

# Ontologies and Metadata for eLearning

## 1.1 Using Metadata for eLearning

eLearning, or online learning, stands for all forms of Internet-enabled and/or computer supported learning. It refers to the use of computer and computer network technologies to create, deliver, manage and support learning, usually independent of specific locations or times. eLearning can involve complete online courses, where all aspects of learning, from learner enrollment to tuition and support take place online. At the other end of the eLearning spectrum, these elements may well take place in a face to face situation, with only the learning resources available on the internet. Accessibility of learning resources is accompanied by the need to annotate the resources with rich, standardized and widely used metadata.

This chapter gives an overview over the use of metadata in eLearning as well as about innovative approaches and techniques we developed for enhanced eLearning scenarios. The first section will give a brief introduction to the field of metadata and metadata bindings. In section 1.2 we will discuss our experiences with these metadata, metadata schemas and metadata annotations in the context of a large computer science testbed. In section 1.3 we will discuss the need for metadata classifications to describe content / topic of a resource. We will introduce different ontologies we used and discuss their use and how they and other kinds of metadata are used to query the for specific resources in more detail. Section 1.4 generalizes the setting from a server based to a peer-to-peer scenario and discusses the Edutella project in more detail, which represents the first RDF-based peer-to-peer network for digital resources and for the exchange of learning objects and services.

### 1.1.1 Relevant metadata standards

Metadata is data about data that helps us to achieve better search results. Instead of hoping that a full text search through a learning resource will find the author's name nejdl for example, we can annotate the resource with a metadata description "author is nejdl". While this seems plausible, we also easily realize the two major

difficulties this method holds: the technical realisation of "attaching metadata at a resource, and the standardisation of descriptions in order to avoid misunderstanding by using different attributes for the same purpose like "author is nejd", "creator is nejd" or "written by nejd".

Let us first take a look on the idea of using specific vocabularies or schemas for metadata to describe digital resources.

One of the most common metadata schemes on the web today is the "Dublin Core Schema" (DC) by the DCMI. The Dublin Core Metadata Initiative (DCMI) [9] is an organization dedicated to promoting the widespread adoption of interoperable metadata standards and developing specialized metadata vocabularies for describing resources, that enable more intelligent information discovery for digital resources.

Each Dublin Core element is defined using a set of 15 attributes from the ISO/IEC 11179 standard for the description of data elements, including for example: Title, Identifier, Language, Comment. To annotate the author of a learning resource DC suggests to use the element creator, and thus we write `dc:creator = nejd`

Whereas "Simple Dublin Core" uses only the elements from the Dublin Core metadata set as attribute-value-pairs, "Qualified Dublin Core" employs additional qualifiers to further refine the meaning of a resource. The DCMI recommends a set of qualifiers called "Dublin Core Qualifiers" (DCQ), which include for example Name, Label, Definition or Comment as alternative qualifiers to refine the Title element. For a complete description, we refer the reader to [9]. Since Dublin Core is designed for metadata for any kind of (digital) resource, it pays no heed to the specific needs we encounter in describing learning resources. The "Learning Objects Metadata Standard" (LOM) [18] by the Learning Technology Standards Committee (LTSC) of the IEEE was therefore established as an extension of Dublin Core. Each learning object can now be described 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

Work on the LOM schema has started in 1998, the current version is 6.4. Once the standard has been accepted, which hopefully will happen 2003, it will be LOM 1.0. Since LOM was developed to be used for any kind of learning resource, LOM users soon find out that they do not really need to use all 70 attributes. On the other hand, regardless of the very useful work that has been done in developing the LOM standard, the standard still fails to specify important educational aspects of learning resources, which lead us to investigate ways to extend LOM with additional attributes depending on which educational setting learning objects are used in (see [3] for more

details). However, in the following chapter we will concentrate on a minimal set of metadata attributes we have used in a larger computer science testbed and focus on the use of different topic ontologies useful for this context.

### 1.1.2 Metadata bindings

The second issue to investigate is the technical realisation of how to "bind" metadata to a resource? For this purpose, two possible approaches have been developed in the context of the World Wide Web, based on the XML and RDF formalisms. XML stands for Extensible markup language and was derived from a document description language called SGML (an international standard for structured documents). RDF stands for Resource Description Framework, and was explicitly developed to annotate resources referenced by URIs in the context of the World Wide Web. For learning metadata, bindings have been developed for both formalisms, lately in the context of the IMS Global Learning Consortium. The RDF binding of LOM has been developed by a group led by Michael Nilsson from SweLL (Swedish Learning Lab) [22], with input from our own group and input from the Viennese colleagues from the UNIVERSAL project [12]. We will not discuss the details of these types of binding, but shortly describe, why we chose the RDF-binding to annotate our resources.

XML Bindings define an exchange format for metadata. The metadata might be contained in a database and an XML representation is usually generated on demand, for export to other tools and environments. Thus, an XML metadata record is a self-contained entity with a well-defined hierarchical structure, and there is seldom a natural way to reuse other metadata standards (or specific fields from other standards).

On the other hand, RDF statements are just triples consisting of a subject, a predicate and an object, where the subject is referenced by an URL/URI. So we can annotate resources on the Internet, e.g. a resource at <http://www.xyz.com/resource.html>, with its author's name using RDF as follows:

```
Subject: http://www.xyz.com/resource.html
Predicate: dc:creator
Object:"nejdl"
```

Using the XML syntax proposed by the W3C this statement can be written as:

```
<rdf:RDF
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:dc="http://purl.org/dc/elements/1.1#">
<rdf:Description about=http://www.xyz.com/resource.html>
<dc:creator>nejdl</dc :creator>
</rdf:Description>
</rdf:RDF>
```

The namespace "dc:" refers to an URL containing an RDF schema that describes the structure of the metadata attributes of dublin core, dc:creator in this case. This schema is called the RDF-binding for DC. If we want to describe a resource with rdf and use LOM, we write triples like the one above in a rdf-file and refer to the RDF binding of the LOM attributes we use. A final metadata description is just a set of these triples. The use of namespaces makes it a part of a global network of information, where anyone has the capability of adding metadata to any resource, using standardized or specialized schemas describing these metadata. This modularity of the architecture leads to naturally reusable constructs. The LOM RDF binding is directly compatible with Dublin Core RDF binding, and therefore Dublin Core elements are used directly instead of defining new LOM elements for these DC properties. This of course helps a lot to enhance the interoperability between resources that are annotated with DC and others that are annotated with LOM. In the latest draft for the RDF binding of LOM ([22]) for example, about 80 percent of the LOM elements are defined using DC and DCQ.

## 1.2 General Metadata Elements

### 1.2.1 Elements for distributed computer science education

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 different 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 ([27]). We have used this testbed to experiment with different kinds of annotations for the learning materials in these courses, this chapter describes our current setup.

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 resource, we defined a best-practice subset of 15 elements which is summarized in the following table, using the categories defined in LOM:

1. General	1.2 Title	<b>dc:title</b>
	1.3 Language	<b>dc:language</b>
	1.4 Description	<b>dc:description</b>
2.Lifecycle	2.3 Contribute	<b>dc:creator</b> with a <b>lom:entity</b> and the author in vCard format "name surname" dcq:created with the date in W3C format
6.Rights	6.3 Description	<b>dc:rights</b>
7. Relation		<b>dcq:hasFormat</b> <b>dcq:isFormatOf</b> <b>dcq:hasPart</b> <b>dcq:isPartOf</b> <b>dcq:hasVersion</b> <b>dcq:isVersionOf</b> <b>dcq:requires</b> <b>dcq:isRequiredBy</b>
9.Classification		<b>dc:subject</b> for content classification. This attribute links to an entry in a hierarchical ontology, that is an instance of <b>lom_cls:Taxonomy</b> (see next section)

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. All RDF-triples are then imported into a relational database, to customize the display of the resources described and to query for specific learning resources. Let us take a closer look at the attributes and their use. Our examples contain metadata descriptions of 6 different courses, 5 of them are part of the ULI project:

1. ULIAlgorithmtheorie.rdf : Description of the ULI lecture "Theory of Algorithms" held in winter 2001 in Freiburg by Prof. Ottman.
2. ULIDatenbanken.rdf : Description of the ULI lecture "Databases" held in winter 2001 in Freiburg by Prof. Lausen.
3. ULIInternetApplications.rdf : Description of the ULI lecture "Algorithms for Internet Applications" held in winter 2001 in Karlsruhe by Prof. Schmeck.
4. ULIMultimediatechnik.rdf : Description of the ULI lecture "Techniques for Multimedia" held in winter 2001 in Mannheim by Prof. Effelsberg.
5. ULIKi.rdf : Description of the lecture "Artificial Intelligence" held in winter 2002 in Hannover by Prof. Nejd.

6. Softwaretechnik.rdf : Description of the lecture "Software engineering" held in winter 2001 in Hannover by Priv. Doz. Steimann.

The complete RDF-files can be found at <http://www.kbs.uni-hannover.de/Uli>

### 1.2.2 Details

#### dc:title

The title of a learning resource or a construct. As the title is usually the first thing to be displayed as a result of a query, it should be as explicit as possible.

```
<dc:title >Techniques for Multimedia WS 2001 (Mannheim) Part 2a from 17.10.2001:
Compression1</dc:title>
(Example from course 4.)
```

#### dc:description

Further description of the learning resource, either as a list of keywords as in our example, or as a full text.

```
<dc:description >Internet history, internet technology, IP, DNS, Routing, TCP, IP
and ATM1</dc:description>
(Example from course 3.)
```

#### dc:creator

The creator of the resource will be displayed as a part of a lom:entity in the vCard Format "name surname". Usually the author will appear once in the definition of the course, and is inherited to all parts of the course. If the course contains resources from other authors, these resources will have a new dc:creator annotation.

```
<dc:creator>
  <lom:entity>
    <vCard:FN>Wolfgang Nejd</vCard:FN>
  </lom:entity>
</dc:creator>
(Example from course 5.)
```

#### dcq:created

The time, when the course was created, formatted in W3C format. Usually only one appearance in the definition of the course.

```

<dcq:created>
  <dcq:W3CDTF>
    <rdf:value>2001-09-15</rdf:value>
  </dcq:W3CDTF>
</dcq:created>
(Example from course 4.)

```

### **dcq:hasPart, dcq:isPartOf**

The structure of a course is described using these attributes. A unit for example links with `dcq:hasPart` to its chapters which link with `dcq:isPartOf` back to the unit they belong to (`dcq:isPartOf` could be inferred automatically as inverse property of `dcq:hasPart`).

### **dc:language**

The language of the learning resource. An important search criterium to ensure that you receive only results you can understand.

### **dcq:hasVersion, dcq:isVersionOf, dcq:hasFormat, dcq:isFormatOf**

Two resources are versions of the same content. If a resource has two equivalent versions, e.g. one german, one english, both versions are connected via `dcq:isVersionOf`. If a resource is divided up into smaller versions, we display this hierarchy by annotating the "bigger" resource with `dcq:hasVersion` and the parts with `dcq:isVersionOf`, in order to know in which direction we can inherit properties. In the special case of two resources with identical content, but different technical versions, e.g. slides and videos, we use `dcq:isFormatOf` and `dcq:hasFormat`. The resource that is easier to display is annotated with the "higher-rated" `dcq:hasFormat`.

### **dcq:requires, dcq:isRequiredBy**

We use these attributes, if the content from one resource cannot be understood without knowledge of another resource. The resource receives an `dcq:requires` entry, while the resources with the background information receives a `dcq:isRequiredBy`.

## **1.3 Content Classification with dc:subject**

General metadata annotation as discussed in the last section is useful, but not enough for finding specific resources. Another important metadata description is classifying the content of a learning resource. It is obvious that self-defined keywords can only be a first solution to this problem. To provide for better search results, keywords used should be part of an ontology, in order to specify both sub- and super-topics.

Defining a private ontology for a specific field unfortunately works only in the closed micro world of a single university. To be more general, we therefore decided to use ontologies which are already part of internationally accepted classification systems. In this section we will introduce the three different solutions we came up with.

### 1.3.1 One complete ontology - the ACM CCS

The ACM Computer Classification system ([1]) has been used by the Association for Computer Machinery since several decades to classify scientific publications in the field of computer science. On the basic level, we find 11 nodes that split up in two more levels. Part of the classification hierarchy is reproduced in the following.

- A. General Literature
- B. Hardware
- C. Computer Systems Organization
- D. Software
  - D.0 GENERAL
  - D.1 PROGRAMMING TECHNIQUES
  - D.2 SOFTWARE ENGINEERING
  - D.3 PROGRAMMING LANGUAGES
  - D.4 OPERATING SYSTEMS
  - D.m MISCELLANEOUS
- E. Data
- F. Theory of Computation
- G. Mathematics of Computing
- H. Information Systems
- I. Computing Methodologies
  - I.0 GENERAL
  - I.1 SYMBOLIC AND ALGEBRAIC MANIPULATION
  - I.2 ARTIFICIAL INTELLIGENCE
    - I.2.0 General
    - I.2.1 Applications and Expert Systems
    - I.2.2 Automatic Programming
    - I.2.3 Deduction and Theorem Proving
    - I.2.4 Knowledge Representation Formalisms and Methods
    - I.2.5 Programming Languages and Software
    - I.2.6 Learning
    - I.2.7 Natural Language Processing
    - I.2.8 Problem Solving, Control Methods, and Search
    - I.2.9 Robotics
    - I.2.10 Vision and Scene Understanding
    - I.2.11 Distributed Artificial Intelligence
    - I.2.m Miscellaneous
  - I.3 COMPUTER GRAPHICS
  - I.4 IMAGE PROCESSING AND COMPUTER VISION

- I.5 PATTERN RECOGNITION
- I.6 SIMULATION AND MODELING
- I.7 DOCUMENT AND TEXT PROCESSING
- I.m MISCELLANEOUS
- J. Computer Applications
- K. Computing Milieux

The classification has a fourth level containing unordered keywords, thus including about 1600 entries on all four levels. For our use of the ACM CCS as a classification, we also numbered the keyword lists in the fourth level to receive unique ids like: B.1.1.2 for the keyword "Microprogrammed logic arrays" that is accessible via the taxon path: Hardware(B)/CONTROL STRUCTURES AND MICROPROGRAMMING(B.1)/Control Design Styles(B.1.1)

In the context of the ULI project this classification turned out to fit very well, because it covers the whole field of computer science, just as the different ULI courses cover the whole discipline. Typically a course received approximately 5 classification entries from the ACM CCS, and one entry per chapter was a typical distribution. Therefore classification with ACM CCS is excellent for the exchange of complete knowledge modules. If we look for a taxonomy that allows us to annotate different submodules and small, single learning resources, we have two other possibilities: extending the ACM CCS, or looking for another classification system.

### 1.3.2 More details - extending ACM CCS

In an article for the AI magazine in the mid 80's D. Waltz suggested an extension of the node I.2 Artificial Intelligence of the ACM CCS [28]. He refined the keywords in the fourth level as nodes for two more levels, gaining about 100 more entries focussing on the field of artificial intelligence. As an example, the keyword entry "games" in the node I.2.1 Applications and Expert Systems was extended to:

- I.2.1 Applications and Expert Systems
  - I.2.1.0 Cartography
  - I.2.1.1 Games
    - I.2.1.1.0 Chess
    - I.2.1.1.1 Checkers
    - I.2.1.1.2 Backgammon
    - I.2.1.1.3 Biding Games
    - I.2.1.1.4 Wagering Games
    - I.2.1.1.5 War Games
    - I.2.1.1.6 Games, Other
  - I.2.1.2 Industrial automation
  - I.2.1.3 Law
  - I.2.1.4 Medicine and science
  - I.2.1.5 Natural language interfaces
  - I.2.1.6 Office automation

As we had labelled the keywords of the fourth level for the use of the classical ACM CCS as well, it was easy for us to adapt his suggestions to the modern (1998) version of the ACM CCS, leading to a quite detailed ontology to classify our learning resources in the discipline of Artificial Intelligence.

### 1.3.3 An ontology for a specific sub-discipline - the SWEBOK

For our course in software engineering we found a different solution by using the SWEBOK ontology. The SWEBOK has been developed as a Guide to the Software Engineering Body of Knowledge in context of an IEEE/ACM working group. On their webpage [26] the working group states their goal as follows: "The purpose of this guide is to provide a consensually-validated characterization of the bounds of the software engineering discipline and to provide a topical access to the Body of Knowledge supporting that discipline." Almost 500 software engineering professionals from 41 countries have hierarchically structured the field of software engineering in 10 Knowledge Areas and almost 300 topics, based on their number of publications. This therefore represents a very nice example for a consensually derived ontology in a specific community. The main knowledge areas are:

- Software requirements
- Software design
- Software construction
- Software testing
- Software maintenance
- Software configuration management
- Software engineering management
- Software engineering process
- Software engineering tools and methods
- Software quality

To use the SWEBOK for our courses in the context of exchanging learning resources with other peers, it was important to define mappings between these knowledge areas and the ACM CCS. These mappings were for example:

**D.2.1 Requirements/Specifications** maps with *Software requirements*

**D.2.2 Design Tools and Techniques** maps with *Software engineering tools and methods / Software tools / Software design tools* and of course the whole *Software design* (since **D.2.10 Design** no longer used as of January 1998)

**D.2.9 Management** maps to *Software configuration management* and *Software engineering management*, same as **K.6.3 Software Management**

So we finally had what we needed: One ontology to use in a world wide context of exchanging learning resources in the field of computer science, plus two specialized ontologies to classify the content of our own lectures, detailed enough to differentiate between the content of single learning resources, but mapping perfectly to the global ontology, if other peers want to access the resources.

### 1.3.4 How to use dc:subject

To classify a resource, the IEEE Learning Object RDF Binding Guide (Draft Version) [22] suggests the use of dc:subject with elements of a taxonomy that must be found on the Internet. Such a taxonomy hierarchy is an instance of lom\_cls:Taxonomy and must be formatted in a rdf-file where the topics and subtopics are separated using lom\_cls:Taxon and lom\_cls:rootTaxon.

The RDF-files of our three ontologies can be found at [5], [4] and [6]. The main structure is always as in this example from the SWEBOK ontology:

```
<dcq:SubjectScheme rdf:ID="mySWEBOK">
<rdfs:label>Software Engineering Book of Knowledge Field Classification </rdfs:label>
</dcq:SubjectScheme>

<lom_cls:Taxonomy>
  <lom_cls:rootTaxon>
    <swtOnt:mySWEBOKrdf:about="http://www.kbs.uni-hannover.de/Uli/
    SWT_Ontologie.rdf#SWEnginManagement">
    <rdf:value>Software engineering management</rdf:value>
    <lom_cls:taxon>
      <swtOnt:mySWEBOK rdf:about="http://www.kbs.uni-
      hannover.de/Uli/SWT_Ontologie.rdf#SWEnginMeasurement">
      <rdf:value>Software engineering measurement</rdf:value>
      <lom_cls:taxon>
        <swtOnt:mySWEBOK rdf:about="http://www.kbs.uni-
        hannover.de/Uli/SWT_Ontologie.rdf#MeasSWDevelopment">
        <rdf:value>Measuring software and its development
        </rdf:value>
        </lom_cls:taxon>
      </lom_cls:taxon>
    </lom_cls:rootTaxon>
  </lom_cls:Taxonomy>
```

This extract from the ontology shows the definition of one knowledge area (Software engineering management), refined in this example into one subtopic (Software engineering measurement), which is itself refined into a subtopic (Measuring software and its development).

To annotate our learning resources, we link dc:subject to the entry in the ontology:

```
<rdf:Description rdf:about="http://www.kbs.uni-hannover.de/Lehre/SE/OLR/S1T1.pdf">
<dc:subject rdf:resource="http://www.kbs.uni-hannover.de/Uli/
SWT_Ontologie.rdf#MeasSWDevelopment"/ >
</rdf:Description>
```

All RDF-triples representing the ontology are stored in the database as well. Let us now have a look on how we use the content classification for our own learning resources.

### 1.3.5 Querying our learning resources

We have been using a metadata driven system designed and implemented at our institute to display our courses, the open learning repository (OLR). The examples and figures in this case study come from the OLR2. Our open learning repositories, versions 1 to 3, are connected to an Oracle database, which stores the metadata of a course, but not the content. Therefore, it is easy to include material from different sources throughout the internet in the context of a course, if the copyright is granted. If copyright was restricted for certain resources, we have used `dc:rights`, and an appropriate algorithm to hide these resources from general access. For further information about the OLR we refer to [29]. We will now have a closer look on how the metadata are queried.

Parts of our little example and of the SWEBOK Taxonomy can be found in the database as (in Datalog notation):

```
...
'dc:title'('http://www.kbs.uni-hannover.de/Lehre/SE/OLR/S1T1.pdf',
  "Software engineering course, Hannover WS2001 - slide1")
'dc:language'('http://www.kbs.uni-hannover.de/Lehre/KI/OLR/S1T1.pdf',
  'http://www.kbs.uni-hannover.de/Uli/lang.rdf#en')
'dc:subject'('http://www.kbs.uni-hannover.de/Lehre/KI/OLR/S1T1.pdf',
  'http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf#MeasSWDevelopment')
...
'lom_cls:taxon'
  ('http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf#SWEnginMeasurement',
  'http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf#MeasSWDevelopment')
'rdf:value'('http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf#MeasSWDevelopment',
  "Measuring software and its development")
...
```

When we display a learning resource in the OLR, one can decide between looking at the content, by choosing a layer labelled "content" or looking at their classification-entries, by choosing the layer labelled "taxon".

If you choose "taxon" the system will query the database for `dc:subject` entries via the (simple) Datalog query:

```
?- 'dc:subject'('http://www.kbs.uni-hannover.de/Lehre/KI/OLR/S1T1.pdf',X)
```

The Taxon entry is displayed not with its original entry but the system retrieve the 'rdf:value' of this entry from the taxonomy as the title, and also the taxon-structure identified by "taxon"-attribute. Remember that the complete taxonomy is also stored as triples in the database. In our example the entry 'MeasSWDevelopment' will be displayed as:

Software engineering management / Software engineering measurement / Measuring software and its development

We decided to display the entry as a hyperlink, leading to a brief description of the subject.

### 1.3.6 Related resources

Furthermore, the related resources (Resources with the same classification-entry "MeasSWDevelopment" from the same Taxonomy mySWEBOK) are retrieved from the database via:

```
?- 'dc:subject'(X,'http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf#MeasSWDevelopment
```

And displayed as a hyperlink to the resource.

### 1.3.7 Annotating chapters

Since a course is only represented by an RDF-Schema, only single learning resource are usually annotated via their RDF-description. A whole course or a single lecture is defined by additional relationships between these learning resources (often hierarchic). Therefore, lectures and courses usually inherit their taxon-entries from their child-units. If the user chooses the layer "taxon" while he is not in a learning resource, but looks at the structure of a unit, the sub-units of this unit and their taxon entries are displayed together with the learning resource they come from. Displaying the taxon entries of a whole course works in the same way, it only takes one iteration more to receive the taxon entries from the "grandchildren".

## 1.4 Exchanging Learning Resources World Wide - Edutella

### 1.4.1 Decentralized P2P networks

As discussed before, exchanging learning resources is one of the greatest advantages in eLearning and eTeaching. A system for the exchange of learning resources however has to build on the fact that most universities or departments have already established their own way of storing their learning resources and do so locally in almost

all cases. Since all these institutions are not interested in losing this independence by giving their learning resources away to central "knowledge pools" the best way to establish such an exchange system is by building up a peer-to-peer (P2P) network. These P2P networks have already been successful for special cases like exchanging music files, which is encouraging. However, retrieving a song like "Material Girl from Madonna" does not need complex query languages nor complex metadata, so special purpose formats for these P2P applications have been sufficient. In other scenarios, like exchanging educational resources, queries are more complex, and have to build upon standards like IEEE-LOM/IMS [18, 15] as discussed before, which might even be complemented by domain specific extensions.

Furthermore, by concentrating on domain specific formats, current P2P implementations appear to be fragmenting into niche markets instead of developing unifying mechanisms for future P2P applications. There is indeed a great danger (as already discussed in [10]), that unifying interfaces and protocols introduced by the World Wide Web get lost in the forthcoming P2P arena.

#### 1.4.2 Edutella infrastructure

The Edutella project [11, 21, 20] addresses these shortcomings by building on the W3C metadata standard RDF [17, 7] as the basis for a so-called schema-based P2P network. The project is a multi-staged effort to scope, specify, architect and implement an RDF-based metadata infrastructure for P2P-networks based on the recently announced JXTA framework [13]. The initial Edutella services are *Query Service* (standardized query and retrieval of RDF metadata), *Replication Service* (providing data persistence / availability and workload balancing while maintaining data integrity and consistency), *Mapping Service* (translating between different metadata vocabularies to enable interoperability between different peers), *Mediation Service* (define views that join data from different metadata sources and reconcile conflicting and overlapping information) and *Annotation Service* (annotate materials stored anywhere within the Edutella Network).

The Edutella infrastructure aims to provide the metadata services needed to enable interoperability between heterogeneous JXTA applications. The main application area we have been focussing on is the P2P exchange of educational resources (using schemas like IEEE LOM, IMS, and ADL SCORM [2] to describe course materials), but other application areas are possible as well.

#### 1.4.3 Edutella query service

Having agreed on the basic infrastructure and the use of RDF and LOM metadata for annotating our resources, we still need a standardized query language to cope with the different solutions each peer may have found to structure and store its learning resources. The Edutella Query Service is a standardized query exchange mechanism for RDF metadata stored in distributed RDF repositories and serves both as query interface for individual RDF repositories located at single Edutella peers as well as

query interface for distributed queries spanning multiple RDF repositories (storing RDF statements based on arbitrary RDFS schemata).

One of the main purposes is to abstract from various possible RDF storage layer query languages (e.g., SQL) and from different user level query languages (e.g., RQL, TRIPLE): The Edutella Query Exchange Language and the Edutella Common Data Model provide the syntax and semantics for an overall standard query interface across heterogeneous peer repositories for any kind of RDF metadata. The Edutella network uses the query exchange language family RDF-QEL-i (based on Datalog semantics and subsets thereof) as standardized query exchange language format which is transmitted in an RDF/XML-format.

We will start with a simple RDF knowledge base and a simple query with the following RDF XML Serialization:

```
<lib:Book about='http://www.xyz.com/sw.html' >
  <dc:title>Software Engineering</dc:title>
</lib:Book>

<lib:Book about='http://www.xyz.com/ia.html' >
  <dc:title>Intelligent Agents</dc:title>
  <dc:subject
    rdf:resource='http://www.kbs.uni-hannover.de/Uli/ACM_CCS#I.2' />
</lib:Book>

<lib:Book about='http://www.xyz.com/ai.html' >
  <dc:title>Artificial Intelligence</dc:title>
</lib:Book>

<lib:AI-Book about='http://www.xyz.com/pl.html' >
  <dc:title>Prolog</dc:title>
</lib:AI-Book>
```

To simplify the query, we assume that the book on intelligent agents is annotated with the ACM CCS node I.2 ARTIFICIAL INTELLIGENCE. Otherwise, we could easily query for supertopics of the entry by using the `lom_cls:taxon` attribute and the fact that the ontology is also part of the knowledge base. We will show you a small example of this handling of subtopics later.

Edutella peers can be highly heterogeneous in terms of the functionality (i.e., services) they offer. A simple peer has RDF storage capability only. The peer has some kind of local storage for RDF triples (e.g., a relational database) as well as some kind of local query language (e.g., SQL). In addition the peer might offer more complex services such as annotation, mediation or mapping.

To enable the peer to participate in the Edutella network, Edutella wrappers are used to translate queries and results from the Edutella query and result exchange format to the local format of the peer and vice versa, and to connect the peer to the Edutella network by a JXTA-based P2P library. To handle queries, the wrapper uses the common Edutella query exchange format and data model for query and result

representation. For communication with the Edutella network, the wrapper translates the local data model into the Edutella Common Data Model ECDM and vice versa, and connects to the Edutella Network using the JXTA P2P primitives, transmitting the queries based on ECDM in RDF/XML form.

In order to handle different query capabilities, Edutella defines several RDF-QEL-i exchange language levels, describing which kind of queries a peer can handle (conjunctive queries, relational algebra, transitive closure, etc.) The same internal data model is used for all levels.

#### *Edutella Common Data Model (ECDM)*

The ECDM is based on Datalog, which is a well-known non-procedural query language based on Horn clauses without function symbols. A Datalog program can be expressed as a set of rules/implications (where each rule consists of one positive literal in the consequent of the rule (the head), and one or more negative literals in the antecedent of the rule (the body)), a set of facts (single positive literals) and the actual query literals (a rule without head, i.e., one or more negative literals). Literals are predicates expressions describing relations between any combination of variables and constants such as `title(http://www.xyz.com/book.html, 'Artificial Intelligence')`. Disjunction is expressed as a set of rules with identical head. Additionally, we can use negation as failure in the antecedent of a rule, with the semantics that such a literal cannot be proved from the knowledge base. A Datalog query then is a conjunction of query literals plus a possibly empty set of rules [25].

Datalog queries easily map to relations and relational query languages like relational algebra or SQL. In terms of relational algebra, Datalog is capable of expressing selection, union, join and projection and hence is a relationally complete query language. Additional features include transitive closure and other recursive definitions.

The example knowledge base in Datalog reads

```
title('http://www.xyz.com/ai.html', 'Artificial Intelligence').
type('http://www.xyz.com/ai.html', Book).
title('http://www.xyz.com/ia.html', 'Intelligent Agents')
subject('http://www.xyz.com/ia.html', 'http://www.kbs.uni-hannover.de/Uli/ACM_CCS#I.2').
type('http://www.xyz.com/ia.html', Book).
title('http://www.xyz.com/sw.html', 'Software Engineering').
type('http://www.xyz.com/sw.html', Book).
title('http://www.xyz.com/pl.html', 'Prolog').
type('http://www.xyz.com/pl.html', AI-Book).
```

Each RDF repository can be viewed as a set of ground assertions either using binary predicates as shown above, or as ternary statements “s(S,P,O)”, if we include the predicate as an additional argument. In the following examples, we use the binary surface representation.

We will have a closer look on the evaluating of the following query (plain English)

“Return all resources that are a book having the title “Artificial Intelligence” or have content, that is a subtopic of ”Artificial Intelligence” or that are an AI book.”

**Example Query in (binary) Datalog notation.**

```
aibook(X) :- title(X, 'Artificial Intelligence'),
             type(X, Book).
aibook(X) :- type(X, AI-Book).
aibook(X) :- subject(X, 'http://www.kbs.uni-hannover.de/Uli/ACM_CCS#I.2').

?- aibook(X).
```

Since our query is a disjunction of three (purely conjunctive) subqueries, its Datalog representation is composed of three rules with identical heads. The literals in the rules' bodies directly reflect RDF statements with their subjects being the variable X and their objects being bound to constant values such as 'Artificial Intelligence'. Literals used in the head of rules denote derived predicates (not necessarily binary ones). The query expression “aibook(X)” asks for all bindings of X, which conform to the given Datalog rules and our knowledge base, with the results:

```
aibook('http://www.xyz.com/ai.html')
aibook('http://www.xyz.com/ia.html')
aibook('http://www.xyz.com/pl.html')
```

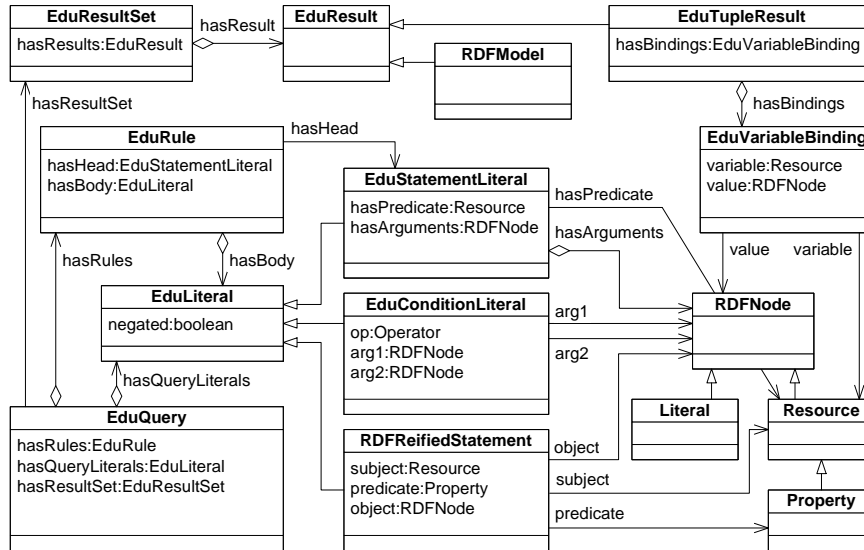
It is of course easy to extend the examples to subtopics of 'Artificial Intelligence', if our ontology is also a part of the knowledge base. We would then extend our query with the search for supertopics, by defining the new concept 'content' and then replacing the last subquery with:

```
aibook(X):- content(X,'http://www.kbs.uni-hannover.de/Uli/ACM_CCS#I.2').

content(X,Y):- subject(X,Y).
content(X,Y):- subject(X,Z),
                subtopic(Z,Y).
suptopic(Z,Y):- taxon(Y,Z).
suptopic(Z,Y):- taxon(Y,W),
                taxon(W,Z).
```

Internally Edutella Peers use a Datalog based model to represent queries and their results. Figure 1.1 visualizes this data model as UML class diagram. The Edutella Wrapper API includes the ECDM as well as wrappers for different query languages, and is available as source code from the Edutella Project Page: <http://edutella.jxta.org/>.

Our current prototype environment features a set of different peers to demonstrate various aspects of the translation from ECDM to local query languages. It contains



**Fig. 1.1.** Edutella Common Data Model (ECDM)

the QEL query exchange mechanism, a simple mediator and the wrapping of different repository peer types, including an OLR (Open Learning Repository) based peer [8] using a subset of IMS/LOM RDF metadata stored in a relational database, a DbXML-based peer [23] as a prototype for an XML repository using a simple mapping service to translate from RDF-QEL-1 queries (conjunctive queries) to Xpath queries over the appropriate XML-LOM schema, AMOS-II-based peers [24] with local repositories, KAON-based peers [19] allowing remote annotation [14] using an RDF-based ontology format, and an O-Telos-Peer [16] and [30].

## 1.5 Acknowledgements

The basic versions of the OLR were created and implemented by Boris Wolf. We also gratefully acknowledge important input and discussion on the design and use of the OLR from Hadhami Dhraief. The Edutella project has had numerous contributors, too many to mention here, but we refer to the various Edutella publications written together with them. We gratefully acknowledge all their input and work on these topics. We want to especially thank Wolf Siberski, Martin Wolpers, Changtao Qu, Steffen Staab, Christoph Schmitz and Bernd Simon for recent input to our work.

## Bibliography

### References

1. *The ACM Computing Classification System–1998 Version*  
valid in 2002, <http://www.acm.org/class/1998/>
2. Technical Team ADLSCORM *Specification VI.2*  
<http://www.adlnet.org/Scorm/scorm.cfm>
3. H. Allert, H. Dhraief, W. Nejdl. *How are Learning Objects Used in Learning Process?*  
ED-MEDIA 2002, World Conference on Educational Multimedia, Hypermedia & Telecommunications, Denver Colorado, United States, June 24-29, 2002
4. J. Brase *AIOnt – An extension of the ACM CCS for the field of Artificial Intelligence, based on a paper by D.L. Waltz*  
<http://www.kbs.uni-hannover.de/Uli/AI-Ontologie.rdf>
5. J. Brase *mySWEBOK – A classification for Software engineering, based on the SWEBOK*  
[http://www.kbs.uni-hannover.de/Uli/SWT\\_Ontologie.rdf](http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf)
6. J. Brase *ACMCCS – The ACM Classification for Computer Science as a rdf-file*  
[http://www.kbs.uni-hannover.de/Uli/SWT\\_Ontologie.rdf](http://www.kbs.uni-hannover.de/Uli/SWT_Ontologie.rdf)
7. D. Brickley and R. V. Guha *W3C Resource Description Framework (RDF) Schema Specification*  
<http://www.w3.org/TR/1998/WD-rdf-schema>
8. H. Dhraief, W. Nejdl, B. Wolf and M. Wolpers *Open Learning Repositories and Metadata Modeling*  
International Semantic Web Working Symposium (SWWS), jul 2001, Stanford, CA
9. The Dublin Core Metadata Initiative  
<http://dublincore.org/>
10. R. Dornfest and D. Brickley *The Power of Metadata*  
excerpted from the book "Peer-to-Peer: Harnessing the Power of Disruptive Technologies, 2001  
<http://www.openp2p.com/pub/a/p2p/2001/01/18/metadata.html>
11. *The Edutella Project*  
<http://edutella.jxta.org>
12. S. Guth, G. Neumann, B. Simon *UNIVERSAL - Design Spaces of Learning Media*  
Proceedings of the 34th Hawaii International Conference on System Sciences, Maui (USA), January, 2001
13. L. Gong *Project JXTA: A Technology Overview*  
<http://www.jxta.org/project/www/docs/TechOverview.pdf>
14. S. Handschuh, S. Staab and A. Maedche *CREAM - Creating relational metadata with a component-based, ontology-driven annotation framework*  
Workshop on Knowledge Markup and Semantic Annotation at the First International Conference on Knowledge Capture (K-CAP'2001), Victoria, BC, Canada
15. *IMS Learning Resource Metadata Specification VI.2.1*  
<http://www.imspjct.org/metadata/index.html>
16. M. Jarke, R. Gallersdörfer, M. Jeusfeld, M. Staudt and S. Eherer "ConceptBase - a deductive object base for meta data management  
Journal on Intelligent Information Systems 1995, 4(2): 167 -192
17. O. Lassila and R. R. Swick *W3C Resource Description Framework (RDF) Model and Syntax Specification*  
<http://www.w3.org/TR/REC-rdf-syntax>

18. Learning Technology Standards Committee of the IEEE: *Draft Standard for Learning Objects Metadata IEEE P1484.12.1/D6.412*. June 2002).  
[http://ltsc.ieee.org/doc/wg12/LOM\\_1484\\_12\\_1\\_v1\\_Final\\_Draft.pdf/](http://ltsc.ieee.org/doc/wg12/LOM_1484_12_1_v1_Final_Draft.pdf/)
19. A. Mädche, S. Staab, R. Studer, Y. Sure and R. Volz *SEAL — Tying Up Information Integration and Web Site Management by Ontologies*  
IEEE Data Engineering Bulletin Mar 2002  
<http://www.research.microsoft.com/research/db/debull>
20. W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér and T. Risch *EDUTELLA: a P2P Networking Infrastructure based on RDF*  
11th International World Wide Web Conference Hawaii, USA, May 2002  
<http://edutella.jxta.org/reports/edutella-whitepaper.pdf>
21. W. Nejdl, B. Wolf, S. Staab and J. Tane *EDUTELLA: Searching and Annotating Resources within an RDF-based P2P Network*  
Semantic Web Workshop, 11th International World Wide Web Conference, may 2002, Honolulu, Hawaii, USA
22. M. Nilsson. *IMS Metadata RDF binding guide* May 2001  
<http://kmr.nada.kth.se/el/ims/metadata.html>
23. C. Qu and W. Nejdl *Towards Interoperability and Reusability of Learning Resources: A SCORM-conformant Courseware for Computer Science Education*  
Proc. of 2nd IEEE International Conference on Advanced Learning Technologies (IEEE ICALT 2002), Kaza, Russia, September 2002.
24. T. Risch and V. Josifovski *Distributed Data Integration by Object-Oriented Mediator Servers*  
Concurrency and Computation: Practice and Experience 2001 13(11):933 - 953
25. A. Silberschatz, H. F. Korth and S. Sudarshan *Database Systems Concepts*  
McGraw-Hill Higher Education 2001 4
26. *The guide to the Software engineering body of Knowledge*  
<http://www.swebok.org>
27. *The ULI-project homepage*  
<http://www.uli-campus.de>
28. D. Waltz *Scientific datalink's artificial intelligence classification scheme*  
The AI Magazine 1985, 6(1):58-63.
29. B. Wolf, H. Dhraief, M. Wolpers, W. Nejdl. *Open Learning Repositories and Metadata Modeling*  
International Semantic Web Working Symposium (SWWS) Stanford University, California, USA, July 30 - August 1, 2001
30. M. Wolpers, W. Nejdl and I. Brunkhorst *Using an O-Telos Peer to Provide Reasoning Capabilities in an RDF-based P2P-Environment*  
Proceedings of the International Workshop on Agents and Peer-to-Peer Computing, Bologna, Italy, July, 2002