Framework) [6] and the described metadata standards are not available up to date. Therefore the metadata for the listed courses have been created using a standard text editor without any specialized functions for XML data. Naturally this is not the recommended way. Intelligent tools can help the metadata creator at this point with entering the metadata records.

In addition to the trivial metadata elements (such as title, creator, description, ...) structural relationships between parts of a course have been described with the Dublin Core qualifiers like *hasPart* and *hasVersion*. A comprehensive metadata record for a complete course can easily consist of several hundred lines of RDF code. To access specific content of a given course requires that all resources are accurately described by structural relationships and relationships in terms of logical sequences of content.

To classify the single course assets specific classification schemes have been employed. For the computer science domain the ACM CCS classification [7] and for the engineering domain the Engineering Index EI classification have been used [8]. Both classification schemes have been coded in RDF for improved machine-readability [9][10]. The ACM classification has been refined in certain taxologies to allow better characterization of the topic of course elements.

Our metadata elements have been encoded in RDF and RDF Schema (for the full RDF descriptions of the courses cf. [11] and [12]). In addition to the trivial metadata

elements (such as title, creator, description, ...) structural relationships between parts of the courses have been described with the Dublin Core terms qualifying the relationship between two resources. For a complete description of the metadata set we used to annotate courses we refer to [13] or [14].

One ambitious project at the Learning Lab Lower Saxony, Hannover, Germany, is Edutella in which the development and implementation of a peer-to-peer architecture for the exchange of educational resources is focused [15]. This architecture relies on RDF as the basic syntax for metadata.

1.2 Motivation and Problem Description

Our motivation in describing the courses with LOM metadata was to achieve better retrieval results when searching for educational content and to allow more precise queries. This requires that all resources of a given course are accurately described by structural relationships and relationships in terms of logical sequences of content (cf. Figure 1).

Doing that, the first thing we noticed was that several fields are redundant in that they can be easily derived from other fields. In some cases, metadata elements are simply inverse attributes to other ones: The qualified relationship “hasPart” between two resources implies the inverse relationship “isPartOf” where RDF subject and object are interchanged (i.e. the directed RDF arc between both is reversed). What is necessary here is clearly a set of logical rules which can be processed by an inference engine to create all these implicit metadata elements or RDF statements from the existing ones and add them to get complete annotations.

In this paper, we will use TRIPLE [16], a modular rule language that has been designed for querying and transforming RDF statements, which makes it very suitable for our purposes.

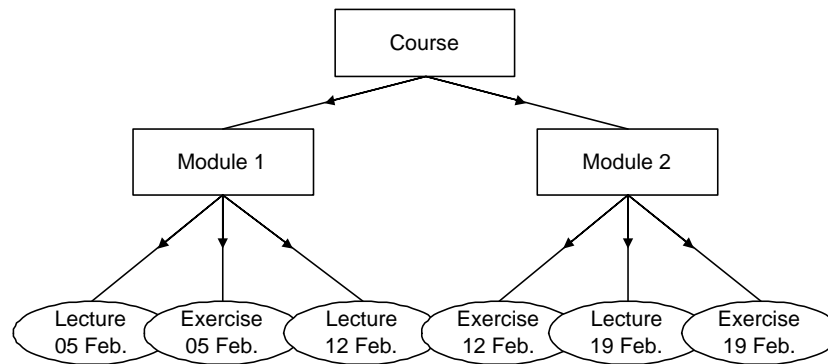


Fig. 1. Hierarchical Structure of a Course Defined via *dcterms:hasPart*

Another point to notice is that the specifications for the LOM data model are mainly on the syntactical level, but leave out important semantical information. What is needed

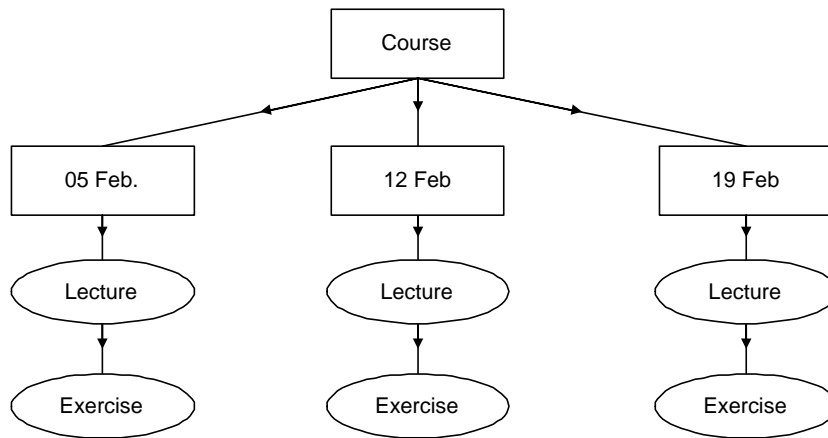


Fig. 2. Temporal Structure of a Course Defined via *dcterms:hasPart*

here are axioms (we can use the inference rules mentioned above as integrity constraints) which provide a formal basis for a more precise description of the usage of all LOM elements. One example is the *is Part Of* relationship between two resources. In our definition this relationship describes the hierarchical structure in terms of course modules. If we have an exercise that is part of a course module then the subject of the exercise can be inherited to the course module along the *is Part Of* relationship, confer figure 1.

However one could use an “is Part Of” qualifier in terms of a temporal relationship as shown in figure 2. Often the sequence of topics in both lecture and exercise are not completely synchronized, so that inheriting subject attributes along the relationship is not reasonable. At this point, adding axioms (which hold for our interpretations but not for other ones) is an important means for adding semantical information and thus clarifying how we use the LOM metadata elements in our context, thus improving the exchangeability of LOM metadata records between different applications. Informally, our use of these relations is defined in the following table; we will show appropriate axioms formalizing this meaning in a later section.

Relationship	Description
<i>dcterms:hasPart</i> <i>dcterms:isPartOf</i>	The hierarchical structure of a course is defined with this relationship. Confer figure 1
<i>dcterms:hasVersion</i> <i>dcterms:isVersionOf</i>	Resources use this relationship if they are versions of the same content, but differ, for example, in the creator or the language. It is possible that one resource has different versions.
<i>dcterms:hasFormat</i> <i>dcterms:isFormatOf</i>	Resources are formats of each other, if they only differ in their technical format, e.g. the slides and the video capture of the same lecture.

1.3 Ontologies

Ontologies, which have recently got a lot of attention in the context of the Semantic Web, provide a shared and common understanding of a domain that can be communicated between people and application systems like agents. They are developed to facilitate knowledge sharing and reuse [17]. In the simplest case, an ontology describes a hierarchy of concepts related by subsumption relationships. In more sophisticated ontologies, suitable axioms are added in order to express other relationships between concepts and to constrain their intended interpretation [18].

In our context we start from a comprehensive though not complete description of learning resources by metadata elements. These metadata elements (represented in RDF, the main language of the Semantic Web) provide a specific model of our learning resources in terms of (hierarchical) structure and using attributes to express keywords and creator information. Adding axioms here helps both in completing our metadata annotations, as well as checking their consistency with regards to an intended interpretation.

1.4 TRIPLE

The rule language TRIPLE has been especially designed for applications in need of RDF reasoning and transformation and is therefore an very good choice for the definition of our rules and axioms. For a complete description of TRIPLE we refer to [16]. As we see in the next section, TRIPLE can also be used for a formal description of LOM elements. The TRIPLE core language is based upon Horn logic which is syntactically extended to support RDF primitives like namespaces, resources and statements (triples, which gave TRIPLE its name). Namespaces are declared via clause-like constructs of the form `nsabbrev:= namespace`. An RDF statement (triple) is inspired by F-Logic object syntax written as

```
subject [ predicate → object ]
```

TRIPLE uses the usual set of connectives and quantifiers for building formulae from statements and Horn atoms, i.e. AND, OR, NOT, FORALL, EXISTS, \leftarrow , \rightarrow , etc. The following example shows how we can use TRIPLE for inferencing:

```
rdf:= "http://www.w3.org/...rdf-syntax-ns#".
dc:= "http://purl.org/dc/elements/1.0/".
dfki:= "http://www.dfki.de".
```

```
@dfki:documents{
  dfki:d_01_01 [
    dc:title→ "TRIPLE" ;
    dc:creator→ "Michael Sintek" ;
    dc:creator→ "Stefan Decker" ;
    dc:subject→ "RDF" ;
    dc:subject→ "triples; ... ].
```

```
inheritance_along(dcterms:hasPart,dc:language).
```

```

FORALL  $S, D$  search( $S, D$ ) ←
   $D[\text{dc:subject} \rightarrow S]$ .

FORALL  $DI, V, P1, P2$   $DI[P2 \rightarrow V] \leftarrow$  EXISTS  $D2$ 
  ( $DI[P1 \rightarrow D2]$  AND
   $D2[P2 \rightarrow V]$  AND
  inheritance_along( $P1, P2$ )).

FORALL  $D2, S$   $D2[\text{dc:subject} \rightarrow S] \leftarrow$  EXISTS  $DI$ 
  ( $DI[\text{dc:subject} \rightarrow S]$  AND
   $D2[\text{dcterms:isFormatOf} \rightarrow DI]$ ).
}

```

In this example, first we define three namespaces for RDF, Dublin Core and for the domain <http://www.dfki.de/>. A document with the reference ID `d_01_01` is then being described with Dublin Core metadata, e.g. `dfki:d_01_01 [dc:title→"TRIPLE"]` (several statements can be abbreviated as shown in the above example). The subject of the related RDF statement is `d_01_01`, the predicate is `dc:title` and the object is "TRIPLE". The example includes three rules: The first searches for documents D having the specified subject S . The second rule inherits the DC language attribute from one document $D2$ that is part of the document DI to the latter. The last rule inherits the subject attribute to documents that are available in other technical formats.

How can these rules help with the creation and processing of RDF-encoded LOM records? We present in this paper three applications using TRIPLE. We can define rules that can be processed in order to check metadata consistency, allow semi-automatic metadata creation and enrich search results. As a beneficial side effect we can provide a more formal definition of LOM elements extending the one provided in [2].

1.5 Project Background

Our courses have been created in previous eLearning and research projects financed by the Federal Ministry of Education and Research, Germany, and the Ministry for Science and Culture of Lower Saxony, Germany. These resources support several lectures at our universities and are organized as hierarchically structured courses written primarily in HTML, using some additional embedded multimedia elements. We are currently using these resources also as test beds for various other projects that focus on intelligent applications to support e-learning. In one of these projects, the Edutella Project [15], we develop a peer-to-peer infrastructure for learning (and other digital) resources. To allow better reusability and retrieval of these resources in a peer-to-peer network all resources are with LOM metadata. Other applications processing these metadata records include digital libraries systems and learning management systems (LMS).

2 Inference Rules and Axioms for a Formal Description of LOM

In the Appendix you will find the complete table of LOM elements, expanded with their inference rules. In this section we will only discuss the rules and some examples.

2.1 Rules

In the following rules, we use $R, R1, R2, \dots$ as abbreviation for learning resources. $P, P1, P2, \dots$ represent predicates from the LOM standard, $O, O1, \dots$ are values of a predicate. The metadata attributes from dterms that define relations between resources as *dterms:hasPart*, *dterms:hasVersion*, etc. play an important role in our annotation, because most of our inference rules are especially useful when we take the relations between learning resources into account. In the following, we use A as a placeholder since many of the rules work for different attributes.

Inverse Attributes The first rules, quite obvious, describe the fact that some attributes have inverse attributes. If there is a *dterms:hasPart* relationship between R1 and R2, than there has to be also a *dterms:isPartOf* relationship between R2 and R1. Inverse predicates are marked in the LOM table with *inverse(A1,A2)*. The rule is defined as:

```
FORALL R1, R2, A1
  R1 [A1->R2 ]<- EXISTS A2
  (R2[A2->R1]AND
  inverse(A1, A2)).
```

```
inverse(A1, A2) <- inverse(A2, A1).
```

Transitive Attributes Transitive attributes, like *dterms:hasPart*, are marked in the LOM table with **transitive(A)**. The rule is defined as:

```
FORALL R1,R3,A
  R1[A ->R3]<- EXISTS R2
  (R2[A ->R3]AND
  R1[A ->R2]AND
  transitive(A)).
```

Inheritance Predicates can also be inherited along certain attributes. As the attribute *dterms:hasPart* and *dterms:isPartOf* are used to structure a course, a lot of predicates like *1.3 Language*, *1.5 Keyword*, *2.2 Status*, etc. can be inherited from a lecture unit to the whole lecture, expressed via the following inference rule:

```
FORALL R1,P,O
  R1[P ->O]<- EXISTS R2, A
  R1[A ->R2]AND
  (R2[P ->O]AND
  inheritance_along(A,P)).
```

Predicates that are inherited in such a way along a certain attribute are marked in the LOM table with **inheritance_along(A,P)**.

A special situation occurs for the predicate *7.1 Relation Kind* where the metadata instance *dterms:requires* is used, to describe the background knowledge for a learning resource. The value of this predicate is only inherited along a *dterms:hasPart*,

for example, if the learning resource providing the background knowledge is not also connected via `dcterms:hasPart`. Situations like this can be handled with the following inference rule:

```
FORALL R1,R3,P
  R1[P ->R3]<- EXISTS R2,A
  (R1[A ->R2]AND
  R2[P ->R3]AND
  NOT R1[A ->R3]AND
  outwardInheritance_along(A,P)).
```

Predicates that are inherited in such a way along a certain attribute are marked in the LOM table with **outwardInheritance_along(A,P)**.

Some inverse relationships like `dcterms:hasFormat` and `dcterms:isFormatOf` are so strong that every predicate value from a resource is inherited to its related resources. The following inference rule describes this fact:

```
FORALL R1,P,O
  R1[P ->O]<- EXISTS R2,A1,A2
  (R1[A1 ->R2]) OR (R1[A2 ->R2])
  AND R2[P ->O]AND
  inverse(A1,A2) AND
  inverseInheritance_along(A,P).
```

Predicates that are inherited in such a way along a certain attribute are marked in the LOM table with **inverseInheritance_along(A,P)**.

Aggregation Sometimes the value of a predicate is the sum of values of predicates from other resources. For example, if a resource is separated in different parts via `dcterms:hasPart`, the size of the resource as defined with the predicate *4.2 Size* is the sum of the parts' sizes. For this nontrivial task we use a TRIPLE aggregator:¹

```
FORALL R1,P,S
  R1 [P ->S ]<- EXISTS A
  (sum_along(A,P) AND
  S is sum{O |EXISTS R2 (R1 [A ->R2 ]AND R2 [P ->O ])}).
```

Predicates that are added in such a way along a certain attribute are marked in the LOM table with **sum_along(A,P)**.

The “aggregation” level of a resource (see, e.g., predicate *1.8 Aggregation level*) can take a value between 1 and 4. The LOM standard defines that a collection of level 1 resources (e.g. raw media or fragments) forms a resource of level 2 (e.g. a lesson). The same applies to collections of level 2 or level 3 resources. The value of the predicate “aggregation level” for a resource that has certain child resources identified via `dcterms:hasPart` can then be defined as the maximum value of the aggregation level of the child resources plus 1. The following rule is valid if the maximum value of the aggregation level of all child resources is less than 4.

¹ TRIPLE aggregators have the form $agg\{V \mid Q(V)\}$ with $agg \in \{\text{sum, avg, max, . . .}\}$, just as in the F-Logic FLORA.

```

FORALL R1,P,S1
  R1 [P ->S1]<- EXISTS A,S
    (maxSum_along(A,P) AND
     S is max{O | EXISTS R2(R1[A ->R2]AND R2[P ->O])}
     AND ((S <4 AND S1 is S+1) OR S1 is 4)).

```

Predicates that are aggregated in such a way along a certain attribute are marked in the LOM table with **maxSum_along(A,P)**.

2.2 Inference Rules for Content Classification

To classify the content of a learning object the IMS binding guide [21] suggests to link the attribute *dc:subject* to an ontology that is available as an RDF file in the internet and is structured using the attribute *lom_cls:taxon*. A detailed description about the use of ontologies for the content classification of learning resources can be found in [13]. If this semantic structure can also be accessed by our TRIPLE engine, we can formulate the following inference rule. Inferred facts use a new attribute, *relatedTopic*. Queries are then formulated using *dc:subject*. If there are no results we proceed to formulate a query with *relatedSubject*.

```

FORALL R,O2
  R[relatedSubject ->O2]<- EXISTS O1
    (R[dc:subject ->O1]AND
     (O1[lom_cls:taxon ->O2]OR
      O2[lom_cls:taxon ->O1])).

```

3 Usage in a Metadata Authoring Environment

Now that we have given rules and axioms for LOM attributes, what have we gained? We see two major fields of applications: validation and semi-automatic creation of metadata annotations, and enrichment of search results using TRIPLE views. Let us have a closer look at these possibilities.

3.1 The ULI Test Bed

The ULI project (University teaching network for computer science) is funded by the German government, and tries to establish an exchange of course material, courses and certificates in the area of computer science. 11 German universities with 18 professors have agreed to exchange their courses and to allow students from one university to attend courses at another university, using advanced eLearning technologies. For more information about the project, we refer to [22]. We have used this test bed to experiment with different kinds of annotations for the learning materials in these courses.

Though the courses usually differ in the kind and amount of learning materials they use, their use of learning resources is surprisingly homogeneous. The average course is divided in 6 to 7 units or knowledge modules which themselves can be split into 3 to 7 learning resources. This leads to an average number of about 35 learning resources per

course, with a learning resource being the slides of the lecture, a video or any other set of pages dealing with one subject.

For annotating these ULI resources, we defined a best-practice subset of 15 elements which is summarized in Table 1, using the categories defined in LOM, extended by our inference rules.

It turned out that these 15 attributes are enough to annotate and query our resources and represent a compromise between more abstract and more detailed annotation sets. The annotations of one whole course can be included in a single RDF file. Some querying tools like the Edutella peer discussed in the next section allow querying over the RDF file, using the RDF query language RDQL (cf. [23] for more details).

Using the inference rules described in the earlier chapter, we were able to create new expanded RDF files for each course. Extended with the implicit information about the course material, querying this files enhanced our query results in the context of the Edutella project.

3.2 Querying Course Descriptions Using Edutella File-based Peers

The Edutella network provides a peer-to-peer infrastructure for educational materials annotated with RDF. The following sections give an overview how the data in Edutella are stored and how the information can be queried. We conclude this with an example setup for ULI courses to show how annotated course material can be published within the Edutella network. In our context a peer is a computer that stores RDF metadata descriptions of courses. It is also called provider-peer or provider from here. A consumer is a peer that sends queries to the network and retrieves the results from the provider peers. It normally has some kind of user interface that lets user define queries (s. Fig. 4).

Edutella is based on the JXTA framework. JXTA is an Open Source project supported and managed by Sun Microsystems. In essence, JXTA is a set of XML based protocols to cover typical P2P functionality. It provides a Java binding offering a layered approach for creating P2P applications. In addition to remote service access (such as offered by SOAP), JXTA provides additional P2P protocols and services, including peer discovery, peer groups, peer pipes, and peer monitors. Therefore JXTA is a very useful framework for prototyping and developing P2P applications. For further information about the JXTA project we refer to [24].

For storing and exchanging information in the Edutella network a data model was introduced (ECDM). This data model allows to describe all query-information in the language QEL. One important advantage of Edutella is the possibility to convert from ECDM/QEL to different provider query languages with so called wrappers (SQL, RDQL, Google, OLR, ...). Figure 3 illustrates this.

The consumer interface allows the user to build a query which is then parsed into the QEL ECDM. The query is sent to the peers in the network. Providers convert the QEL query back into their native query language, e.g. SQL or RDQL.

We use a file-based provider to publish the ULI course metadata in our test bed. This provider is based on a knowledge base which consists of files containing all RDF metadata for the course annotations. Currently, we support RDQL to query the knowledge base.

Table 1. LOM Elements and the Corresponding Inference Rules

LOM category	Metadata name	used attribute	inference rules
1. General	1.2 Title	dc:title	none
	1.3 Language	dc:language	inheritance_along (dcterms:hasPart ,Language).
	1.4 Description	dc:description	inheritance_along (dcterms:hasPart ,Description).
2. Lifecycle	2.3 Contribute	dc:creator with a lom:entity and the author in vCard format dcq:created with the date in W3C format	inheritance_along (dcterms:hasPart ,Entity). inheritance_along (dcterms:hasPart ,Date).
6. Rights	6.3 Description	dc:rights	inheritance_along (dcterms:hasPart ,Entry).
7. Relation		dcq:hasFormat dcq:isFormatOf dcq:hasPart dcq:isPartOf dcq:hasVersion dcq:isVersionOf dcq:requires dcq:isRequiredBy	inverse(dcterms:hasFormat , dcterms:isFormatOf). inverse(dcterms:hasPart , dcterms:isPartOf). inverse(dcterms:requires , dcterms:isRequiredBy). inverse(dcterms:hasVersion , dcterms:isVersionOf). outwardInheritance_along (dcterms:hasPart , dcterms:requires). inverseInheritance_along (dcterms:hasFormat , dcterms:requires). OutwardInheritance_along (dcterms:hasVersion , dcterms:requires). transitive(dcterms:hasPart). transitive(dcterms:isPartOf).
9. Classification		dc:subject for content classification. This attribute links to an entry in a hierarchical ontology, that is an instance of lom_cls:Taxonomy (see next section)	inheritance_along (dcterms:hasPart ,Entry). inverseInheritance_along (dcterms:hasFormat ,Entry). inheritance_along (dcterms:hasVersion ,Entry).

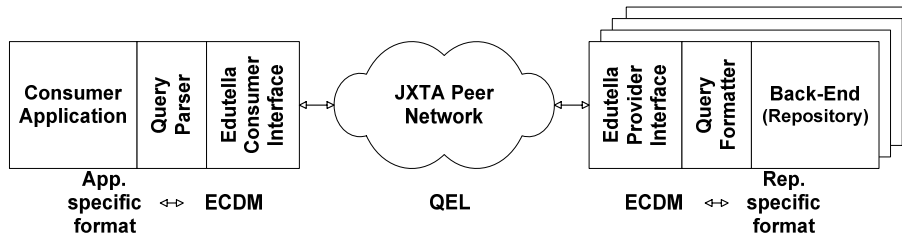


Fig. 3. Query Processing in Edutella

Figure 4 shows a query result set based on the ULI AI course. The search was for I.2.8.0 (I. Computing Methodologies / ARTIFICIAL INTELLIGENCE / Problem Solving, Control Methods, and Search / Backtracking) in the ACM classification. The only result is a single learning resource. There is no author annotated because there is only one author for the complete course.

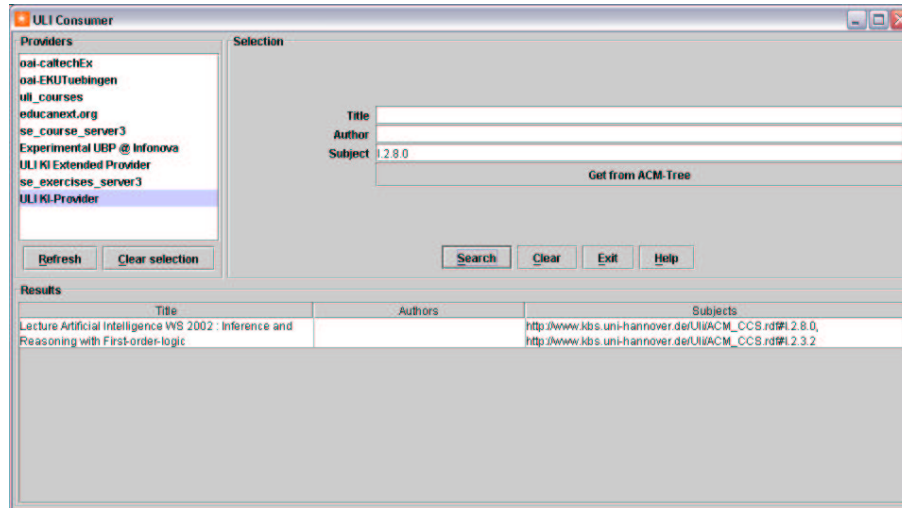


Fig. 4. Query Results on Not-inferred Data

Querying the RDF files that were extended using our inference rules, the file-based provider will provide a result set shown in 5. Now we see the unit to which the resource belongs to and the complete course, because the content information is inherited upward. Also the resource and the unit inherited the author from the course.

ULI Consumer

Providers

- oai-collectEx
- oai-EKUTuebingen
- u1i_courses
- educanext.org
- se_course_server3
- Experimental UBP @ Infnova
- ULI NI Extended Provider**
- se_exercises_server3
- ULI NI-Provider

Selection

Title: _____

Author: _____

Subject: 1.2.8.0

Get from ACM-Tree

Refresh Clear selection Search Clear Exit Help

Results

Title	Authors	Subjects
Lecture Artificial Intelligence I WS 2002 (Hannover)	Wolfgang Nejdl	http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.11.1 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.4.2.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.1.1.4 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.1.6 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.3.6 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.3.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.3.1 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.3.2 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.3.7 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.3.4 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.5 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.7 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.4.2.1 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.4.6 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.3.2.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.4
Lecture Artificial Intelligence I WS 2002 (Hannover): Knowledge-modul 3: Reasoning	Wolfgang Nejdl	http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.3.7 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.4.2.1 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.3.2
Lecture Artificial Intelligence WS 2002: Inference and Reasoning with First-order-logic	Wolfgang Nejdl	http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.8.0 , http://www.kbs.uni-hannover.de/ULI/ACM_CCS/rd/1.2.3.2

Fig. 5. Query Results on Inferred Data

3.3 Supporting the Creation of Course Descriptions

If we use the set of rules given as integrity constraints, we can test whether a set of metadata attributes for a course is well-defined with respect to our intended meaning. Such a “validator” for metadata course descriptions based on TRIPLE could be part of any tools used for creating these descriptions, and could also be used to check whether metadata to be merged from different resources follow the same semantics.

Using our rules as inference rules allows for semi-automatic annotation / extension of course metadata. As the inference rules for the hasPart relationship define that an attribute like dc:creator is inherited from a learning resource to the module and the course it is a part of, we can extend our metadata using this rule the other way around. The creator of the metadata descriptions can then annotate the top course element with an author field using dc:creator and the annotation tool will use the inference rules to suggest the use of this author as a default setting for every learning resource.

3.4 TRIPLE Views

One major feature of the TRIPLE language is the possibility to create different views of metadata sets using inference rules. Since we have already written our rules in TRIPLE, we can use them to create different views over existing metadata sets. Let us assume, that we have a course description *course* in which we search for learning resources with a certain dc:subject entry. If the results we received are not enough for our purpose, we can expand the search on a special *contentview* of the description, created via:

```
FORALL course @contentview(course) {
  FORALL S,P,O
    S [P ->O ]<-
    S [P ->O ]@ course.
  FORALL R,C2
    R[dc:subject ->C2]<- EXISTS C1
    (R[dc.subject ->C1]AND
    C1[lom_cls:taxon ->C2]).
  FORALL R1,C
    R1[dc:subject ->C]<- EXISTS R2
    (R1[dcterms:hasVersion ->R2]AND
    R2[dc:subject ->C]).
  FORALL R1,C
    R1[dc:subject ->C]<- EXISTS R2
    ((R1[dcterms:hasFormat ->R2]OR
    R1[dcterms:isFormatOf ->R2]) AND
    R2[dc:subject ->C]).
  FORALL R1,C
    R1[dc:subject ->C]<- EXISTS R2
    (R1[dcterms:hasPart ->R2]AND
    R2[dc:subject ->C]). }
```

This view is created from the original description using every inference rule that concerns the *dc:subject* attribute. A search on this expanded view will also find resources that cover super topics of the original dc:subject entry. We also find resources that have parts that cover this topic, that are formats of resources covering this topic etc.

4 Conclusion and Further Work

We have shown in this paper how to complement LOM metadata by suitable rules and axioms expressed in the TRIPLE language, and have discussed how extending the purely syntactic definition given in the LOM specifications by such rules and axioms can help us in various ways: Inference rules can help us avoid redundant metadata annotation and derive additional metadata from existing ones. Using these rules as integrity constraints helps us to define the constraints on LOM fields, making clear our intended meaning and use of these LOM fields, resulting in easier exchange of LOM metadata between different applications and contexts. Finally, we have shown how we can use the view mechanism of TRIPLE to extend existing metadata sets to provide more complete answers to our queries.

We are working to include these technologies in the context of an annotation tool based on the Ontomat plug-in by the university of Karlsruhe (cf. [25]). This tool will hopefully be finished this summer and allow everyone to easily create a metadata course description. Combined with the Edutella file-based peer it should form a "starter-kit" for the Edutella P2P network.

References

1. Dublin Core Metadata Initiative
URL: <http://dublincore.org/>
2. IEEE Learning Technology Standards Committee (LTSC), IEEE P1484.12 Learning Object Metadata Working Group.
URL: <http://ltsc.ieee.org/wg12/>
3. Dublin Core Qualifiers
URL: <http://www.purl.org/dc/terms/>
4. Course *Signal Transmission I and II*
URL: <http://www.ifn.ing.tu-bs.de/sue/>
5. Course *Artificial Intelligence I*
URL: <http://www.kbs.uni-hannover.de/Lehre/KI1/WS02/>
6. Resource Description Framework
URL: <http://www.w3.org/RDF/>
7. The ACM Computing Classification System [1998 Version]
URL: <http://www.acm.org/class/1998/>
8. Classification Hierarchy adopted from Engineering Information Inc., USA
URL: <http://eels.lub.lu.se/>
9. The ACM Computing Classification System [1998 Version] coded in RDF
URL: http://www.kbs.uni-hannover.de/Uli/ACM_CCS.rdf
10. Engineering E-Library Sweden (EELS) Classification coded in RDF
URL: <http://www.ifn.ing.tu-bs.de/tv/painter/eels/eels.rdf>
11. RDF Description of the course *Signal Transmission II*
URL: <http://www.ifn.ing.tu-bs.de/sue/sue.rdf>
12. RDF Description of the course *Artificial Intelligence I*
URL: http://www.kbs.uni-hannover.de/Uli/ULI_KI.rdf
13. J. Brase, W. Nejdl *Ontologies for eLearning*
Technical Report, to be published in *Handbook on Ontologies*, Springer 2003

14. J. Brase, W. Nejdl *Annotation for an open learning repository for computer science*
Technical Report, to be published in *Annotation for the Semantic Web*, IOS-press 2003
15. W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, et al. *Edutella: A P2P Networking Infrastructure Based on RDF*, 11th International World Wide Web Conference (WWW2002), Hawaii, USA, May 2002.
16. M. Sintek and S. Decker: *TRIPLE—A Query, Inference, and Transformation Language for the Semantic Web*, 1st International Semantic Web Conference (ISWC2002), Sardinia, Italy, June 2002.
17. D. Fensel. *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, Springer Verlag, Heidelberg, 2001
18. N. Guarino (ed.). *Formal Ontology in Information Systems*, Proceedings of FOIS'98, Trento, Italy, 6-8 June 1998
19. *Long Version of this paper*
URL: http://www.kbs.uni-hannover.de/Arbeiten/Publikationen/2003/iswc03_long.pdf
20. *The web ontology language (OWL)—Abstract syntax and semantics*, W3C working draft Feb 2003
URL: <http://www.w3.org/TR/owl-semantics/>
21. M. Nilsson. *RDF binding of LOM metadata*, CID, KTH Stockholm, Sweden, May 2001.
URL: <http://kmr.nada.kth.se/el/ims/metadata.html>
22. *The ULI Project Homepage*
URL: <http://www.uli-campus.de>
23. *RDQL—RDF Data Query Language*
URL: <http://www.hpl.hp.com/semweb/rdql.htm>
24. *The JXTA Project Homepage*
URL: <http://jxta.org>
25. *The OntoMat homepage*
URL: <http://annotation.semanticweb.org/tools/ontomat>

A The Learning Objects Metadata Standard Schema LOM - extended with inference rules

Nr	Name	Value space	Inference Rules
1	General		
1.1	Identifier	-	none
1.2	Title	-	none
1.3	Language	LanguageID = Langcode	inheritance_along(dcterms:hasPart ,Language).
1.4	Description	-	inheritance_along(dcterms:hasPart ,Description).
1.5	Keyword	-	inheritance_along(dcterms:hasPart ,Keyword). inverseInheritance_along(dcterms:format ,Keyword). inheritance_along(dcterms:hasVersion ,Keyword).
1.6	Coverage	-	inheritance_along(dcterms:hasPart ,Coverage). inverseInheritance_along_ dcterms:format (Coverage). inheritance_along(dcterms:hasVersion ,Coverage).
1.7	Structure	atomic, collection, networked, hierarchical linear	none, should be defined by the author
1.8	Aggregation Level	1,2,3,4	maxSum_along (dcterms:hasPart ,AggregationLevel).
2	Life Cycle		
2.1	Version	-	none
2.2	Status	draft, final revised, unavailable	inheritance_along(dcterms:hasPart ,Status).
2.3	<i>Contribute</i>		
2.3.1	Role	author, publisher, unknown, ...	inheritance_along(dcterms:hasPart ,Role).
2.3.2	Entity	vCard	inheritance_along(dcterms:hasPart ,Entity).
2.3.3	Date	Datatype: DateTime	inheritance_along(dcterms:hasPart ,Date).
3	Meta-Metadata		
3.1	Identifier	-	none
3.3	<i>Contribute</i>		
3.2.1	Role	creator, validator	inheritance_along(dcterms:hasPart ,Role).
3.2.2	Entity	vCard	inheritance_along(dcterms:hasPart ,Entity).
3.2.3	Date	Datatype: DateTime	inheritance_along(dcterms:hasPart ,Role).
3.3	Metadata Schema	Repertoire of ISO/IEC 10646-1:2000	inheritance_along (dcterms:hasPart ,Metadata Schema).
3.4	Language	see 1.3	inheritance_along(dcterms:hasPart ,Language).
4	Technical		
4.1	Format	MIME types	inheritance_along(dcterms:hasPart ,Format).
4.2	Size	ISO/IEC 646:1991	sum_along(dcterms:hasPart ,Size).
4.3	Location	Repertoire of ISO/IEC 10646-1:2000	none
4.4	Requirement	see LOM-Standard	inheritance_along(dcterms:hasPart ,Requirement).
4.5	Installation Remarks	-	inheritance_along (dcterms:hasPart ,Installation Remarks).
4.6	Other Platform	- Requirements	inheritance_along (dcterms:hasPart ,Other Platform Requirements).
4.7	Duration	Datatype: Duration	sum_along(dcterms:hasPart ,Duration).

5	Educational		
5.1	Interactivity Type	-	inheritance_along (dcterms:hasPart ,Interactivity Type).
5.2	Learning Resource Type	exercise, simulation, questionnaire, ...	inheritance_along (dcterms:hasPart ,Learning Resource Type).
5.3	Interactivity Level	low, medium, high, very high	none, should be defined by the author
5.4	Semantic Density	low, medium, high, very high	none, should be defined by the author
5.5	Intended End User Role	teacher, author, learner, manager	none
5.6	Context	school, higher education , training, other	none
5.7	Typical Age Range	-	none
5.8	Difficulty	easy, medium, difficult, very difficult	none
5.9	Typical Learning Time	Datatype: Duration	
5.10	Description	-	none
5.11	Language	LanguageID = Langcode	none
6	Rights		
6.1	Cost	yes,no	inheritance_along (dcterms:hasPart ,lom-rights:cost).
6.2	Copyright and Other Restrictions	-	inheritance_along (dcterms:hasPart ,Copyright and Other Restrictions).
6.3	Description	-	inheritance_along(dcterms:hasPart ,Description).
7	Relation		
7.1	Kind	dcterms: hasPart,isPartOf requires, isRequiredBy hasVersion, isVersionOf hasFormat, isFormatOf references, isReferencedBy isBasedOn, isBasisFor	inverse(dcterms:hasPart , dcterms:isPartOf). inverse(dcterms:requires , dcterms:isRequiredBy). inverse(dcterms:hasVersion , dcterms:isVersionOf). inverse(dcterms:hasFormat , dcterms:isFormatOf). inverse(dcterms:references , dcterms:isReferencedBy). inverse(dcterms:isBasedOn , dcterms:isBasisFor). outwardInheritance_along (dcterms:hasPart , dcterms:requires). inverseInheritance_along (dcterms:hasFormat , dcterms:requires). OutwardInheritance_along (dcterms:hasVersion , dcterms:requires). transitive(dcterms:hasPart). transitive(dcterms:isPartOf).
8	Annotation		
8.1	Entity	vCard	inheritance_along(dcterms:hasPart ,Entity).
8.2	Date	Datatype: DateTime	inheritance_along(dcterms:hasPart ,Date).
8.3	Description	-	inheritance_along(dcterms:hasPart ,Description).

9	Classification		
9.1	Purpose	discipline, idea, prerequisite, etc. see the LOM Standard	inheritance_along(dcterms:hasPart,Purpose). inverseInheritance_along(dcterms:hasFormat,Purpose). inheritance_along(dcterms:hasVersion,Purpose).
9.2	<i>Taxon Path</i>		
9.2.1	Source	Repertoire of ISO/IEC 10646-1:2000	inheritance_along(dcterms:hasPart,Source). inverseInheritance_along(dcterms:hasFormat,Source). inheritance_along(dcterms:hasVersion,Source).
9.2.2	<i>Taxon</i>	-	
9.2.2.1	Id	Repertoire of ISO/IEC 10646-1:2000	inheritance_along(dcterms:hasPart,Id). inverseInheritance_along(dcterms:hasFormat,Id). inheritance_along(dcterms:hasVersion,Id).
9.2.2.2	Entry		inheritance_along(dcterms:hasPart,Entry). inverseInheritance_along(dcterms:hasFormat,Entry). inheritance_along(dcterms:hasVersion,Entry).
9.3	Description		inheritance_along(dcterms:hasPart,Description). inverseInheritance_along(dcterms:hasFormat,Description). inheritance_along(dcterms:hasVersion,Description).
9.4	Keyword		inheritance_along(dcterms:hasPart,Keyword). inverseInheritance_along(dcterms:hasFormat,Keyword). inheritance_along(dcterms:hasVersion,Keyword).