Integrating Adaptive Hypermedia Techniques and Open RDF-Based Environments

Peter Dolog, Rita Gavriloaie, Wolfgang Nejdl[*], Jan Brase[†]

ABSTRACT

The World Wide Web has not only revolutionized the area of traditional hypermedia, it is also starting to influence adaptive hypermedia research. The main feature of the World Wide Web has been simplicity, standards based protocols and formats, and open environments and systems. In this paper we will investigate the potential of these aspects for adaptive hypermedia systems, based on a discussion of existing standards useful for adaptive hypermedia and on experiences we have gained through an RDF-based P2P infrastructure we have been developing in the context of several projects.

Categories and Subject Descriptors

H.5.4 [**Information Interfaces and Presentation**]: Hypertext/Hypermedia—*architectures, navigation, user issues*; H.3.3 [**Information storage and retrieval**]: Information Search and Retrieval—*query formulation*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*distributed applications, distributed databases*

General Terms

Design, Standardization

Keywords

Personalized Queries, RDF-based P2P Networks, Adaptation, User Modeling Standards, Ontologies

1. INTRODUCTION

Hypermedia applications are established technology for delivering information and learning materials. The basic idea of the hypermedia / hypertext paradigm is that information is interconnected by links, and different information items can be accessed by navigating through this link structure. The World Wide Web, by implementing this basic paradigm in a simple and efficient manner, has made this model the standard way for information access on the Internet. Additionally, various adaptive hypermedia functionalities have been investigated to allow personalized access and display of information and learning materials. They can be divided into adaptive content presentation techniques and into adaptive navigation techniques [10, 11]. Techniques, which adapt the content that is displayed include conditional inclusion/removing/dimming of fragments (text, figures, etc.), stretchtext, and sorting of fragments, based on user knowledge, interests and goals or environment abilities. Navigation support by means of link adaptation helps user to chose the best way from currently presented information to other information, according to needs, level of knowledge, interests, preferences etc. Techniques include adaptive link sorting, adaptive link hiding, direct guidance, adaptive link annotation and generation, and adaptive navigation maps.

Obviously, the knowledge about a user is very important in such adaptive systems. The systems like AHA! [6], ELM-ART [27], SQL Tutor [23], or KBS-Hyperbook [14] have introduced personalized navigation and content presentation based on the knowledge the user has. SQL Tutor also adapts navigation and presentation according to the history of previous sessions (problems solved, etc.)

In an open environment like the World Wide Web such navigational hints and personalization features would arguable be even more useful. To do this, however, we have to adapt adaptation functionalities from the closed architectures of conventional systems to open environment, and we have to investigate the possibilities of providing additional metadata in this open environment as additional input for these adaptation functionalities.

Hypermedia systems have historically been based on conventional client/server architectures. Recently open hypermedia approaches have tried to accommodate distributed content, but still use a central server and central data structures to integrate and serve distributed content. Peer-to-Peer infrastructures go a step further, and both allow the provision of distributed services as well as building upon open standards to describe distributed content in the WWW environment.

In the context of several projects we have recently investigated how to enable access to distributed information in the Internet, building on peer-to-peer infrastructures and semantic web technologies (see e.g. [24]). We have discussed different kinds of peers and functionalities and have introduced a query exchange language to query for information in such a peer-to-peer network [25]. In this network, the Edutella network, information is distributed over and provided by independent peers, who can interact and interchange information with others. Both data and metadata (described in RDF) can be distributed in an arbitrary manner. We have been focussing on managing educational material in this network, but all other kinds of resources can be managed as well.

Such open environments can benefit from knowledge and experiences taken from hypermedia and adaptive hypermedia field. As it is in conventional hypermedia systems, we want to adapt information and navigation based on the knowledge about users. How-

[*]Learning Lab Lower Saxony, University of Hannover, Expo Plaza 1 D-30539 Hannover, Germany, {dolog,gavriloaie,nejdl}@learninglab.de

[†]Information Systems Institute, University of Hannover, Appelstrasse 4 D-30167 Hannover, brase@kbs.uni-hannover.de

WWW2003, May 20–24, 2003, Budapest, Hungary.
ACM 1-58113-680-3/03/0005.xxx.

ever, these descriptions have to be standardized and also can be distributed in the network, similarly to descriptions of content and descriptions of other aspects of the information. Figure 1 depicts an example of storing different kinds of metadata at different peers in the network.

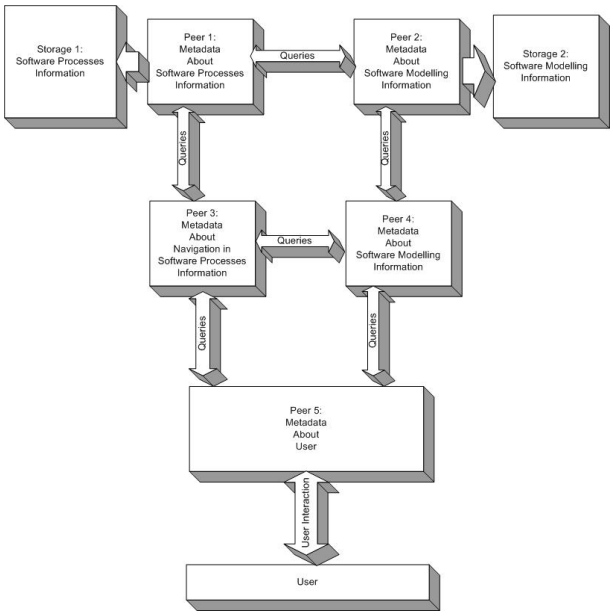

Figure 1: Hypermedia and P2P approach.

In this paper we discuss the possibilities for using standardized metadata to describe and classify information stored in an RDF-based P2P network and to describe knowledge, preferences and experiences of users accessing this information. Furthermore, we will investigate how to implement adaptive functionalities in such a P2P network, with the final goal of implementing personalized access to distributed learning materials providing a smart learning space [20] for the learner.

The rest of the paper is structured as follows. In section 2 we describe the metadata standards and their usage in our case. The section is divided into two parts. In the first part we discuss metadata for educational resources. The second part is devoted to metadata for describing a user. In section 3 we describe how metadata can be accessed in P2P environment by queries and how several adaptive hypermedia techniques can be formulated as queries over the network. The architecture and interface of our system is explained in section 4. In section 5 we relate our approach to other work. Finally, we draw some conclusions and proposals for further work in section 6.

2. USING METADATA STANDARDS

2.1 Describing Educational Resources

2.1.1 From Dublin Core to LOM

Let us first take a look on the idea of using specific schemas and vocabularies to describe digital resources. One of the most common metadata schemas on the web today is the "Dublin Core Schema" (DC) by the DCMI. The Dublin Core Metadata Initiative (DCMI) 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 for example *dc:creator = nejdl*.

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 [1]. Since Dublin Core is designed for metadata describing 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. A learning object in this context is any entity, digital or non-digital that may be used for learning, education or training. LOM Schema defines a structure divided into 9 categories: *general, life cycle, meta-metadata, technical, educational, rights, relation, classification, annotation*.

Each category represents a group of data elements that in turn can contain subelements. All data elements are optional.

Work on the LOM schema has started in 1998, the current version is 6.4. LOM became an official standard in June 2002. 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 chapters 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.

2.1.2 Using RDF Metadata

We will use examples from the ULI project (http://www.uli-campus.de/), which is a cooperative project between 11 universities, sponsored by the German Federal Department of Education and Research. ULI offers a full study program consisting of distributed computer science courses covering a wide range of computer science topics, providing both, multimedia content as well as accompanying distance learning services.

As a test and application area, ULI provides a large pool of rather homogeneous educational resources. Although the courses are provided by different universities, they cover the same field, are integrated into the same study program, and are all being taught by computer science professors, therefore providing a good testbed for personalization.

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 (see table below). 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 in the next section.

1. General	1.2 Title	**dc:title**
	1.3 Language	**dc:language**
	1.4 Description	**dc:description**
2.Lifecycle	2.3 Contribute	**dc:creator** with a **lom:entity** and the author in vCard format "name surname" dcq:created with the date in W3C format
6.Rights	6.3 Description	**dc:rights**
7. Relation		**dcq:hasFormat** **dcq:isFormatOf** **dcq:hasPart** **dcq:isPartOf** **dcq:hasVersion** **dcq:isVersionOf** **dcq:requires** **dcq:isRequiredBy**
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)

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.
Example:

```
<dc:title> Techniques for Multimedia WS 2001
(Mannheim) Part 2a from 17.10.2001:
Compression 1 </dc:title>
```

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

```
<dc:description > Internet history,
internet technology, IP, DNS, Routing,
TCP, IP and ATM1</dc:description>
```

dc:creator. The creator of the resource will be displayed as a part of a lom:entity in the vCard Format "first name name". 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.
Example:

```
<dc:creator>
   <lom:entity>
      <vCard:FN>Wolfgang Nejdl</vCard:FN>
   </lom:entity>
</dc:creator>
```

dcq:created. Specifies when the course was created using the W3C format. Usually only one appearance in the definition of the course.
Example:

```
<dcq:created>
   <dcq:W3CDTF>
      <rdf:value>2001-09-15</rdf:value>
   </dcq:W3CDTF>
</dcq:created>
```

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 criterion 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*.

2.1.3 Topic Ontologies for Content Classification

Personalized access means that resources are tailored according to some relevant aspects of the user. Which aspects of the user are important or not depends on the personalization domain. For educational scenarios it is important to take into account aspects like whether the user is student or a teacher, whether he wants to obtain a certain qualification, has specific preferences, and, of course, which is his knowledge level for the topics covered in the course.

Preferences about learning materials can be easily exploited, especially if they coincide directly with the metadata and metadata values used. For users preferring Powerpoint presentations for example, we can add the literal dc:format(Resource, powerpoint) to queries searching appropriate learning materials.

Taking user knowledge about topics covered in the course into account is more tricky. The general idea is that we annotate each document by the topics covered in this document. Topics can be covered by sets of documents, and we will assume that a user fully knows a topic if he understands all documents annotated with this topic. However, though the standards we have just explored only provide one attribute (dc:subject) for annotating resources with topics, in reality we might want to have different kinds of annotations, to distinguish between just mentioning a topic, introducing a topic, and covering a topic. In the following we will simply assume that dc:subject is used for "covered" topics, but additional properties for these annotations might be useful in other contexts.

Additionally, it is obvious that self-defined keywords cannot be used in our context, as we have to use a controlled vocabulary / an ontology for annotating documents and describing user knowledge (see also [16]). Defining a private ontology for a specific field unfortunately works only in the closed microworld of a single university. To be more general, we therefore decided to use ontologies which are already part of internationally accepted classification systems.

ACM CCS as a topic ontology for learning objects. The ACM Computer Classification system ([2]) 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.1.0 General
    - D.1.1 Applicative (Functional) Programming
    - D.1.2 Automatic Programming
    - D.1.3 Concurrent Programming
    - D.1.4 Sequential Programming
    - D.1.5 Object-oriented Programming
    - D.1.6 Logic Programming
    - D.1.7 Visual Programming
    - D.1.m Miscellaneous
  - 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
- 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: D.1.3.1 for the keyword *Parallel programming* that is accessible via the taxon path: *Software(D)/PROGRAMMING TECHNIQUES(D.1)/Concurrent Programming(D.1.3)*.

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. These techniques are discussed in more detail in [8].

To classify a resource, the IEEE Learning Object RDF Binding Guide (Draft Version) [26] 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*. As discussed, we used ACM CCS, the appropriate RDF files can be found at *http://www.kbs.uni-hannover.de/Uli/ACM_CCS.rdf*. The main structure is as follows:

```
<dcq:SubjectScheme rdf:ID= ACM CCS >
<rdfs:label>ACM Computer Classification system
</rdfs:label>
</dcq:SubjectScheme>
<lom_cls:Taxonomy>
   <lom_cls:rootTaxon>
   <ACM:ACM CCS rdf:about= http://www.kbs.
   uni-hannover.de/Uli/ ACM CCS.rdf#D >
   <rdf:value>Software</rdf:value>
       <lom_cls:taxon>
       <ACM:ACM CCS rdf:about= http://www.kbs.
       uni-hannover.de/Uli/ACM CCS.rdf#D.1>
       <rdf:value>PROGRAMMING TECHNIQUES
       </rdf:value>
          <lom_cls:taxon>
          <ACM:ACM CCS rdf:about= http://www.kbs.
          uni-hannover.de/Uli/ACM CCS.rdf #D.1.6>
          <rdf:value>Logic Programming
          </rdf:value>
          </lom_cls:taxon>
       </lom_cls:taxon>
   </lom_cls:rootTaxon>
</lom_cls:Taxonomy>
```

This subset of the ontology shows the definition of one main node of the ACM CCS (Software), refined in this example into one subtopic (PROGRAMMING TECHNIQUES), which is itself refined into a subtopic (Logic Programming).

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/AI/ OLR/S1T1.pdf >

<dc:subject rdf:resource= http://www.kbs.
uni-hannover.de/Uli/ ACM CCS.rdf#D.1.6 / >

</rdf:Description>
```

2.2 Describing Users

In recent years there have been some efforts to standardize the information about a user, which should be maintained by a system. The two most important examples for such standards are PAPI [17] and IMS LIP [19]. Both standards deal with several categories for information about a user. Figure 2 depicts the conceptual view of PAPI profile (figure taken from [17]).

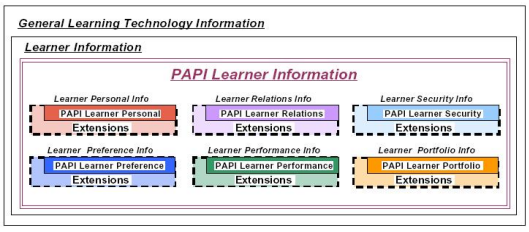

Figure 2: The core categories in PAPI profile.

PAPI distinguishes *personal*, *relations*, *security*, *preference*, *performance*, and *portfolio* information. The *personal* category contains information about names, contacts and addresses of a user.

Relations category serves as a category for specifying relationships between users (e.g. classmate, teacheris, teacherof, instructoris, instructorof, belongsto, belongswith). *Security* aims to provide slots for credentials and access rights. *Preference* indicates the types of devices and objects, which the user is able to recognize. *Performance* is for storing information about measured performance of a user through learning material (i.e. what does a user know). *Portfolio* is for accessing previous experience of a user. Each category can be extended.

Similarly IMS LIPS standard contain several categories for data about a user. The categories are depicted in the fig. 3 (figure taken from [19]).

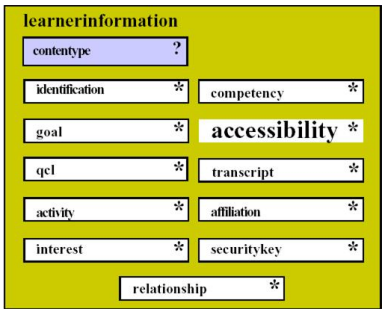

Figure 3: The core categories for IMS learner information package learner data.

The *identification* category represents demographic and biographic data about the user. The *goal* category represents learning, career and other objectives of the learner. The *QCL* category is used for identification of qualifications, certifications, and licenses from recognized authorities. The *activity* category can contain any learning related activity in any state of completion. The *interest* category can be any information describing hobbies and recreational activities. The *relationship* category aims for relationships between core data elements. The *competency* category serves as slot for skills, experience and knowledge acquired. The *accessibility* category aims for general accessibility to learner information by means of language capabilities, disabilities, eligibility, and learning preferences. The *transcript* category represents institutional-based summary of academic achievements. The *affiliation* category represents information records about membership of professional organizations. The *security key* is for set passwords and keys assigned to a learner.

An example of accessibility category data is depicted in fig. 4.

2.2.1 User Profiles in RDF

In general we have three general possibilities how to handle adaptation and user profiles in an open environment: *overlay model based on resources*, *overlay model based on ontology*, *overlay or stereotype model based on ontology and resource description*.

Overlay model based on resources. The first possibility is to use an overlay model based on resources. This means that user knowledge is measured according to the resources which have been read / visited etc. The user model is maintained at the user site. The problem with this approach in our open environment is that we can have several resources, which belong to one topic or concept from a vocabulary.

Overlay model based on an ontology. The second approach is similar to the first one, but the level of knowledge is handled according to the conceptual model or ontology. This means that we can precisely identify what resources should not be displayed because they belong to topics already understood by a user. The user model is maintained at the user site. The problem here is to map different topologies and to measure distance between topics in one or more ontologies.

```
<learnerinformation>
  <contentype><referential>
      <sourcedid><source>
          IMS_LIP_V1p0_Example
        </source>
        <id>basic_1001</id>
      </sourcedid></referential>
  </contentype>
  <accessibility>
    <contentype><referential>
        <indexid>accessibility_01</indexid>
    </referential></contentype>
    <language><typename>
        <tysource sourcetype="imsdefault"/>
    <tyvalue>German</tyvalue></typename>
      <contentype><referential>
          <indexid>
            language_01
          </indexid>
      </referential></contentype>
      <proficiency profmode="OralSpeak">
        Excellent
      </proficiency>
      <proficiency profmode="OralComp">
        Excellent
      </proficiency>
      <proficiency profmode="Read">
        Good
      </proficiency>
      <proficiency profmode="Write">
        Poor
      </proficiency>
    </language></accessibility>
</learnerinformation>
```

Figure 4: An example of LIP Accessibility information.

Overlay or stereotype model based on ontology and resource description. The third approach is based on the idea that resources and constraints for their use should be stored together as closely as possible. This means that we state directly in the resource description, for which group of students, which level of knowledge or for from which specific domain the resource is appropriate. The resource then contains descriptions like:

"I (resource xyz) am intended to be used by users, who are studying computer science, telecommunications, or physics and have knowledge about the topic I cover greater than average and who are interested in this topic". The user model is maintained at the user site but is matched directly with the resource descriptions.

Additional attributes from LOM can be useful as well:

- `<lom-edu:intendedEndUserRole rdf:resource="&lom-edu;Manager"/>`
- `<lom-edu:context rdf:resource="&lom-edu;School"/>`

We can also use `dcterms:audiencelevel` and the `lom:AgeRange` for focussing on specific audiences:

```
<dcterms:audience>
 <lom:AgeRange>
  <rdf:value>7-12</rdf:value>
 </lom:AgeRange>
</dcterms:audience>
```

and for preference

```
<lom-edu:language>
 <dcterms:RFC1766>
  <rdf:value>en<rdf:value>
 </dcterms:RFC1766>
</lom-edu:language>
```

To specify the required level of knowledge we could introduce a new category to LOM (adaptation category for example). Another possibility is to use a relation category and the properties of PAPI or IMS. The second case makes it easier to query for appropriate resources, because we can directly map and compare what we have in the user profile and what we have in the resource description. It also means that we need to classify the learning resource according to the user profile required for accessing this learning object. LOM provides the classification category with the *purpose* element to do this. The *purpose* element has several subelements: *prerequisite*, *educational*, *objective*, *accessibility restrictions*, *educational level*, *skill level*, *security level*, or *competency*. We decided to use the *accessibility restriction* subelement, in order to define constraints for accessing the learning object.

```
<lom-cls:accessibilityRestrictions
 rdf:resource="http://learninglab.de/papi#"/>
  <papi:performance>
    <rdf:Description rdf:ID="performance_1">
     <papi:learning_competency rdf:resource=
    "http://www.kbs.uni-hannover.de/Uli/
    ACM_CCS.rdf/#D.1"/>
    <papi:granularity>topic</papi:granularity>
    <papi:performance_coding>number
    </papi:performance_coding>
    <papi:performance_metric>0-1
    </papi:performance_metric>
    <papi:performance_value>greater_than(0.5)
    </papi:performance_value>
    <papi:performance_bucket>
      <rdf:Description
      rdf:ID="performance_bucket_1">
       <papi:performance_bucket_name>
       time_on_task
       </papi:performance_bucket_name>
       <papi:performance_bucket_value>10min
       </papi:performance_bucket_value>
      </rdf:Description>
    </papi:performance_bucket>
   </rdf:Description>
  </papi:performance>
</lom-cls:accessibilityRestrictions>
```

Figure 5: Accessibility constraints for a specific resource.

Directly using these user model fields (PAPI) allows us to directly search for resources, which conform to the user profile. For example, the resource with the restricted access specified in fig. 5 is intended for a user, whose level of knowledge about the *programming techniques* topic from ACM CCS is greater than 0.5. The example of a user model, which satisfies the constraint / access restriction described above is depicted in fig. 6.

Figure 6 depicts an example of a user model, where the user knows about *programming techniques* at the level of 0.6. This level of knowledge has been derived from an appropriate annotation for the (already read) `S5T1.pdf` resource and evaluated by the test `Test_S5T1`. For the topic we use the competence field from the PAPI profile. To indicate the level of knowledge, we use *granularity* (i.e. we measure the level of knowledge for each topic), *performance coding* (in numbers), *performance metric* (from 0 to 1) and *performance value* (0.6). We also use *bucket* to specify the time, which was required for performing the test.

```
<rdf:RDF
 xmlns:rdf=
 "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs=
 "http://www.w3.org/2000/01/rdf-schema#"
 xmlns:papi=
 "http://learninglab.de/papi#">
<rdf:Description rdf:ID="student1">
  <papi:performance>
    <rdf:Description rdf:ID="performance_1">
      <papi:issued_from_identifier rdf:resource=
      "http://www.kbs.../Test_S5T1"/>
    <papi:learning_competency rdf:resource=
    "http://www.kbs.uni-hannover.de/Uli/
    ACM_CCS.rdf/#D.1"/>
    <papi:learning_experience_identifier
    rdf:resource="http://www.kbs.../S5T1.pdf"/>
    <papi:granularity>topic</papi:granularity>
    <papi:performance_coding>number
    </papi:performance_coding>
    <papi:performance_metric>0-1
    </papi:performance_metric>
    <papi:performance_value>0.6
    </papi:performance_value>
    <papi:performance_bucket>
      <rdf:Description
      rdf:ID="performance_bucket_1">
       <papi:performance_bucket_name>
       time_on_task
       </papi:performance_bucket_name>
       <papi:performance_bucket_value>10min
       </papi:performance_bucket_value>
      </rdf:Description>
    </papi:performance_bucket>
   </rdf:Description>
  </papi:performance>
</rdf:Description>
</rdf:RDF>
```

Figure 6: User profile performance and record.

We can use several other categories of PAPI for example *preference* and *identification*, a few other fields are still unclear.

The RDF graph for a performance record in our system is depicted in the fig 7.

3. QUERYING FOR APPROPRIATE RESOURCES

3.1 Queries in an RDF-based P2P Network

P2P applications have proven very successful for special cases like exchanging music files. Unfortunately, so far they have been restricted to application specific metadata schemas and simple keyword-based searches, which is both insufficient for our purposes. Retrieving a song like “Material Girl from Madonna” does not need complex query languages nor complex metadata, but looking for an “Advanced Course in Algebra for Computer Science Students” will only be successful with more complex metadata (such as those described above) and also needs more advanced query capabilities.

The Edutella project [24, 25] (http://edutella.jxta.org) addresses these shortcomings of current P2P applications by building on the W3C metadata standard RDF [22, 9], and is based on the assumption that all resources managed within the network are described by RDF metadata. The Edutella Query Service is a standardized query exchange mechanism for RDF metadata stored in distributed RDF

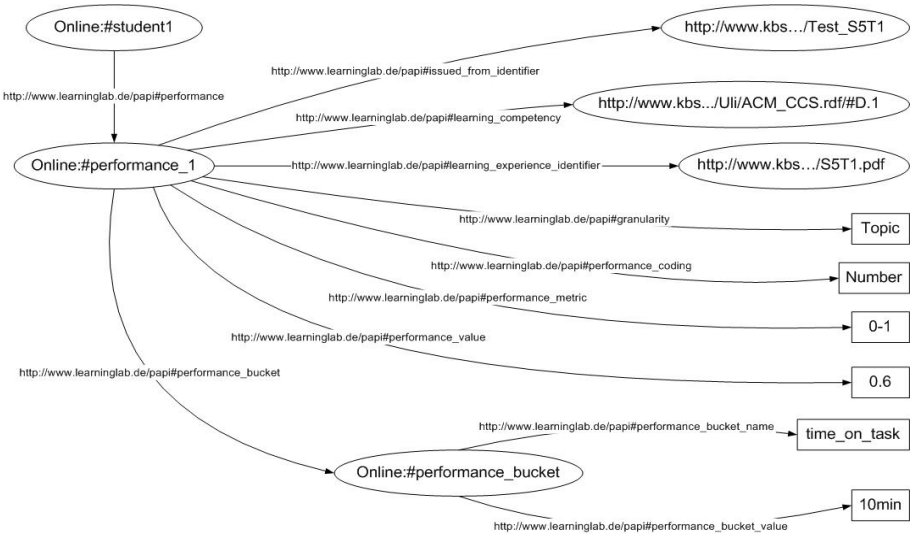

Figure 7: Example of RDF graph for user profile.

repositories and using arbitrary RDFS schemata.

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 [13]. 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. The ECDM is based on Datalog, which is a well-known non-procedural query language based on Horn clauses without function symbols. Datalog queries easily map to relations and relational query languages like relational algebra or SQL or to logic programming languages like Prolog. 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.

3.2 Adaptive Functionalities as Queries

Based on the RDF metadata managed within the Edutella network, we can now cast adaptive functionalities as RDF-QEL / Datalog queries over these resources, which are then distributed through the network to retrieve the appropriate learning resources. Personalization queries are then sent not only to the local repository, but to the entire Edutella network. In the following we use first order logic as well as Prolog notation to express these queries.[1] In our examples, we represent RDF statements, which are triples consisting of subject (S), predicate(P) and object (O), as binary Prolog predicates, where P represents the functor of the predicate, S and O the first and the second arguments (which is sufficient for all queries where the predicate is instantiated).

In our example we use a navigation map (see 4.2 for the user interface). On this navigation map we can realize several adaptive navigation techniques. The navigation map is a query over P2P network, visualized as a tree structure. The query comprises two parts — querying the metadata and querying the user profile.

In this way we can implement several adaptive hypermedia techniques. *Link annotation* [11] is implemented by an `annotate(+Page, +User, -Color)` predicate. We use a traffic light metaphor to express the suitability of the resources for the user, taking into account the user profile. Links are annotated with icons having different colors. A red icon marks a document that is too difficult, a green icon represents a document that is recommended for reading and a gray icon is used for a document that is considered to be already understood by the user.

The example for recommended page annotation is

```
annotate(Page, User, green) :-
recommended(Page, User), !.
```

We use the following criterion to determine a "recommended document": "A document is recommended for the user if it has not been understood yet and if all its prerequisites have already been understood."

```
recommended(Page, User) <--
not_understood_page(Page, User),
prerequisites(Page, Prereq),
forall P in Prereq understood_page(P,User).
```

The criterion above and a query asking for recommended pages expressed in Prolog:

```
recommended(Page, User) :-
    not_understood_page(Page, User),
    prerequisites(Page, Prereq),
    not (member(P, Prereq),
```

[1]Datalog queries are a subset of Prolog programs and of predicate logic.

```
    not_understood_page(P, User)).

?- recommended (Page, user)
```

Similarly we can support *adaptive link hiding*. The difference is only that instead of color annotation we display the appropriate links or we hide the inappropriate ones.

```
display(Page, User) :-
    topic(Page, Topic),
    knows(User, Topic, Rating),
    Rating > 0,5.

topic(Page, Topic) :-
    findall(X, dc_subject(Page, X), Topic).

?- display (Page, user)
```

This predicate can be used for example before the `annotate` predicate and says that the link is visible when the user level of knowledge is greater than 0,5. Then `annotate` predicate is used just for visible links. It means that our navigation map also supports *adaptive navigation map* technique.

Direct guidance can be derived from our prerequisite relationships. It means that the resources are sorted according to the prerequisite relationships graph. Another example of direct guidance is the next, initial and end arrow symbol in the map, which we used in another testing environment [12] (see fig. 8).

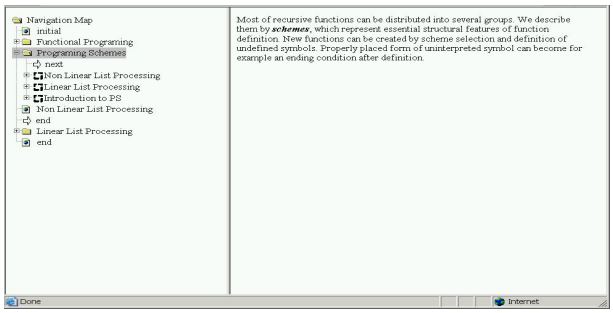

Figure 8: Example of the direct guidance.

Adaptive link generation techniques can be used in our approach for generating an alternative resource in the navigation map for currently recommended resource. So we can have a predicate, which finds all resources relevant to the resource according to the `dc_subject`. *Adaptive link sorting* can be used for ordering the resources in the map according to the interest of a user, taking prerequisite relationships into account.

So far we experimented only with packaged resources (Powerpoint presentations, PDF files, HTML pages). If we allow resources, which are compositions from several learning objects, we could also experiment with adaptive content presentation techniques. We could for example have resources structured as XML documents and use anchors and XPath expressions to identify and display / hide / dim fragments. This is an area of future research.

4. SYSTEM DESCRIPTION

4.1 Architectural View

While most of our Edutella infrastructure is written in Java, we have used MINERVA (a commercial ISO-Prolog compiler and interpreter written in Java) for our prototype implementing adaptive functionalities in the Edutella context. This was motivated by several reasons. First, RDF-QEL queries in Edutella are based on Datalog semantics and are therefore easily parsed and mapped into Prolog statements. Second, as Datalog is a powerful query language, it becomes easier to express various adaptive functionalities in our system. Third, because MINERVA is implemented in Java, it is easy to interface MINERVA with the rest of the Edutella infrastructure, and also with the usual Java user interface methods for running our peer as a browser applet (including all adaptive and interface functionalities).

From the architectural point of view our system consists of several interconnected parts, see Figure 9.

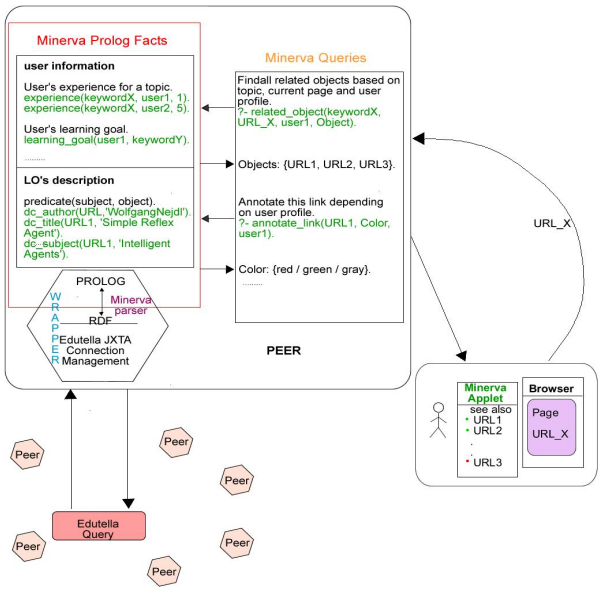

Figure 9: Architectural view of the system.

Minerva Prolog facts are stored in a main memory database. Each set of Prolog facts represents RDF metadata. Information contained in the database can be differentiated into two types:

- Metadata about the user
- Metadata about learning objects

The *user profile / information* describes different aspects of the user that are relevant for providing personalized access. Currently, the adaptive functionalities in our system are based on user's knowledge, but this user model can be easily extended with other user characteristics. The current prototype focuses on topics and user knowledge ratings (continuous values between 0 and 1) for these topics. User interactions with the system are recorded as set of visited resources and update the user model dynamically. User profiles could also be exported as RDF data.

Learning objects' descriptions provide information about the learning materials (lectures, etc) and about the relations between them. For content classification we use ontologies as described previously, which are also stored in the Prolog database (parsed from the appropriate RDF representation). Conceptual links between resources are expressed as sets of prerequisite statements.

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.

One part of the wrapper is the *MINERVA RDF parser* which translates RDF statements into Prolog predicates. RDF-QEL queries from the Edutella query are transformed into Prolog programs. The connection to the Edutella network is done using existing Edutella/JXTA libraries in Java and uses the socket interface. Adaptive functionalities are expressed as *Prolog queries*, which can also be shipped as RDF-QEL queries through the Edutella network.

The application is running locally, in the client's browser, embedded as an applet into a normal HTML file. The user can access the application from anywhere, independent of the platform that is used. The only thing needed is a Java enabled Internet browser.

4.2 Interface

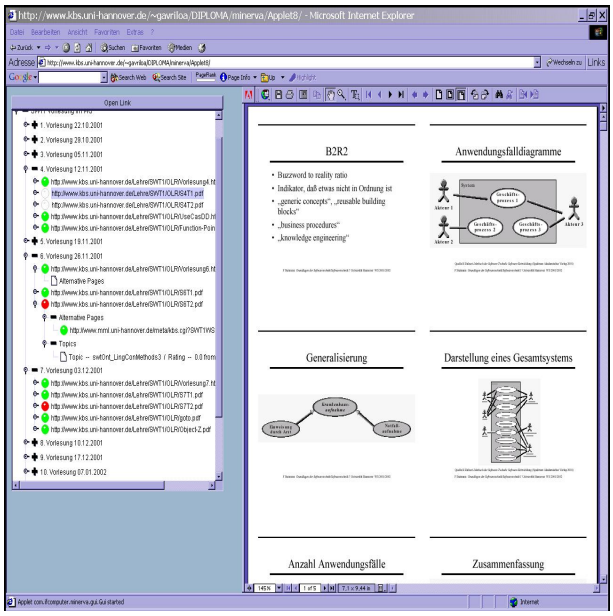

Figure 10: The user interface.

The information is presented in two different frames. The left-hand frame displayed the course structure according to metadata and prerequisite requirements. The user can navigate this structure and can open documents in the right-hand frame. Each resource is annotated according to the current user profile to express its suitability for the user. For annotations we use a traffic light metaphor.

The user can easily customize the adaptive functionality of the system. Currently the system supports two different methods to evaluate the user's knowledge and to generate the annotations accordingly. The user can switch between them from the graphical interface.

The navigational structure is very explicit and easy to manipulate. We support representation of alternative pages that is resources that are semantically the same but have different representation formats. Next to each resource we show the topics contained by that resource and also the user's current rating for each topic.

5. RELATED WORK

A very interesting paper in this context is the one by Bailey et al. [4], which aims to describe adaptive hypermedia techniques in an open hypermedia environment. [4] relates basic fundamental open hypermedia model concepts with adaptive hypermedia techniques. These basic concepts of open hypermedia models are data objects, context objects, behavior objects, concepts, levels of detail, links, and tours.

Data objects represent information items, context objects are associated to data objects and state in which context items are visible. Behavior objects are associated to data objects and include actions, which are performed, whenever some event on the data object occurs. Links, concepts, levels of detail, and tours represent different association over objects. These concepts are used within the contextual link server, which groups and references the resources in the web by means of these concepts.

In our work we have built upon yet a more open environment, where data objects are resources, and managed somewhere in the P2P network. These resources are annotated by RDF metadata representing the different kinds of attributes described [4], but as general metadata instead of as specific kinds of objects. Resources also can have associated accessibility restrictions, which are visibility constraints — contexts — from a user point of view. Because we provide a more expressive language for specifying contexts — Datalog based queries and constraints — we can have more complex rules for specifying accessibility in general, not only visibility constraints. Behavior like update of user profiles can also be associated within the RDF annotation of the resource and as Datalog programs. RDF annotations provide several possibilities for specifying relationships and association, as defined by the RDFS schema, and topic ontologies are defined as RDF data again in the form of topic ontologies.

If we compare our work with standard models for adaptive hypermedia systems such as the one used in AHA! [7] for example, we observe that they define several model like conceptual, navigational, adaptation, teacher and user models. These models either correspond to ontologies / taxonomies in our case, to different schemas describing teacher and user profile, and to schemas describing the navigational structure of a course. Adaptation functionalities are expressed as Datalog queries in our model, while the adaptation model in AHA uses a rule based language encoded into XML. At the level of concept or information items AHA! provides functionalities to describe requirements [5] for the resource, which state what is required from a user to visit that information. In our case we describe these requirements by Datalog based accessibility constraints.

Our work is also related to [15, 16], and extends it by investigating the different standards relevant for adaptive functionalities in an open environment and how to use queries to implement that functionality.

6. CONCLUSIONS AND FURTHER WORK

In this paper we investigated the possibility of implementing traditional adaptive hypermedia functionalities in an open environment, and discussed existing standards for describing learning objects and user models as a necessary prerequisite for such open adaptive systems. We discussed how these information can be expressed as RDF metadata and how we can use queries over this metadata in a distributed network to implement different adaptive functionalities. We also discussed as a very important feature of our architecture, that it needs no central server or data repository, and can be implemented as a system based on distributed peers in an extendable P2P network. We finally discussed a system implemented as a prototype for such an open adaptive hypermedia system, which has been implemented as a peer in the RDF-based P2P network Edutella.

In our further work we will continue to improve RDF bindings (only prototype versions have been defined for some of them, especially PAPI and IMS LIP). We will also experiment with resource compositions and adaptive presentation techniques and different kinds of queries for adaptive functionalities.

The open environment raises many questions regarding to user privacy (see e.g. [21] for the discussion). This includes not only user's concerns about abuse of his private features but also privacy restrictions of law in several countries. However, the privacy issues require further research.

7. ACKNOWLEDGMENTS

This work is partially supported by EU/IST ELENA project (`http://www.elena-project.org`), PADLR project (`http://www.learninglab.de/padlr/index.html`), and ULICampus project (`http://www.uli-campus.de/english/project.html`).

8. REFERENCES

[1] The dublin core metadata initiative. http://dublincore.org/.

[2] The acm computer classification system. http://www.acm.org/class/1998/, 2002.

[3] ALLERT, H., DHRAIEF, H., AND NEJDL, W. How are learning objects used in learning process? In *Proceedings of the ED-MEDIA 2002, World Conference on Educational Multimedia, Hypermedia & Telecommunications* (June 2002).

[4] BAILEY, C., HALL, W., MILLARD, D., AND WEAL, M. Towards open adaptive hypermedia. In *Proccedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002)* (Malaga, Spain, May 2002).

[5] BRA, P. D., AERTS, A., SMITS, D., AND STASH, N. AHA! version 2.0. In *Proceedings of the AACE ELearn'2002 conference* (Oct. 2002), pp. 240–246.

[6] BRA, P. D., AND CALVI, L. AHA!: An open adaptive hypermedia architecture. *The New Review of Hypermedia and Multimedia 4* (1998), 115–139.

[7] BRA, P. D., HOUBEN, G.-J., AND WU, H. AHAM: A dexter-based reference model for adaptive hypermedia. In *Proc. of ACM Conference on Hypertext and Hypermedia* (Darmstadt, Germany, Feb. 1999), K. Tochtermann, J. Westbomke, U. Wiil, and J. Leggett, Eds., pp. 147–156.

[8] BRASE, J., AND NEJDL, W. Ontologies in elearning. Tech. rep., university of hannover, Nov. 2002. to be published in "Handbook on Ontologies", Springer-Verlag 2003.

[9] BRICKLEY, D., AND GUHA, R. V. W3c resource description framework (rdf) model and syntax specification. http://www.w3.org/TR/1998/WD-rdf-schema/. Accessed on October 25, 2002.

[10] BRUSILOVSKY, P. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction 6*, 2-3 (1996), 87–129.

[11] BRUSILOVSKY, P. Adaptive hypermedia. *User Modeling and User-Adapted Interaction 11*, 1-2 (2001), 87–100.

[12] DOLOG, P., AND BIELIKOVÁ, M. Transforming state diagrams into navigation objects in hypermedia applications, July 2002. Technical Report. Department of Computer Science and Engineering, Slovak University of Technology.

[13] GONG, L. Project jxta: A technology overview. Technical report, SUN Microsystems. http://www.jxta.org/project/www/docs/TechOverview.pdf. Accessed on October 25, 2002.

[14] HENZE, N., AND NEJDL, W. Adaptivity in the kbs hyperbook system. In *2nd Workshop on User Modeling and Adaptive Systems on the WWW at the World Wide Web Conference (WWW 8), and the Seventh International Conference on User Modeling (UM 99)* (Toronto, Canada, May 1998).

[15] HENZE, N., AND NEJDL, W. Adaptation in open corpus hypermedia. *IJAIED Special Issue on Adaptive and Intelligent Web-Based Systems 12* (2001).

[16] HENZE, N., AND NEJDL, W. Knowledge modeling for open adaptive hypermedia. In *Proccedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2002)* (Malaga, Spain, May 2002).

[17] IEEE. IEEE P1484.2/D7, 2000-11-28. draft standard for learning technology. public and private information (papi) for learners (papi learner). Available at: http://ltsc.ieee.org/wg2/. Accessed on October 25, 2002.

[18] IEEE-LTSC. Ieee lom working draft 6.1. Available at: http://ltsc.ieee.org/wg12/index.html. Accessed on October 25, 2002.

[19] IMS. IMS learner information package specification. Available at: http://www.imsproject.org/profiles/index.cfm. Accessed on October 25, 2002.

[20] KIESLINGER, B., SIMON, B., VRABIC, G., NEUMANN, G., QUEMADA, J., HENZE, N., GUNNERSDOTTIR, S., BRANTNER, S., KUECHLER, T., SIBERSKI, W., AND NEJDL, W. Elena - creating a smart space for learning. In *Late-Breaking topics session at International Semantic Web Conference (ISWC 2002)* (Sardinia, Italy, 2002).

[21] KOBSA, A. Personalized hypermedia and international privacy. *Communications of the ACM 45*, 5 (2002), 64–67.

[22] LASSILA, O., AND SWICK, R. W3c resource description framework (rdf) model and syntax specification. Available at: http://www.w3.org/TR/REC-rdfsyntax/. Accessed on October 25, 2002.

[23] MITROVIĆ, A. Porting SQL-Tutor to the web. In *Proc. ITS'2000 workshop on Adaptive and Intelligent Web-based Education Systems* (June 2000), pp. 37–44.

[24] NEJDL, W., WOLF, B., QU, C., DECKER, S., SINTEK, M., NAEVE, A., NILSSON, M., PALMR, M., AND RISCH, T. Edutella: A p2p networking infrastructure based on RDF. In *Proceedings of the 11th International World Wide Web Conference (WWW2002)* (Hawaii, USA, June 2002).

[25] NEJDL, W., WOLF, B., STAAB, S., AND TANE, J. Edutella: Searching and annotating resources within an RDF-based p2p network. In *Proceedings of the Semantic Web Workshop* (Honolulu, Hawaii, May 2002).

[26] NILSSON, M. Ims metadata rdf binding guide. http://kmr.nada.kth.se/el/ims/metadata.html, May 2001.

[27] WEBER, G., AND SPECHT, M. User modeling and adaptive navigation support in www-based tutoring systems. In *Proc. of User Modelling '97* (June 1997), A. Jameson, C. Paris, and C. Tasso, Eds., pp. 289–300.