The Future of Learning Object Metadata Interoperability

Towards an Interoperability Framework for Metadata Standards

Mikael Nilsson <mini@nada.kth.se>

KMR Group, NADA, Royal Institute of Technology, Stockholm, Sweden

Pete Johnston <p.johnston@ukoln.ac.uk>

UKOLN, University of Bath, Bath, United Kingdom

Ambjörn Naeve <amb@nada.kth.se>

KMR Group, NADA, Royal Institute of Technology, Stockholm, Sweden

Andy Powell <a.powell@ukoln.ac.uk>

UKOLN, University of Bath, Bath, United Kingdom

Learning Objectives

This chapter will help you understand:

- the notion of metadata interoperability,
- the fundamental principles behind the metadata formats used for describing learning objects,
- how and when metadata standards can, and cannot, be used in combination,
- the role of abstract models for metadata in enabling metadata interoperability,
- how application profiles can be used to customize metadata standards,
- the importance of metadata semantics, and
- future directions of learning object metadata standards.

Executive Summary

A central task when managing learning objects is the administration of metadata. Metadata may consist of many kinds of information about the learning object, from descriptions and subject classifications to relations to other learning objects and accessibility characteristics.

There are currently a number of metadata standards in use within the e-learning community. IEEE Learning Object Metadata (LOM) is usually regarded as the most central standard in this field, but in recent years it has become apparent that standards from other communities, such as digital libraries, online multimedia and e-Government also play an important role for e-learning systems. The reason is simple: many potential learning objects have their origin in other kinds of repositories of digital content, and their metadata, while described in a way that fits the original community, is of great value in an e-learning context too.

This chapter explains a number of major difficulties encountered when trying to use such metadata in combination and explores a number of developments that will lead to solutions to the problems. Three major difficulties will be analyzed:

- the markedly differing metadata formats and abstract models,
- the apparent utopia of application profiles, and
- the elusive notion of metadata semantics.

At a first glance, the major problem of metadata interoperability seems to be about formats: the different standards all use different methods of encoding their information. Nowadays, many standards use XML-based encodings, but using XML is not a guarantee for interoperability. The chapter will explain the complex issues arising from the use of different syntaxes, such as XML, RDF and HTML meta tags.

Even if the syntax issue would be solved, many issues remain. Some standards, such as Dublin Core, rely on an abstract framework that fits into many syntaxes. We will explain the point of abstract models for metadata and how they support metadata interoperability.

Underlying formats and abstract frameworks is the subtle notion of semantics. With the rise of the Semantic Web initiative of the W3C, the semantics of metadata descriptions has received increasing attention. Semantics turns out to be a central aspect of metadata interoperability. We try to explain why this is so, and draw out the implications for learning object metadata standards.

Setting formats and semantic issues aside, we will explore what it means to combine metadata from different standards. Many metadata standards are actually based on one of the core standards, through the use of so-called application profiles. However, looking more closely at this notion, it turns out that it is highly problematic and not very well supported by the current set of metadata standards.

It turns out that all these issues are deeply connected, and that the future development of learning object metadata promises to dramatically improve syntactic and semantic interoperability as well as modularity of metadata systems. The chapter will discuss how some of this is already possible today, and how tomorrow's learning object metadata standards will fit into this framework.

Introduction

The administration and exchange of metadata is a central activity in systems that manage learning objects. Metadata considerations are fundamental when creating interoperable e-learning tools, and metadata standards have been among the very first learning technology standards to mature. But despite enormous progress in the harmonization of learning object metadata standards, culminating with the release of the IEEE Learning Object Metadata standard in 2002, there remains a core of unsolved issues with respect to metadata interoperability.

Obscured by difficulties in precisely defining the metadata concept, and behind an abundance of conflicting recommendations and opaque metadata formats, these issues are not apparent to the casual observer. This chapter will bring some of these interoperability issues to the surface, and in the process give some insight into how they might be addressed in future learning object metadata standards.

Metadata for Learning Objects

Metadata is usually defined as “data about data”, i.e., any kind of information that in some way references or describes aspects of some other piece of information. Metadata is introduced in information management systems in order to support certain administrative operations, including searching, displaying summaries or configuring interfaces. In essence, metadata creates a level of indirection, allowing systems to manage resources without ever having to delve into their physical or digital internals.

In an e-learning context, metadata may consist of many kinds of information about a learning object, from descriptions and subject classifications to accessibility characteristics and relations between learning objects. For example, learning object metadata may be used by cataloguing software for indexing, by learning management systems for matching learners with relevant resources, and by content players that configure the learning object to the user's environment and needs.

Background

Metadata in a broad sense is barely something new. Library catalogues are metadata, and allow librarians to manage a large library without unnecessarily having to deal with the physical books themselves. The same holds for maps that allow you to manage land, gravestones that give you information about deceased persons, and so on. Indeed, the two latter examples highlight that the term "metadata" is used to include descriptions that provide information about things that are not necessarily information artefacts.

Today, the term “metadata” usually refers to information with one fundamentally different characteristic as compared to the above examples: it is *machine-processable*, i.e. it is expressed in a way that allows computers to search, sort and present metadata without human intervention. Metadata in this modern sense has been part of computer systems since their early days, for example in file systems where file names and file permissions are metadata about the files.

With the rise of computer networks, metadata gained a new kind of importance. Geographically separated systems with different implementations, but managing the same kinds of data needed to communicate, and metadata standards focused on interoperability *between* systems were developed. Early examples of metadata standards include standards for library information exchange (the MARC format) and standards for geo-spatial data, used for map making. Another metadata standard of enormous importance is IETF RFC 822 from 1982 that specifies the format of e-mail headers, enabling email systems to transfer messages from the sender's computer to that of the addressee.

The growing use of Internet technology, and in particular, the World Wide Web became a strong driving force for the development of more generally applicable metadata standards. With the WWW a whole new usage pattern of metadata surfaced. Not only were the resources described of a much more diverse nature, but the applications using the metadata were also of many different kinds. The users of metadata were not only large computer systems but also individuals in front of their desktop computers.

This diversity of systems and resources leads to many new demands on metadata standards in general, and provides the fundamental functional requirements on learning object metadata standards in particular. We will argue that current learning object metadata standards are not fully up to the task of managing this diversity, and we will then describe how this problem needs to be tackled.

Defining Metadata

It can be argued that the above definition of metadata as "data about data" may be too narrow because it does not allow information about non-digital things, such as persons, places or books to be called metadata. On the other hand, it may be too broad because it allows any kind of description, such as an image of a learning object, to be considered metadata.

In practice, most modern metadata standards adopt a definition of metadata that allows descriptions about digital or non-digital things alike, usually collectively termed *resources*, but limits the type of descriptions to a very restricted kind of data, as defined by the metadata standard.

In this chapter, we will use the term "metadata" in the sense of the following modern definition:

> *Machine-processable information about resources*

This definition encompasses not only human-assigned information about a resource (such as name/title, subject and creator), but may also be used for information relating to e.g.

- the life cycle of a piece of information (different versions, history, etc.)
- technical aspects of the resource (size, format, functionality, etc.)
- relations between resources and aggregations of resources (lessons comprised of learning objects etc.)

It encompasses information not only about digital resources, but also about e.g.

- learners and teachers (history, competencies, etc.)
- events (location, participants etc.)
- abstract notions (pedagogical designs, terms in taxonomies etc.)

We will later return in more detail to the notion of machine-processability, which is central for understanding the future developments in learning object metadata standards.

The Notion of Interoperability

What, then, do we mean with the all-important term *interoperability* in a metadata context? Learning object metadata interoperability refers to the ability of different systems to exchange information about resources. Metadata created by a human user in one system and then transferred to a second system will be processed by that second system in ways which are consistent with the intentions of the user who created the metadata.

However, for metadata standards, the goals for interoperability have been set much higher than this basic level. Duval, Hodgins, Sutton, and Weibel (2002) set forth four fundamental principles for interoperability *between metadata standards.* Three of these are repeated in the Dublin Core – IEEE LTSC Memorandum of Understanding ("Memorandum", 2000). These are:

- **Extensibility**, or the ability to create structural additions to a metadata standard for

application-specific or community-specific needs. Given the diversity of resources and information, extensibility is a critical feature of metadata standards and formats.

- **Modularity**, or the ability to combine metadata fragments adhering to different standards. Modularity is stronger than simple extensibility in that it requires that metadata from different standards, including metadata extensions from different sources, are usable in combination without causing ambiguities or incompatibilities.
- **Refinements**, or the ability to create semantic extensions, i.e., more fine-grained descriptions that are compatible with more coarse-grained metadata, and to translate a fine-grained description into a more coarse-grained description.

While important, the fourth principle mentioned in Duval et al (2002), *multilingualism,* is not the kind of technical problem we will discuss in this chapter. However, based on the above discussion, we will add another principle to this set:

- **Machine-processability**, or the ability to automate processing of metadata. Particularly important is automatic processing of extensions, modules and refinements.

The wider notion of interoperability presented here is unfortunately not fully realized in current learning object metadata standards. This chapter is devoted to explaining why this is so, and to describing how the above principles may be realized in future standards. The principles of extensibility, modularity and refinements will be central in this analysis, and machine-processability will form an overarching theme.

Metadata Standards in the LO Domain

There are currently a number of metadata standards in use within the e-learning domain. IEEE Learning Object Metadata (LOM), published in 2002, is usually regarded as the dominant standard in this field, but in recent years it has become apparent that standards from other communities, such as digital libraries, digital multimedia and e-Government also play an important role. The reason is simple: many potential learning objects have their origin in other kinds of communities, and are described in a way that fits that specific community. The division of resources into categories such as “learning objects”, “library material” etc. is fading in favour of a broader notion of multi-purpose content.

Apart from the IEEE LOM standard, some of the most important metadata standards that are relevant for learning objects are:

- The Dublin Core metadata standard, popular on the World Wide Web and in the digital library community;
- MPEG-7, a complex metadata standard for digital video;
- MODS, an XML encoding of parts of the de facto library metadata standard MARC;
- A number of specifications from the IMS Global Learning Consortium, such as IMS Content Packaging, IMS Question and Test Interoperability and IMS Learner Information Package that have metadata parts.

Additionally, a number of metadata standards and specifications that are based on one of the above are also available. Based on Dublin Core are for example EdNA, a metadata standard for the Australian Education Network, and GEM, a US government-sponsored Gateway to Educational Materials. Based on LOM we find among many others the RDN/LTSN LOM application profile (RLLOMAP) and the Curriculum Online Metadata Schema. The IMS

metadata standard and SCORM also reuse LOM as a basis on top of which they build their own frameworks.

These various standards and specifications have been developed to meet different requirements, and to support the needs of different communities. In some cases standards reflect the broadly shared requirements of a large community; in others, they reflect more specific requirements of a smaller or more specialised community, perhaps defined by activity/interest or by geopolitical boundaries.

The development of these specifications has highlighted the necessity of being able to use component parts of different standards in combination. Because these standards are not designed to be compatible, this is unfortunately not possible today. We will later see precisely why, and describe a path to a long-term solution.

Dublin Core

The Dublin Core Metadata Initiative was started in 1995 as a reaction to the problems of finding resources on the growing World Wide Web. It is used worldwide by a broad range of systems and organizations on the WWW and in various closed infrastructures.

Initially, Dublin Core consisted of 15 terms which were designed to express simple textual information about resources. The project has since grown to accommodate around 80 terms, some of which are of general nature (such as “title” and “subject”), while others are community-specific (such as “educationalLevel” or “bibliographicCitation”).

The terms in Dublin Core come in three kinds: *properties* (also called “elements”), *syntax encoding schemes* and *vocabulary encoding schemes*. Properties are used to describe a specific aspect of a resource, while the two kinds of encoding schemes are used to specify details of the value of a property. Properties are defined independently of each other, and Dublin Core allows metadata containing any number and combinations of properties to be used to describe a resource.

The term “Simple DC” is sometimes used to describe a usage pattern of Dublin Core metadata that limits itself to the original 15 terms in the Dublin Core Element Set, used in a pattern where each is optional and repeatable.

IEEE LOM

The IEEE LOM standard has its origins in earlier work within the European ARIADNE project and the IMS Global Learning Consortium, beginning in 1995. In 2002, IEEE finally approved LOM as an international standard, and LOM has since enjoyed an ever-increasing support from other specification bodies and application developers within the e-learning field.

LOM consists of a single hierarchy of 76 elements divided into nine categories, and specifies vocabularies and allowed syntaxes for the value of each element. It can be used to convey not only metadata useful for resource discovery, but also information such as aspects of the lifecycle of a learning object and pedagogical features.

While the terms in Dublin Core are defined and used independently of each other, the LOM standard specifies the structure of the whole of its hierarchy of metadats in a single standard. The standard specifies where in this hierarchy each element may appear, whether it may be repeated, whether ordering matters, and so on. The meaning of a LOM element depends on the precise structural context in which it appears.

In effect, LOM specifies both the elements themselves and a set of rules for using the elements in combination, a basic example of a so-called “application profile”. One advantage of this approach is that it allows for much stricter validation of LOM data as compared to Dublin Core, something that makes LOM immediately usable without further customization.

MPEG-7, MODS and the IMS Standards

MPEG-7 is the name of a digital video standard with a heavy focus on the use of metadata to describe the content of a video stream. What makes MPEG-7 interesting is the fact that it has the potential to be deeply integrated into the video production process, something that generally can be expected to result in very high metadata quality. MPEG-7 also represents a challenge in that the resources it describes can be extremely intangible, such as an appearance of a certain person in a movie. By contrast, other metadata standards such as LOM and Dublin Core have been developed in a library tradition, using a document metaphor.

This metadata standard does not contain any information specific to learning, but several parts of the information embedded in MPEG-7 metadata might still be useful for an e-learning application.

MODS has been developed by the Library of Congress to serve as a modern version of the widely used MARC format for library cataloguing data. It is not oriented towards educational applications, but is nevertheless interesting because of the large amounts of high quality library metadata available in this format.

The IMS Global Learning Consortium has created a diverse set of standards for use in e-learning systems. Although only one of them, the LOM-based IMS Metadata specification, calls itself a metadata standard, there are a number of standards within IMS that, as a whole or in part, fit our definition of a metadata standard. The part of IMS Content Packaging that specifies how to describe the structure of a package of learning objects would classify as a metadata standard, as would the description of a learner in IMS Learner Information Package, etc.

The structure of MPEG-7 metadata, MODS and the IMS standards are in many ways similar to LOM in that they are complex, monolithic hierarchies of data elements with strict structural constraints, even though the details of how the hierarchies are constructed differ substantially. For example, MPEG-7 defines a complex so-called Data Description Language (DDL) that is used to customize the metadata format to a certain application. This language is based on a completely different set of principles than the MODS specification or the IMS set of specifications.

From a metadata interoperability perspective we would expect to be able to combine information from all these different standards in descriptions of the persons, artefacts, events, etc. that make up an e-learning system. In practice, this is currently difficult or impossible to do. Instead, each standard lives in isolation, largely incompatible with the others. The reason for this is not tied to any single standard, but originates in the lack of a common platform for metadata standards in general.

The rest of this chapter will focus exclusively on the relationship between Dublin Core and LOM, as this will highlight the most important difficulties with trying to combine two approaches to defining metadata. However, the lessons learned will be applicable to a much broader range of standards, including the standards mentioned above.

Extensibility in Metadata Formats

At a first glance, the major problems of metadata interoperability seem to relate to formats: LOM and Dublin Core use different methods of encoding their information.

We will find, however, that the formats currently used by LOM and Dublin Core actually do allow for extending the format and combining terms from two standards. The problem instead lies on another level, in the *interpretation* or *semantics* of the metadata expressions. In particular, metadata applications will have trouble understanding LOM terms in a DC context, and vice versa.

In order to understand these difficulties, we must first explore how the two standards approach the issue of metadata formats.

Bindings

Both LOM and Dublin Core use a two-layered approach to defining metadata models. In the core standards, an abstract information structure is defined, defining the terms that may be used and their relationships. This information structure can then be encoded in one of several alternative formats, called *bindings*. As an example, Dublin Core currently supports three bindings:

- "meta" tags in HTML/XHTML
- XML, the Extensible Markup Language, a general-purpose text-markup and data exchange language
- RDF, the Resource Description Framework, a general-purpose metadata framework

The situation with LOM is similar. At the time of writing, an XML binding for LOM has just been approved by the IEEE, while an RDF binding is in development.

Bindings to other formats are sometimes necessary, of which some see wide-spread use and others are only used for internal purposes. Many applications use such "private bindings" for, e.g., implementing their metadata in a relational database, or embedding metadata in a private protocol. One such example is the News Metadata Framework (The International Press Telecommunications Council, 2005).

We note that all three formats listed above are specified by the W3C, which should not be surprising as the interoperability problems we are studying arise in a WWW context. Because of the limited generality of "meta" tags and the fact that they will be replaced by an RDF-compatible metadata module in XHTML2, we will not discuss them further here. Instead, we will concentrate on the two major current metadata formats: XML and RDF.

XML-based Formats

An XML document can be represented as a tree structure of *XML elements*. Each element may contain text as well as other XML elements, and may also have *attributes*. While XML has its origins in standards for creating structured text documents, it is widely used to encode data of many kinds.

XML itself does not provide a fixed set of element names and attribute names. Rather, users of XML define their own *XML language*, or in other words: a set of element names and attribute names for use in XML documents and a set of rules for how those named elements and attributes are to be interpreted. For this reason, the XML standard itself is sometimes referred to as a *meta-language,* i.e., a set of rules for defining XML languages.

Each of the metadata standards mentioned above define their own such XML language. One such language is the LOM XML binding defined by the IEEE, exemplified in Example 1.

```
<?xml version="1.0"?>
<lom xmlns="http://ltsc.ieee.org/xsd/LOM" >
  <general>
    <identifier>
      <catalog>URI</catalog>
      <entry>http://www.example.com/objects/Para101</entry>
    </identifier>
    <language>fr</language>
    <description>
      <string language="en">
        This learning object explains parachuting.
      </string>
    </description>
    <structure>
      <source>LOMV1.0</source>
      <value>atomic</value>
    </structure>
  </general>

  <educational>
    <description>
      <string language="en">
        Useful for learning some flight-related French terminology.
      </string>
    </description>
    <language>en</language>
  </educational>
</lom>
```

Example 1. A LOM XML metadata instance

This XML file is a metadata description (albeit somewhat unrealistic) of a learning object about parachuting. The LOM XML binding tells us in detail how to interpret each XML element. For example, we can see that although the atomic learning object is in French ("fr"), it is intended for English-speaking learners ("en"), and the real purpose is to learn flight-related French terminology.

The LOM XML binding thus specifies the precise semantics of each XML element, *in the context it appears*. As we can see from the example above, the XML element "language", when taken on its own, is ambiguous; it must be interpreted differently when it appears as a sub-element (or *child*) of the "general" and "educational" elements, respectively. It is therefore necessary for the LOM XML binding to specify the semantics of the complete XML document as a whole, taking all parent/child relations into account.

Another XML language is specified in the Dublin Core XML encoding guidelines. Example 2 shows a resource described using Dublin Core and encoded in that language:

```
<?xml version="1.0"?>
<metadata resource="http://www.example.com/objects/Para101"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:dcterms="http://purl.org/dc/terms/">

  <dc:subject xsi:type="dcterms:LCSH">tl750</dc:subject>
```

```
  <dc:description>
    This learning object explains parachuting.
  </dc:description>
  <dcterms:isPartOf xsi:type="dcterms:URI">
    http://www.example.com/courses/French344
  </dcterms:isPartOf>
  <dcterms:modified xsi:type="dcterms:W3CDTF">
    2001-07-18
  </dcterms:modified>
  <dc:format xsi:type="dcterms:IMT">
    text/html
  </dc:format>

</metadata>
```

Example 2. A Dublin Core XML metadata instance. Note that Dublin Core does not currently mandate a particular container element in XML description, so we have used an arbitrary "metadata" container element.

From this description and the semantics defined by Dublin Core, we can understand that the resource described is about parachuting (code tl750 in Library of Congress Subject Headings), and that is part of the course identified by "http://www.example.com/courses/French344", that it is in HTML format, etc.

Other XML languages for Dublin Core and for LOM are perfectly possible. These will have their own rules for interpreting the XML data, and will operate independently of the official bindings. Note that such alternative languages may reuse XML element names from the official bindings, but use them together with a different set of rules.

RDF

RDF was designed as an extensible framework for metadata descriptions. It was created within the Semantic Web initiative at the World Wide Web Consortium (W3C), which aims to create a global network of machine-processable data as an extension to the WWW.

Unlike XML, RDF is not a meta-language, i.e., you do not need to create your own RDF-based language. Instead, RDF allows descriptions using parts from different metadata standards and terms from independent vocabularies to coexist within the *same* metadata language. It is thus fair to say that RDF has been designed to fulfil the role of a general-purpose metadata language.

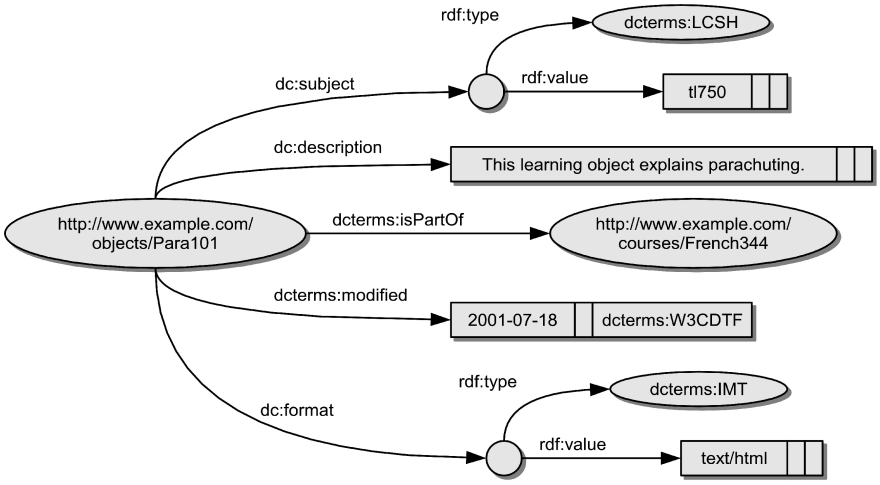

Figure 1. An example of a Dublin Core description expressed in RDF.

RDF metadata is made up of sets of *statements*. Each statement describes a single attribute, or *property*, of a single resource. By combining several statements about the same resource, a metadata description of that resource can be constructed. RDF data can be represented as a nodes-and-arcs diagram, where the nodes represent resources, and arcs represent properties. The Dublin Core example given in the previous section can be expressed in RDF as in Figure 1.

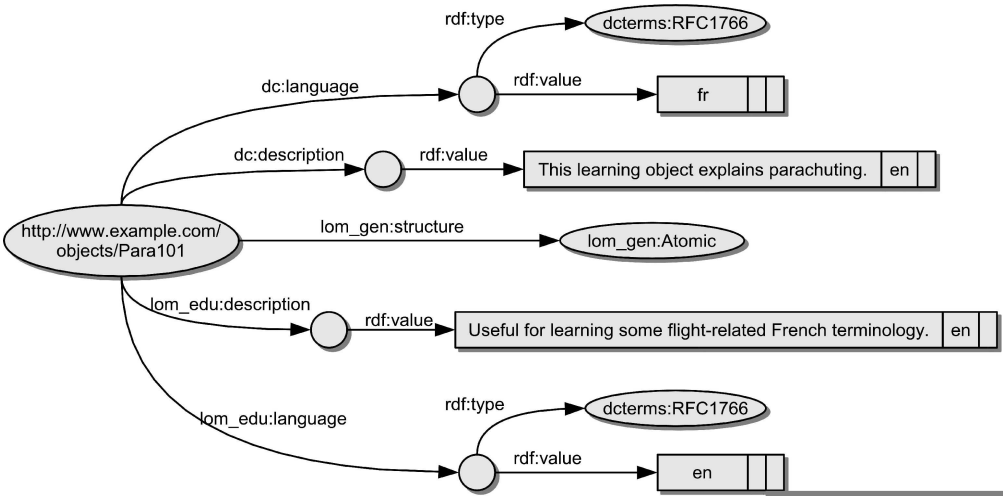

Figure 2. An example of a LOM instance expressed in RDF.

Expressing the LOM example using the draft LOM RDF binding gives us the RDF metadata depicted in Figure 2.

In this example, we can see that in single RDF description, terms from several standards are combined. RDF itself specifies a base vocabulary that is used for specifying resource types (the

"rdf:type" property), Dublin Core specifies a resource type that is used to represent languages (the "dcterms:RFC1766" type), and LOM specifies a property to be used to describe a resource using a value of that type (the "lom_edu:language" property). We note that the LOM RDF binding has chosen to reuse Dublin Core properties for expressing common properties such as "language" and "description".

While the graph notation for RDF is very useful, it cannot be used for exchanging metadata between computer systems. For this purpose, a serialization of RDF into an RDF-specific XML language is used. This RDF/XML language is an example of an XML language that contains XML elements with identical names as XML elements in the Dublin Core XML language (such as "dc:description). But as noted, these elements will now be interpreted using the rules of the RDF/XML language.

It is important not to confuse this RDF/XML serialization with RDF itself, which is not bound to a specific syntax. We will return to a fuller description of RDF and the Semantic Web later.

It is also important to realise that RDF does not allow for incompatible usages of the same terms. In contrast to XML, that allows the reuse of identical XML elements across many different XML languages, with different structural constraints and interpretation, RDF does not leave room for private semantics of properties. For example, the Dublin Core RDF property "dc:language" must be used in accordance with the RDF semantics and RDF constraints defined by the Dublin Core Metadata Initiative in all RDF metadata instances, even when used in, for example, the LOM RDF instance given above.

Extending and Combining Metadata Descriptions

We have seen how both LOM and Dublin Core can be expressed in XML and in RDF. But can we combine terms from both standards in a single document? The answer is both yes and no.

For example, on the surface it seems straightforward to add XML elements from Dublin Core to a LOM XML document. Let us say we want to use the educational description from LOM, and the subject from Dublin Core. Example 3 is the result of extending a LOM XML document with a fragment from Dublin Core.

```
<?xml version = "1.0"?>
<lom  xmlns="http://ltsc.ieee.org/xsd/LOM"
      xmlns:dc="http://purl.org/dc/elements/1.1/"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <general>
    <identifier>
      <catalog>URI</catalog>
      <entry>http://www.example.com/objects/Para101</entry>
    </identifier>
  </general>

  <educational>
   <description>
     <string language="en">
       Useful for learning some flight-related French terminology.
     </string>
   </description>
   <language>en</language>
  </educational>
```

```
  <dc:subject xsi:type="dcterms:LCSH">tl750</dc:subject>

</lom>
```

Example 3. A LOM XML metadata instance, extended with a Dublin Core XML metadata fragment

As we can see, the Dublin Core XML fragment describing the subject of a resource can be added into the LOM XML document. On the other hand, we can do the reverse, starting from the Dublin Core XML document and adding the LOM fragment from the element "Educational.Description". The result is shown in Example 4.

```
<?xml version="1.0"?>
<metadata resource="http://www.example.com/objects/Para101"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:lom="http://ltsc.ieee.org/xsd/LOM" >

  <dc:subject xsi:type="dcterms:LCSH">tl750</dc:subject>

   <lom:educational>
    <lom:description>
      <lom:string lom:language="en">
        Useful for learning some flight-related French terminology.
      </lom:string>
    </lom:description>
    <lom:language>en</lom:language>
  </lom:educational>

</metadata>
```

Example 4. A Dublin Core XML metadata description, extended with a LOM XML metadata fragment

Note how we need to bring in the whole "educational" LOM element in order to create a context for the "description" element. Simply copying the "description" element will be ambiguous, as it is used in LOM to mean different things in different contexts. Also, we chose not to bring with us the "General.Identifier" element, as the Dublin Core structure already provides that information.

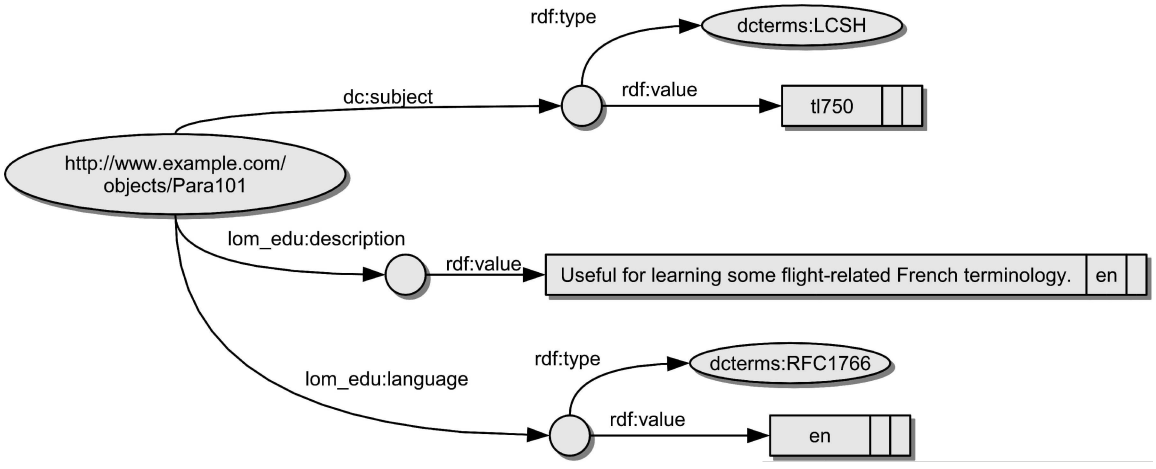

Figure 3. A combined LOM and Dublin Core metadata description, expressed in RDF.

How about doing the same kind of combination in RDF? It is just as straightforward: we simply merge the two diagrams in our examples, and arrive at an RDF description looking like Figure 3. In fact, our original LOM RDF example in Figure 2 already showcases this kind of combination.

One important difference between RDF and XML is that XML creates two cases: one case where a LOM XML instance is extended with Dublin Core XML metadata, and one case where a Dublin Core XML description is extended with LOM XML metadata (and this combinatorial problem increases if we add a third standard to the mix). By contrast, RDF does not distinguish between the two cases – the results are identical.

Mixing standards thus seems possible in both XML and RDF. Unfortunately, straightforward as both examples appear, insurmountable problems start to appear as we examine how metadata applications are to process the metadata we have constructed. The tool we need to understand the difficulties is called an *abstract model*, and we now turn to this subject before returning to our examples.

Abstract Models for Metadata

In order to be format-independent, both LOM and Dublin Core base are based on the notion of an *abstract model*. The abstract model specifies the concepts used in the standard, the nature of terms and how they combine to form a metadata description. The abstract model is the key used by a metadata application to unlock the secrets of a metadata expression given in a specific format, thus making it possible for a single standard, though expressed in several different formats, to still be understood in a uniform way by users and applications. An early effort to produce such framework for Dublin Core was presented in Bearman, Miller, Rust, Trant and Weibel (1999).

The abstract models of LOM and Dublin Core are fundamentally different in several ways, and these differences are a major source of difficulties when trying to combine the standards. As we will see, applications will find that terms from one standard make little sense if interpreted in the context of the other standard.

The Dublin Core Abstract Model

The Dublin Core Abstract Model (Powell, Nilsson, Naeve and Johnston, 2005) defines the kinds of terms that can be used in Dublin Core metadata descriptions. Just as in RDF, a *property* is used to describe a single aspect of a resource. In a Dublin Core metadata description, any number of properties and their associated *values* may be used to describe a resource. The abstract model tells us that values can be referenced using a *value URI*, and further described in *related descriptions*. Values (such as titles) can be represented as *value strings* or using *rich representations* (images, HTML, etc.).

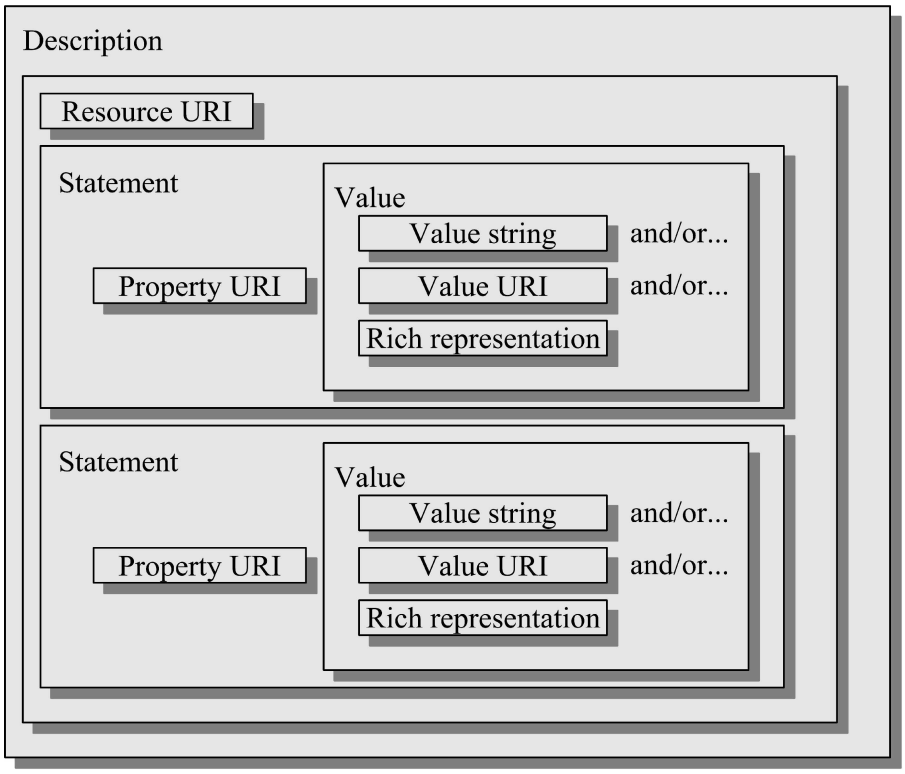

Figure 4. A simplified overview of the Dublin Core abstract model.

Syntax encoding schemes can be used to specify the precise syntax of value strings, while *vocabulary encoding schemes* are used to indicate a controlled vocabulary used as source of a value. An overview of the Dublin Core abstract model is found in Figure 4.

Using these constructs, it is possible to create very complex metadata descriptions. While some formats do not support all constructs in the abstract model (for example, HTML meta tags do not currently support the notion of related descriptions), the different formats all share the same common understanding of the basic notions of *properties* and *values*.

The LOM Abstract Model

Similarly, the LOM standard uses an abstract model to specify the structure of LOM metadata descriptions. In contrast to the *property-value* model used by Dublin Core, LOM uses a hierarchical structure of *elements-within-elements*. Each element can be either a container element, thus containing other elements, or a leaf element, which holds a value of a certain data type. The top-level elements are called *categories*.

The abstract model of LOM is somewhat similar to the XML element structure (though the two should not be confused). Unlike XML, LOM does not allow attributes on elements, nor does it allow text content in elements other than leaf elements. The same is true for the other metadata standards we have mentioned: MODS, MPEG-7 and the IMS standards – like XML, they are hierarchical in nature, but they are neither identical to XML, nor compatible with each other.

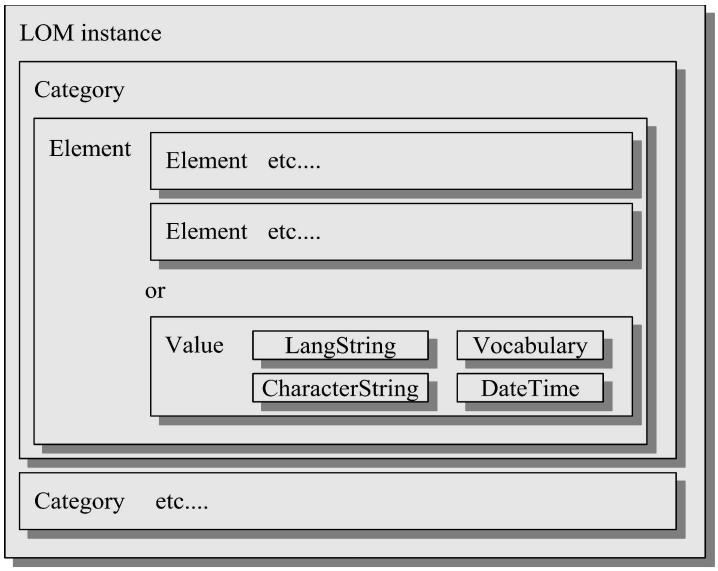

Figure 5. An overview of the LOM abstract model.

Interpreting Metadata Through the Lens of an Abstract Model

A binding is constructed by specifying how each kind of concept in the abstract model is to be encoded in a particular format. Conversely, the binding also specifies how to interpret data given in a specific format in terms of the abstract model. For example, when interpreting the Dublin Core XML example in Example 2 using the rules of the Dublin Core XML language, we can infer that "dcterms:modified" is a property, and "dcterms:W3CDTF" is a syntax encoding scheme for the value string "2001-07-18".

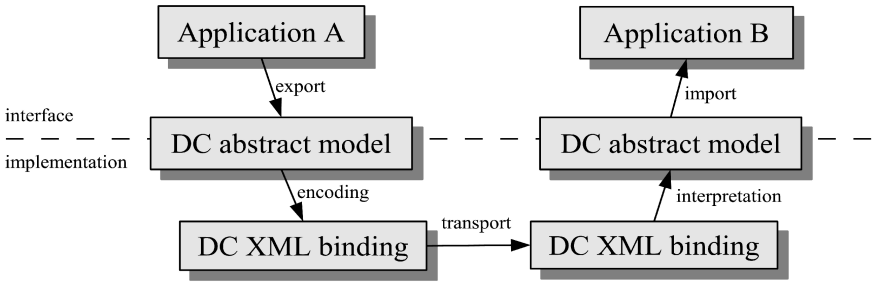

Figure 6. The process of encoding/interpretation of metadata within the framework of an abstract model.

This fundamental process of *encoding/interpretation* is described in Figure 6. Application A uses the Dublin Core abstract model to represent some metadata about a resource. This metadata is encoded using the Dublin Core XML binding, and transferred to another application. Application B will use the rules of the Dublin Core XML binding to interpret the XML data in terms of the Dublin Core abstract model. This representation of the metadata can then be used in the application. The LOM abstract model is similarly used by LOM applications as an intermediate layer between the application and the bindings.

When two applications want to exchange Dublin Core metadata, they understand metadata through the lens of the abstract model. The abstract model functions as an opaque interface, an API, to the metadata. In practice, the exchange is realized using one of the Dublin Core bindings, but the details of the formats are of no interest to the applications, which instead analyse the metadata in terms of the interface given by the abstract model.

Note that it is possible to produce applications that process metadata without regard to the abstract model. Such *ad-hoc processing* of metadata records requires that the precise content of the records is well-known in advance, which is the case in many systems where extensibility, modularity and refinements are not really issues. In contrast, the kind of *interoperable processing* based on the abstract model as described above is necessary when an application needs to be prepared for metadata constructs that do not fall within the limits of such a precise description. Thus, it should be clear that interoperable processing is a basic prerequisite for metadata interoperability.

Combining XML Languages

If we now try to understand what is really going on in the process of extending one standard using terms from another standard, the problem is much more evident. Let us recall Example 4 given earlier, in which a Dublin Core XML metadata description was extended using a LOM XML fragment. We saw that assembling the combined metadata description seems to work.

The step of *interpreting* the format in terms of the abstract model is the step that leads to difficulties when combining standards. The process is depicted in Figure 7. As in Figure 6, Application A produces Dublin Core metadata in the Dublin Core XML format, while Application C produces LOM metadata in the LOM XML format and inserts that into the Dublin Core XML metadata as in Example 4 above. Application B, which understands the Dublin Core abstract model, tries to interpret this combined XML document using the Dublin Core abstract

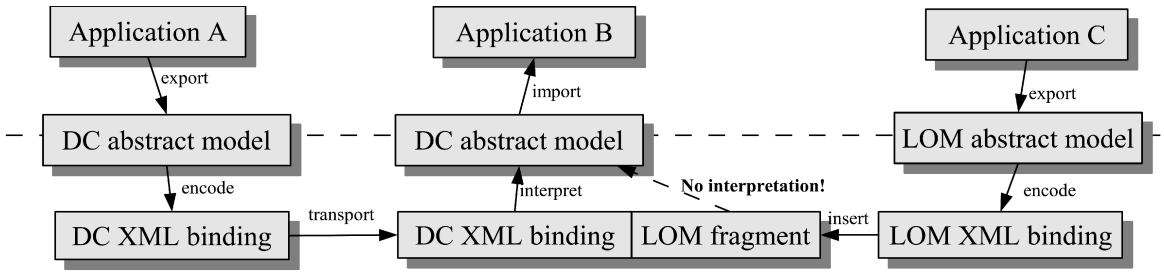

Figure 7. Combining the XML languages of LOM and Dublin Core.

model. The LOM XML fragment does not in any sense follow this model, and so is completely incomprehensible to the Application B. As an example, the application will try to interpret "lom:educational" as a property, but the child-element "lom:description" does not make sense as a value of that property, so the interpretation will fail.

In fact, without manual intervention the metadata description will also be incomprehensible if Application B was a LOM application, as it does not follow the LOM XML guidelines. For example, the root element is different from what LOM demands.

Trying the other way around, extending LOM XML with Dublin Core XML fragments, results in precisely the same kind of difficulties. The Dublin Core XML fragment does not follow the LOM abstract model, and the encoded information is therefore inaccessible to a LOM application. For example, the Dublin Core XML fragment uses XML attributes that cannot be meaningfully interpreted as LOM elements.

Another issue that arises in both cases is that of handling of resource identity, as the two standards have different conventions for how to identify which resource the metadata describes.

The result can be summarized in the following table:

Format	***Extended with fragment from***	***Processable by LOM application***	***Processable by Dublin Core application***
LOM XML	Dublin Core XML	Only LOM part	None
Dublin Core XML	LOM XML	None	Only Dublin Core part

So it seems extending the current XML formats for LOM and Dublin Core using terms from the other standard is a meaningless syntactic exercise, showing that the formats are, in fact, mutually incompatible. The same is true for most XML languages for metadata, such as MODS and MPEG-7 – they are based on different abstract models, and combining them will not increase interoperability.

In many ways, it is similar to trying to combine, say, English and Chinese text in a single Unicode document and expecting the combination to make sense, or to combine source code fragments from two different programming languages based on the premise that they use the same character encoding. There is actually no interoperability present, and the different metadata fragments might just as well be transmitted in separate XML files.

In order to fulfil our five metadata interoperability principles, we must find a better approach.

Combining RDF Descriptions

What happens if we try the same exercise with the RDF? The first difference, as mentioned earlier, is that the two cases of extending LOM with Dublin Core data or vice versa both lead to the same result. There is only one resulting RDF description.

The second difference is that being a metadata standard in its own right, RDF also brings us an abstract model with certain built-in base semantics. This means that RDF descriptions taken from different standards will be processable by a pure RDF application based on the RDF abstract model and semantics. The process is depicted in Figure 8, where, compared to Figure 7, the format used for exchange has been changed to RDF, and the Dublin Core application (Application B) has been replaced with an RDF application. Note how this differs from the case of XML, where a generic XML application will not be able to meaningfully process any of the information contained in different XML languages for metadata.

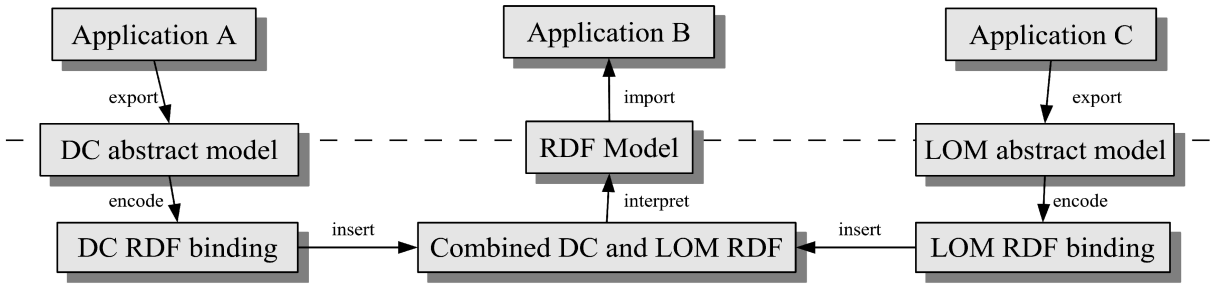

Figure 8. Combining RDF metadata from LOM and DC, interpreted through the RDF model.

Now, the Dublin Core abstract model is mostly compatible with this base semantics of RDF. Any metadata conforming to the Dublin Core abstract model can be translated into RDF and back. As a consequence, Dublin Core applications (in the place of Application B in Figure 8) are actually able to process the LOM metadata expressed in RDF. LOM properties will be correctly understood as properties, and their values and datatypes will be processable. This means that any metadata standard that is completely independent of Dublin Core, but is still expressed in RDF, will be partially processable by a Dublin Core application. This is no coincidence – RDF and Dublin Core has been heavily influenced by each other during their development.

By comparison, a LOM application (in the place of Application B in Figure 8) will only be able to process those parts of the RDF files that have been mapped from LOM elements, and will not be able to understand, for example, Dublin Core metadata expressed in RDF. The reason is that

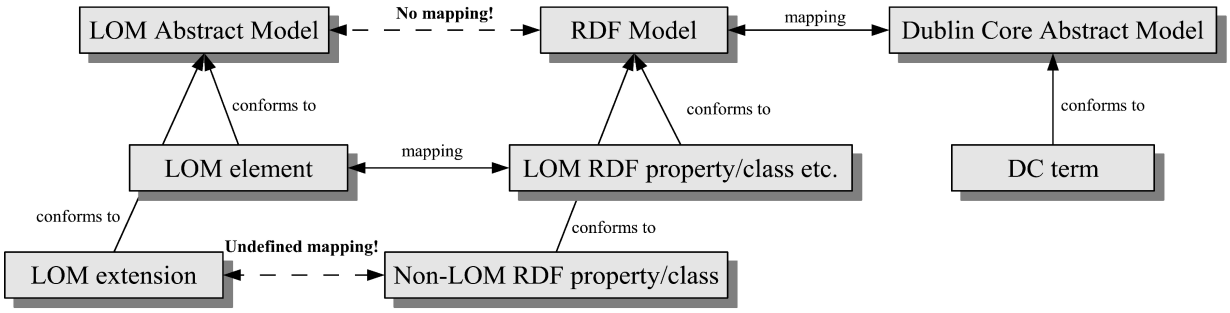

Figure 9. Mapping LOM elements to RDF is done one-by-one, while any term conforming to the Dublin Core abstract model can be mapped through the generic mapping to RDF.

the LOM elements must be translated individually, in an idiosyncratic way, to RDF – there is no way to construct a general translation of the elements-in-elements-based abstract model of LOM into the property-value-based abstract model of RDF and back. In other words, the abstract model of LOM and the base semantics of RDF are fundamentally incompatible (Nilsson, Palmér, Brase, 2003). This translation therefore cannot specify how to interpret general RDF descriptions, other than those which come from LOM, in terms of the LOM abstract model. Conversely, the LOM RDF binding cannot specify how to translate extensions of LOM into

RDF, as each of these extensions must be analyzed individually in order to determine how to represent them in RDF.

The same incompatibility exists between any two metadata standards where one is based on an elements-in-elements model and the other is based on a property-value model, for example MODS and Dublin Core.

The result can be summarized in the following table:

Format	*Processable by LOM application*	*Processable by Dublin Core application*	*Processable by RDF application*
LOM+Dublin Core RDF	Only LOM part	Dublin Core part + most of LOM part	Dublin Core part + LOM part

Reusing "Elements" Across Metadata Standards

What we have seen in this chapter is that mixing different metadata standards in the XML format does not work the way we would want it to. Using RDF as a common format works well with standards that use an abstract model compatible with RDF, but is still problematic for LOM and other standards based on an elements-in-elements model.

The CORES Resolution (Baker and Dekkers, 2002), which has been signed by both the IEEE LTSC and the Dublin Core Metadata Initiative, encouraged the owners of metadata standards to assign URI references to their "elements", the "units of meaning comparable and mappable to elements of other standards", but it did not specify what "comparable and mappable" meant. As a consequence the owners of different standards assigned URI references to "elements" that are created within different abstract models and uses metadata formats that rely on those incompatible abstract models for their meaning and interpretation. The assignment of a URI reference to an "element" means that it can be unambiguously cited, but it does not change the nature of the "element": and it does not mean that it is meaningful to use a URI reference for a LOM element as, e.g., a property URI in a Dublin Core metadata description. Similar incompatibilities have been noted between, e.g., RDF and MPEG-7 (van Ossenbruggen, Nack and Hardman, 2004 and Nack, van Ossenbruggen and Hardman, 2005).

The conclusion we may draw from the analysis in this section, is that we must not confuse the components used in a metadata format and the constructs in the abstract model. The components in a metadata format, such as "element URIs" may seem to be similar and compatible, but in reality they belong to completely different frameworks that might not be compatible. There are several problematic scenarios:

- Mixing two metadata formats created to conform to different abstract models, such as Dublin Core XML and LOM XML. A similar example is trying to use parts of a Dublin Core RDF description serialized in the RDF/XML language together with elements from another XML language such as the LOM XML language. As LOM and RDF use incompatible abstract models, this also leads to nonsense metadata constructs (Johnston, 2005a).
- In general, reusing metadata terms or elements adhering to different abstract models, regardless of the metadata format used, such as reusing a Dublin Core element URI in a LOM metadata description. As we have seen, this leads to nonsensical metadata constructs, as the URIs of Dublin Core and of LOM must be interpreted in terms of different abstract

models.

- Mixing two different bindings of the *same* standard, when those two bindings apply *different* interpretations to the use of *similar* components in the metadata format. This is the case with the Dublin Core XML binding, which must be interpreted using a different set of rules than the RDF/XML serialization of the Dublin Core RDF binding, though they contain component parts that are confusingly similar.

So we must conclude that the notion of reusing "elements" between metadata standards and formats using incompatible abstract models is fundamentally flawed. While assigning URI references for the component parts of a metadata standard is clearly a worthwhile effort in other ways, this does not really address the fundamental issue when creating interoperable metadata standards, namely the compatibility of their respective abstract models.

In conclusion, we see that in order to reuse components of different standards in a machine-processable way, the following criteria must be met:

1. The components must be unambiguously identified, so that components from different sources can be clearly distinguished and their origins can be separated. This is addressed by the CORES resolution.
2. The components must adhere to compatible abstract models. There is currently no resolution to address this, although the Dublin Core – IEEE Memorandum of Understanding ("Memorandum", 2000) points in this direction.
3. A metadata format must be used that allows for consistent interpretation of the components with respect to their respective abstract models. This too is mentioned in the "Memorandum", but has yet to be realized.

Metadata Mappings

One solution that has been proposed for solving the incompatibilities between metadata standards is to produce *mappings* between them. Several such systems have been implemented (see for example Godby and Childress (2003)). Mappings are useful, but suffer from a set of major problems:

- Every mapping requires manual construction, defeating the goal of machine-processability. Such a mapping must also be actively maintained in order to continue to be useful.
- The differences in abstract models necessarily make mappings incomplete and sometimes ambiguous, leading to very imperfect interoperability. Mappings may be complex because they may have to operate not on stand-alone "elements" but on complex nested constructs.
- Each new metadata standard requires a new set of mappings to each other relevant standard, creating an astounding complexity. This can be somewhat relieved by mapping all standards to a common "base standard". But as we have seen, the notion of a common base standard for metadata standards with incompatible abstract models is very problematic.
- Realizing such mappings that are able to preserve not only the metadata constructs themselves but also their semantics (including refinements) seems to be impossible in principle in many cases.
- Mappings does not really solve the problem of *combining* parts from different standards, only that of translating between standards.

As described in Johnston (2005a) and Nilsson et al (2003), and exemplified by the LOM RDF binding, mapping between incompatible abstract models involves a complex re-modelling process, and it is not always possible to make the resulting mapping bi-directional.

A Way Forward

It seems clear, then, that in order to achieve interoperability between metadata standards we need to focus on the features of their respective abstract models. In particular, the abstract models of the standards will need to be compatible, so that information expressed using one standard will be available to applications using other standards.

The long-term solution is to go even further, to a *common* abstract model. Having all metadata standards expressed using a common abstract model would greatly increase interoperability in several ways. It would also create a natural separation between the specification of the structure of metadata descriptions and the declaration of metadata terms used within that structure, so that both LOM and Dublin Core would appear as metadata vocabularies within that one structure.

There are already initiatives to develop a common abstract model that covers both LOM and Dublin Core, but unfortunately it seems to be impossible to arrive at such a model without re-engineering at least one standard to retrofit it to the new abstract model, which naturally is a major undertaking. But it seems clear that this is the only long-term solution to the interoperability problems we have seen here. Reaching out to embrace the other important metadata standards, such as MODS, MPEG-7 and the IMS set of standards is then the logical next step. In addition, great care must be taken to ensure that such an abstract model does not conflict with the emerging metadata format for the Web: RDF, which we will describe in more detail below.

Application Profiles: Mixing and Matching of Metadata Vocabularies

We now turn to an area where interoperability between metadata standards matters in a concrete way. In order to support community-specific and regional needs, metadata standards generally support a notion of customisation through *application profiles*. Enabling such customisations of metadata standards are one of the ultimate goals of metadata interoperability as we have described it in this chapter. In this section we will describe how application profiles rely on the interoperability features of the respective metadata standards, and the importance of interoperability between metadata vocabularies.

The metadata standards we have discussed use slightly different notions of application profiles. Combined with the differences in abstract models we have discussed previously, this produces significant hurdles for the interoperability that application profiles have been designed to solve.

However, we will see that these different approaches to application profiles to a large extent depend on the differences in abstract models, and that solving the abstract model issue paves the way for a merge into a single approach to application profiles, leading to a marked increase in metadata interoperability.

Metadata Standards and Profiling

The community that develops and uses a metadata standard is rarely completely homogeneous. It is common that in order to be useful to a community of reasonable size, a metadata standard incorporates some degree of flexibility. The developers of services that make use of that standard

take advantage of this flexibility to customise the standard to meet the specific requirements of their service and its audience.

In some cases, such customisation may involve selecting some subset of the full descriptive capability provided by a rich or expressive metadata standard, on the basis that not all of the functions supported by the standard are required in the context of a particular service. In other cases it may involve enhancing the specificity of description to support some particular requirements of a targeted user community.

The term *profile* has been widely used to refer to a document that describes how standards or specifications are deployed to support the requirements of a particular application, function, community or context, and the term *application profile* has recently been applied to describe this tailoring of metadata standards by their implementers.

The process of "profiling" a standard introduces the prospect of a tension between meeting the demands for efficiency, specificity and localisation within the context of a community or service on the one hand, and maintaining interoperability between communities and services on the other. Furthermore, different metadata standards may provide different levels of flexibility: some standards may be quite prescriptive and leave relatively few options for customisation; others may present a broad range of optional features which demand a considerable degree of selection and tailoring for implementation.

We also noted earlier that the development of the World Wide Web had had an impact on the use of metadata and on the development of metadata standards. One reflection of this changed environment is the development of metadata standards that are designed to support generic functions and to be applicable to a broad range of types of resource: the Dublin Core is an example of such a standard.

Another perhaps more subtle aspect is a growing recognition that it is desirable to be able to use community- or domain-specific metadata standards – or component parts of those standards – in combination. It should not be necessary to perform complex, costly and sometimes incomplete mapping of metadata each time resources or metadata move across community boundaries, particularly since, as noted above, new mappings must be designed each time a new community with a different standard joins the network of communication partners.

Rather, it is argued, the implementers of metadata standards should be able to assemble the components that they require for some particular set of functions - and if that means drawing on components that are specified within different metadata standards, that should be possible – safe in the knowledge that the assembled whole can be interpreted correctly by independently designed applications. Duval et al (2002) employ the metaphor of the Lego set to describe this process: an application designer should be able to "snap together" selected "building blocks" drawn from the "kits" provided by different metadata standards to build the construction that meets their requirements, even if the kits that provide those blocks were created quite independently.

Another motivating factor in this approach is the pragmatic desire on the part of the developers of metadata applications to make use of existing work and reduce redundant duplication of effort. If an implementer of metadata standard A has developed a component - say, a classification scheme or controlled vocabulary - which another implementer using metadata standard B regards as useful within their application, they should be able to "reuse" that existing component easily. And further, applications processing the metadata descriptions from the two sources should be able to establish that those reused terms are indeed the same terms.

Heery and Patel (2000) present a compelling vision of metadata implementers "mixing and matching" "data elements", constructing application profiles by selecting from the sets of "data elements" provided by metadata standards and by other implementers.

In the cases of both the Dublin Core and LOM metadata standards, standards developers and implementers recognise the application profile as a mechanism for realising the goals of metadata modularity, extensibility and refinement. Both communities have developed some guidance for the creation of such application profiles, which offer at least some measure of the mixing and matching capability outlined by Heery and Patel (2000). See also "Dublin Core Application Profile Guidelines" (2003), Baker (2003), Duval and Hodgins (2003) and IMS Global Learning Consortium (2000).

As we have argued, the extent to which the DC and LOM standards meet their ambitious goals of extensibility and modularity, and the form in which that extensibility and modularity are implemented, is determined by features of the different abstract models underlying the standards. And indeed this fundamental dependency is reflected in the fact that the two communities present different approaches to the metadata application profile. In both cases, an application profile enumerates the set of terms that may be referenced in some set of metadata descriptions, and provides some, perhaps context-specific, information about how those terms are to be used. Beneath that general similarity, however, lie some significant differences.

Dublin Core Application Profiles

In a Dublin Core application profile, the terms referenced are, as one would expect, terms of the type described by the Dublin Core Abstract Model, i.e. a Dublin Core application profile describes, for some class of metadata descriptions, which properties are referenced in statements and how the use of those properties may be constrained by, for example, specifying the use of vocabulary and syntax encoding schemes. The DC notion of the application profile imposes no limitations on whether those properties or encoding schemes are defined and managed by DCMI or by some agency: the key requirement is that the terms referred to in a DC application profile are compatible with the DC Abstract Model.

It is a condition of that abstract model that all references to terms in a DC metadata description are made in the form of URIs. The URI is a global identifier system. As long as the owner of a URI adopts policies which guarantee the persistence of the URIs they assign - i.e. they provide assurances that once a URI is assigned to a metadata term, it will continue to identify that metadata term and will not be used for another resource - the requirement for unambiguous identification of terms is met. Terms can be drawn from any source, and references to those terms can be made without ambiguity.

This set of terms can be regarded as the "vocabulary" of the application or community that the application profile is designed to support. The terms within that vocabulary may also be deployed within the vocabularies of many other DC application profiles.

In addition to specifying what set of terms is to be used in their metadata descriptions, the developers of a metadata application – in most cases at least – also need to specify how their metadata descriptions are to be expressed for exchange between systems, i.e., they need to specify the use of one or more formats for their metadata records. We have already noted that Dublin Core provides a number of binding specifications which describe how to encode DC metadata in a number of formats, and typically the application developer will select one of these bindings.

Two examples of widely used Dublin Core application profiles are the OAI-DC and RDN-DC application profiles, which we will now describe in more detail.

The OAI-DC application profile

The Dublin Core metadata standard has been widely implemented by services that make use of the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) (Lagoze, Van de Sompel, Nelson and Warner, 2002). The OAI-PMH is a fairly simple protocol that supports the controlled transfer of metadata records over HTTP. The protocol allows the exchange of any metadata that can be serialised in an XML format. As a baseline metadata standard, the OAI-PMH specification requires that all OAI-PMH data providers must support the OAI DC application profile.

In this profile, a metadata description must consist of statements which reference only the fifteen properties of the Dublin Core Metadata Element Set. Properties are optional and repeatable, i.e., there is no requirement that all properties are referenced from statements in a metadata description, and the same property may be referenced in multiple statements. References to values must be made in the form of value strings, and neither vocabulary encoding schemes nor syntax encoding schemes may be used.

The RDN-DC application profile

The Resource Discovery Network (RDN) is a collaborative service provided for the UK Further and Higher Education communities which provides access to high quality Internet resources selected by subject specialists for their value in learning and teaching. The RDN makes use of OAI-PMH to transfer metadata records between partners, but rather than exchanging only OAI-DC records, the RDN deploys its own application profile, RDN-DC, which supports the creation of more expressive metadata descriptions tailored for the discovery requirements of the RDN (Day and Cliff, 2003, Powell, 2003). The profile references a subset of the properties provided by Dublin Core and requires the use of specific vocabulary encoding schemes for some of those properties; it also references some properties that were defined specifically for the requirements of the application.

Those local properties are defined and assigned URIs by the RDN in much the same way as the standard properties provided by the Dublin Core metadata standard and they are referenced in a metadata description, using a URI, in exactly the same way as a property provided by the standard. And indeed although these properties were defined to meet the requirements of one particular community, they may be referenced by the developers of other DC application profiles developing applications for other communities if their usage is perceived as meeting some functional requirement.

LOM Application Profiles

An examination of LOM application profiles reveals a slightly different approach. Instead of mixing and matching elements from multiple schemas and namespaces (Heery and Patel, 2000), it presents customisation of a single standard to address the specific needs of "particular communities of implementers with common applications requirements" (Friesen, Mason and Ward, 2002).

That is, a LOM application profile is designed within the framework of the LOM abstract model. The terms referenced within a LOM application profile are terms of the type described by the LOM abstract model. A LOM application profile describes how the hierarchical structure

described by the LOM standard is adapted to the requirements of an application – and indeed the nature of that adaptation is itself constrained by the LOM standard, which specifies data types and value spaces for each LOM data element and places some limits on the occurrences of LOM data elements within a LOM metadata description. This contrast between the scope of the LOM and Dublin Core metadata standards was noted earlier: while the Dublin Core standard specifies a set of terms for used in metadata descriptions, it adopts a flexible approach to the ways in which those terms are deployed by an application. The LOM standard, on the other hand, both provides a set of data elements and defines a structural pattern of nested elements, with ordering and cardinality constraints, within which those data elements are deployed and interpreted. This set of standard structural constraints might be conceptualised as a "default" or "base" LOM application profile, one to which all other LOM application profiles must conform.

The most widely used mechanism for extending the LOM metadata standard is through the use of custom vocabularies to provide values for LOM data elements and the use of specified taxonomies within the LOM Classification element. Although the LOM abstract model does not require the use of globally unique identifiers for vocabularies and taxonomies, there are mechanisms provided (the "Source" sub-element within a Vocabulary data type item, and the "Source" element of the Classification category) which enable implementers to adopt conventions to distinguish between vocabularies, and to confirm that two references are indeed references to the same vocabulary.

Another common method of customizing LOM is through the tightening of structural constraints, such as making elements mandatory or to remove elements altogether, or putting an upper limit on the number of instances of a certain element. It is also common to produce additional guidelines for the usage of specific elements within the target community, something which is of particular interest for national customizations of LOM such as the UK LOM Core.

The LOM abstract model provides further possibilities for extensibility through the use of what it calls "extended data elements", i.e. the use within a LOM metadata description of data elements other than those defined by the LOM standard itself.

Three widely used LOM application profiles are the UK LOM Core, the RDN-LTSN LOM Application Profile and the Curriculum Online Metadata Schema, which we will now describe in more detail. The first two of these also demonstrate how one more generic application profile (the UK LOM Core) can form the basis for a second, more refined application profile (the RDN-LTSN LOM Application Profile).

UK LOM Core

The UK LOM Core LOM application profile is the result of efforts to promote common practice in the implementation of the LOM in UK educational contexts, in order to improve the ability of LOM metadata applications to exchange effectively the information required to support a number of basic functions (UK LOM Core, 2005).

The UK LOM Core:

- specifies a "core" set of LOM data elements that should be present in LOM metadata instances
- provides information on the use and interpretation of LOM data elements within the UK context
- specifies a small set of vocabularies that should be used to provide values for some LOM

data elements

RDN-LTSN LOM Application Profile

As noted above, the Resource Discovery Network (RDN) provides a Dublin Core application profile for metadata sharing between partners in the network. The RDN has also engaged in collaborative work with a similar network, the Learning and Teaching Support Network (LTSN) (since 2004 a part of the UK Higher Education Academy). Metadata sharing within this broader network was based on the use of a LOM application profile known as the RDN/LTSN LOM Application Profile (RLLOMAP) (Powell, 2005).

RLLOMAP is designed to support a specific set of functions to be delivered by the RDN-LTSN services. However, it is also designed to be compliant with the UK LOM Core. i.e., any LOM metadata description constructed according to RLLOMAP also complies to UK LOM Core. RLLOMAP specifies a set of LOM data elements and provides quite detailed guidelines for their use in the context of the RDN-LTSN community. It also mandates the use of some community-specific vocabularies (in addition to the LOM standard vocabularies) for some elements, and makes recommendations for the use of specified taxonomies for the LOM Classification element.

Curriculum Online Metadata Schema

The Curriculum Online service provides access to multimedia resources which support the curriculum taught in primary and secondary schools in England, and a metadata schema - an application profile of the LOM - was developed to support the specific requirements of this service. In particular, the schema supports the controlled classification of learning resources required to enable the rich searching and browsing functions that are provided to teachers and other users of the Curriculum Online web site (Department for Education and Skills, Simulacra and Schemeta, 2003a, 2003b).

Like RLLOMAP, the Curriculum Online Metadata Schema specifies which elements are required to occur in metadata descriptions and provides guidelines for providing values for those elements.

In addition, it defines some extensions to the LOM standard in the form of some additional data elements and vocabularies for the values of some of these elements. These “extended data elements” include a group of elements to support the description of the “Method of Delivery” of the resource, a group of elements that provide an indication of the cost of a resource, and an element to capture the name of the application used to create the metadata record.

Application Profiles and vocabularies

As can be discerned from the above discussion, the notion of a metadata “vocabulary” is somewhat ambiguous and is used differently in the two standards. In LOM, a vocabulary is a set of tokens with a specified “source” that can be used as values for certain elements. For example, the LOM element “Educational.Difficulty” can be used with values from a vocabulary specified in LOM, and containing the tokens “very low”, “low”, “medium”, “high” and “very high”. The “Source” must then be set to “LOMv1.0”, to indicate that the values are from the LOM standard itself.

In Dublin Core, a vocabulary can be one of two things:

1. A set of concepts as specified by a vocabulary encoding scheme. For example, the “dcterms:LCSH” vocabulary encoding scheme refers to the vocabulary formed by the set of

Library of Congress subject headings.
This corresponds closely to the notion of vocabulary in LOM, with the subtle but notable difference that Dublin Core deals with the concepts themselves (that may be referenced using a value string or a value URI, depending on the application), while LOM deals only with vocabulary tokens, i.e., opaque strings.

2. A set of metadata properties together with their definitions. For example, the Dublin Core Element Set, consisting of the 15 original Dublin Core elements, is such a vocabulary.
The closest correspondence in LOM to this kind of vocabulary is the set of LOM elements.

We will use the term *value vocabulary* to denote the first kind of vocabulary, as the terms in such a vocabulary are used as values in metadata instances. The term *element vocabulary* will be used for the second kind, which signals the close relationship to the imprecise concept of metadata "elements" that we have already encountered. These terms are not in general use, but we have found it useful, for the purposes of this chapter, to distinguish between the two.

Element vocabularies and value vocabularies have fundamentally different characteristics. While value vocabularies are used to construct taxonomies and thesauri that describe relationships between concepts in terms of broader/narrower, containment etc, element vocabularies are used to construct schemas and ontologies that describe how metadata instances are to be constructed.

As noted above, both standards have a notion of value vocabularies that include a notion of "vocabulary source". When specifying a value of a LOM element of the type "Vocabulary", the value may be accompanied with a "Source" string that gives an indication of the origin of the value, and therefore its interpretation. Similarly, Dublin Core uses the concept of vocabulary encoding schemes to specify the origin of a value, which may also be identified using a value URI. Being able to specify the source of a vocabulary is a requirement for interoperable metadata descriptions, and an important prerequisite for modular application profiles.

When it comes to element vocabularies, the situation is less clear. In Dublin Core, terms in element vocabularies, i.e., properties, must be assigned a URI to be usable in Dublin Core metadata descriptions. In this way, Dublin Core enables application profiles to mix Dublin Core properties with other properties in a controlled fashion, as the URI will allow applications to disambiguate between properties from different sources that are used in the same application profile.

However, the data elements defined by the LOM standard, as well as extended elements, are referenced not by globally unique identifiers, but by short human-readable labels like "Identifier" and "Context" (or "General.Identifier" and "Educational.Context", if their category is taken into account). There is an implicit assumption that a human reader or an application reading or processing a LOM metadata description will be able to determine from some contextual information that the data element is that data element defined by the LOM standard.
Perhaps for this reason the term "LOM application profile" appears to have been applied principally, though not exclusively, to those descriptions of LOM implementation that are limited to the data elements specified by the LOM standard, with extensibility restricted to the specification of value vocabularies and taxonomies. Where extended data elements are used in LOM application profiles, the implementer assigns labels to distinguish their data element names from those used for data elements defined by the LOM standard and in other LOM application profiles – but since these are simply arbitrarily chosen labels, rather than identifiers assigned with an identifier scheme, they can not be guaranteed to be unique. For this reason, LOM lacks support for machine-processable reuse of element vocabularies across application profiles.

The situation is aggravated by the fact that the LOM XML binding *does* provide namespace URIs for both the LOM elements and for elements used in extensions to LOM. But as these URIs are not part of the LOM abstract model, they cannot be used outside the LOM XML binding to refer to the relevant LOM element.

Application Profiles and Bindings

The developer of a metadata application – in most cases at least – also needs to specify how metadata descriptions constructed according to their profile are to be expressed when they are exposed for exchange between systems, i.e. they need to specify the use of one or more formats for their metadata records. The developer will probably select one of the bindings specified by the metadata standard. In some cases they may develop a new binding to meet some particular requirements of their context (as is proposed by The International Press Telecommunications Council (2005)). Where an application profile developer develops a new binding, they may choose to optimise that binding for the context of their application, e.g. by supporting only some subset of the constructs in the full abstract model of the standard. In any case, if a new binding is developed it is essential that the developer makes available a description of how the syntactic features they use are to be interpreted in terms of the standard's abstract model. They may choose to provide an algorithm or transformation by which a record conforming to their binding can be converted into a record using a standard binding.

One promising framework for this kind of transformation specifically into RDF that is becoming increasingly popular is GRRDL, described in Hazaël-Massieux and Connolly (2005) as "a mechanism for Gleaning Resource Descriptions from Dialects of Languages; that is, for getting RDF data out of XML and XHTML documents using explicitly associated transformation algorithms, typically represented in XSLT".

The Limitations of Mix and Match in DC and LOM Application Profiles

The first point that we have highlighted is that the DC and LOM concepts of the application profile are both rooted in the corresponding abstract models underpinning those standards. A Dublin Core application profile refers to properties, vocabulary encoding schemes and syntax encoding schemes; a LOM application profile refers to LOM data elements or extended data elements and their value spaces, using the range of datatypes specified by the LOM standard. As has already been discussed these are fundamentally different types of construct: an occurrence of a LOM data element is interpreted through the semantics of the LOM abstract model, and a reference to a property is interpreted through the semantics of the DC abstract model. Neither approach is sufficient to support the Lego-like assembly of a modular metadata description which draws on both the LOM and DC metadata standards.

Secondly, the LOM standard provides not only a set of data elements, but also a default pattern for the use of those data elements, a "base" application profile to which other community- or application-specific LOM application profiles should also conform.

Closely related to this second point is that the LOM abstract model does not define a mechanism for uniquely identifying and referencing data elements within a global context. While the use of extended data elements is possible, the disambiguation of those elements is reliably possible only within a context where the use of names is controlled. The LOM abstract model does not lend itself to the reuse of data elements within a global context, or to the sharing of LOM metadata descriptions beyond a context in which names are controlled.

The DC and LOM application profile constructs are both useful in formalising the way in which

the implementers of metadata standards customise and (to a greater or lesser degree) extend those standards. They also provide a basis for disclosing existing work and encouraging the reuse of components used within existing application profiles, again subject to some limitations. They highlight that a degree of mixing and matching is indeed possible – but only within the framework of the corresponding abstract model. For DC and LOM, the incompatibility of those abstract models means that the application profile construct is not sufficient to address the problem of how to use component parts of those two standards in combination.

Refinement and Metadata Semantics

The word *semantics* in the context of metadata is closely linked to the notion of machine-processability. Semantics usually refers to the assignment of *meaning* to an otherwise meaningless syntax. The need for *human* semantics in metadata is clear – metadata is after all used for encoding information for human consumption. But what are the benefits of *machine* semantics in metadata standards?

The benefits are of several distinct kinds. Semantics are the underpinnings of abstract models for metadata, which we have already seen to be fundamental in metadata interoperability. Semantics provide avenues for automatic discovery of the meaning of metadata expressions, thus allowing metadata applications to partially understand metadata extensions encountered in previously unknown application profiles. Formal semantics provide the foundation for processing metadata in software agents and ontology-based reasoning systems, which provide the basis on which to build machine-processable mappings between semantically overlapping standards.

The Role of Refinements in Dublin Core and LOM

The Dublin Core abstract model provides two basic primitives for the expression of metadata semantics: sub-properties and sub-classes. Both primitives are used to specify so-called *refinements*, that serve the important purpose of allowing more fine-grained descriptions to be understood by applications that only know how to process more coarse-grained descriptions.

Suppose we declare the property “ex:illustrator” to be a sub-property of the Dublin Core element “dc:contributor”. Applications that know the difference between “dc:contributor” and “ex:illustrator” may use the values of the two properties in subtly different ways that are appropriate to the situation. However, an application that does not know how to process the “ex:illustrator” property may still choose to process the value of that property in the exact same way that it would process a value of the “dc:contributor” property. Thus, a resource with an “ex:illustrator” of “Gary Chalk” may be said to simultaneously have an implicit “dc:contributor” of “Gary Chalk”. The formal word for this process of implicit and automatic “creation” of property values is *entailment.*

Note that the process of entailment is mandatory in the sense that it is considered invalid to specify a value of the “ex:illustrator” property that is not at the same time a valid value for “dc:contributor”. This must of course be reflected in the definition of the sub-property: if not all valid values of the sub-property are also valid values of the property, the sub-property definition is invalid. For example, while the values of an “ex:owner” property are sometimes also valid values of “dc:contributor” (as owners sometimes also participate in the creation of a resource), this is not *always* the case. Thus, “ex:owner” cannot be declared a sub-property of “dc:contributor”. The details of how to define refinements and some of their consequences are given in Johnston (2005b).

The other kind of refinement, sub-classes, is used together with the specification of the type of a

resource using the “dc:type” property. For example, the type “dctype:StillImage” is a sub-class of “dctype:Image”. Sub-classing simply means that everything that is of the type “dctype:StillImage” is simultaneously of the type “dctype:Image”. This allows for a fine-grained specification of resource types, while allowing for interoperability with less capable applications.

The process of simplifying metadata records based on refinements is sometimes referred to as *dumb-down*, as it can be used to construct a less refined, but more widely processable metadata record. It can be performed by the application itself, or in a pre-processing step.

LOM does not have a corresponding notion of refinement. In fact, the LOM standard states that “extended data elements should not replace data elements in the LOM structure”. The reason is in part that there is no machine-processable way to specify that a LOM extension refines a LOM element. Therefore, an application would not be able to recognize that an extended LOM element can be processed in the same way as the LOM element it replaces, or dumbed-down to the original LOM element.

Formal and Informal Semantics

Returning again to our metadata format examples, let us try to understand how an application arrives at an understanding of the different metadata expressions.

When processing the LOM XML example in Example 1, an application will first need to know what XML language is being used, as the XML document itself generally does not specify that information. So, given that we know that our data is given in the LOM XML format, the interpretation of each XML element is given by the LOM XML binding – a “description” XML element within an “educational” element must be interpreted as the “Description” LOM element in the LOM category called “Educational”. The LOM standard itself specifies the human semantics of this element: “Comments on how this learning object is to be used”.

Note that in this process, the interpretation must be performed by reference to the published LOM standards. Any machine processing must be manually tailored to each and every element of the metadata structure. This is an example of *informal* semantics, or semantics that is explicit, but not machine-processable.

Let us contrast the previous example with the RDF example from Dublin Core in Figure 1. An RDF application will process the RDF metadata and find an RDF property named “dc:format”. An application can use the URI of the property to obtain a description of the property provided by the authority that defines it (the Dublin Core Metadata Initiative), using the RDF Schema language. That description includes human-readable information about the property, and also machine-processable data describing its relationships to other resources, including refinement relationships with other properties.

The value of the property, “text/html”, is seen by the application to be an instance of “dcterms:IMT”. The Dublin Core RDF Schema provides human-readable information to indicate that this class is the set of all Internet Media Types, or MIME types; it also provides machine-processable data describing the relationship of this class to other resources.

The fact that “dc:format” is a property and “text/html” is an instance of the class “dcterms:IMT”, and further information based on the descriptions of that property and that class, can be inferred with no human intervention.

What we find here is an example of *formal* semantics, where an application can automatically process the metadata structure to arrive at a partial understanding of the metadata. If the metadata

includes properties that refine other properties, these refinements can also be processed automatically, for example in order to perform a dumb-down of the metadata record.

Note that the application does not need to know what metadata standard it is processing, but only needs access to the corresponding machine-processable RDF schemas that describe the element and value vocabularies used in the description. This points to a major difference between XML-based languages and RDF: XML-based languages provide their own, often incompatible semantics. XML specifications such as XML Schema are limited to capturing syntactic features of XML languages, and cannot describe their semantics. On the other hand, RDF provides a basic framework for metadata semantics that all standards expressed in RDF conform to. The formal semantics of RDF is specified in Hayes (2004), and the semantics of RDF metadata can be expressed using the RDF schema language (Brickley and Guha, 2004). Dublin Core has chosen to use RDF Schema as a way to express the formal, machine-processable semantics of the Dublin Core properties and encoding schemes, for use also in metadata formats other than RDF.

An interesting discussion of different kinds of metadata semantics can be found in Uschold and Gruninger (2002). The approach to metadata found in the RDF set of standards has many intriguing features that might serve as a source of inspiration for future learning object metadata standards, so we now turn to a short introduction to RDF and the Semantic Web.

RDF and the Semantic Web

RDF has been created to enable the vision of the "Semantic Web" – a web of machine-processable information, extending the current web. RDF tries to reach this goal by:

- Using a coherent framework based on URIs for identification of metadata elements such as properties, classes and resources. RDF is perhaps best described as a "semantizisable" web, which provides a sufficiently coherent metadata framework that its component parts can be given proper formal semantics without inconsistencies or ambiguities.
- providing a basic abstract model for metadata, with certain built-in semantics. This basic model allows applications to store and process metadata from different standards in a common framework.
- being extensible, both structurally and semantically. We have already seen examples of semantic extensions in the form of refinements, as well as proof of the straghtforwardness of structural extensions when combining several metadata standards.
- being web-capable, unlike traditional databases and knowledge representation systems. While the RDF model is based on previous work on knowledge representation systems, it differs substantially in that it integrates with WWW standards such as XML and URIs.
- being decoupled from the information it describes. In RDF, anyone can express any statements about any resource. It is up to the application to determine trustworthy sources. This allows for multiple descriptions, appropriate for different contexts, of a single resource to co-exist.
- allowing for self-describing metadata. Thanks to its machine semantics, RDF applications can partially process new metadata without previous knowledge of the standards involved.

The RDF standard (Klyne and Carroll, 2004, Manola and Miller, 2004) is by its very nature a semantic standard. In RDF, the tokens used in the format do not merely identify syntactic elements, but by design refer to notions in the real world. By contrast, XML elements are by themselves only syntactic placeholders that need the semantics of an XML language to be given

meaning (Cover, 1998). Similarly, the statements expressed in RDF are not just data structures, such as is the case with XML document trees, but have real-world meanings. Every RDF statement has a real-world interpretation, independently of any other RDF statement. RDF can therefore be described as a framework for extension and recombination of independent statements about the world of resources.

The Semantic Web is a visionary project initiated by the W3C with the stated purpose of realizing the idea of having data on the Web defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration and reuse of data across various applications.

It was motivated by the very same problems that motivates the development of metadata standards: the fact that raw media, in the form of text, HTML, images or video streams, contains meta-information that may be readily deducible from the context for the human consumer (the name of the author, the kind of material contained within, etc.), but is mostly inaccessible to computers. Making this information available to computers in order to enable a whole new class of semantics-aware applications, was the driving vision that created the Semantic Web project.

Most traditional metadata approaches take the view of metadata as being mostly a digital indexing scheme to use in cataloguing and digital libraries. What distinguishes the Semantic Web from these approaches to metadata are two important things:

- The Semantic Web is designed to allow reasoning and inference capabilities to be added to the pure descriptions. This includes stating simple facts such as "a hex-head bolt is a type of machine bolt", but extends to the inference of new relationships from known data. This is an important feature to allow intelligent agents and other software to not only passively consume descriptions, but to act on them as well.
- The Semantic Web is a web-technology that lives on top of the existing web, by adding machine-readable information without modifying the existing Web. It is designed to be globally distributed with all that this implies in terms of scalability and flexibility.

The Semantic Web is a layered structure. XML forms the basis, being the standardized transport format. RDF provides the information representation framework, and on top of this layer, schemas and ontologies provide the logical apparatus necessary for the expression of vocabularies and for enabling intelligent processing of information.

This includes the definition of semantic mappings between overlapping metadata standards. As the metadata constructs are based on a common abstract model, the complexity of mappings and the level of precision in mappings are dramatically increased in comparison to mappings between standards using different abstract models (Uschold and Gruninger, 2002).

Vocabularies, RDF Schemas and Ontologies

Using RDF Schema, the semantics and properties of both element vocabularies and value vocabularies can be expressed in a common framework. For example, Dublin Core provides one element vocabulary, and the LOM RDF binding provides another. RDF schema allows for the description of relationships between terms not only within one single standard, but also across standards. It also allows for description of any number of attributes of the vocabulary terms themselves, using any RDF properties. For example, the Dublin Core term "dcterms:abstract" is described by the Dublin Core RDF schema as depicted in Figure 10.

RDF Schema contains a base semantics that is used in practically all RDF descriptions, and that

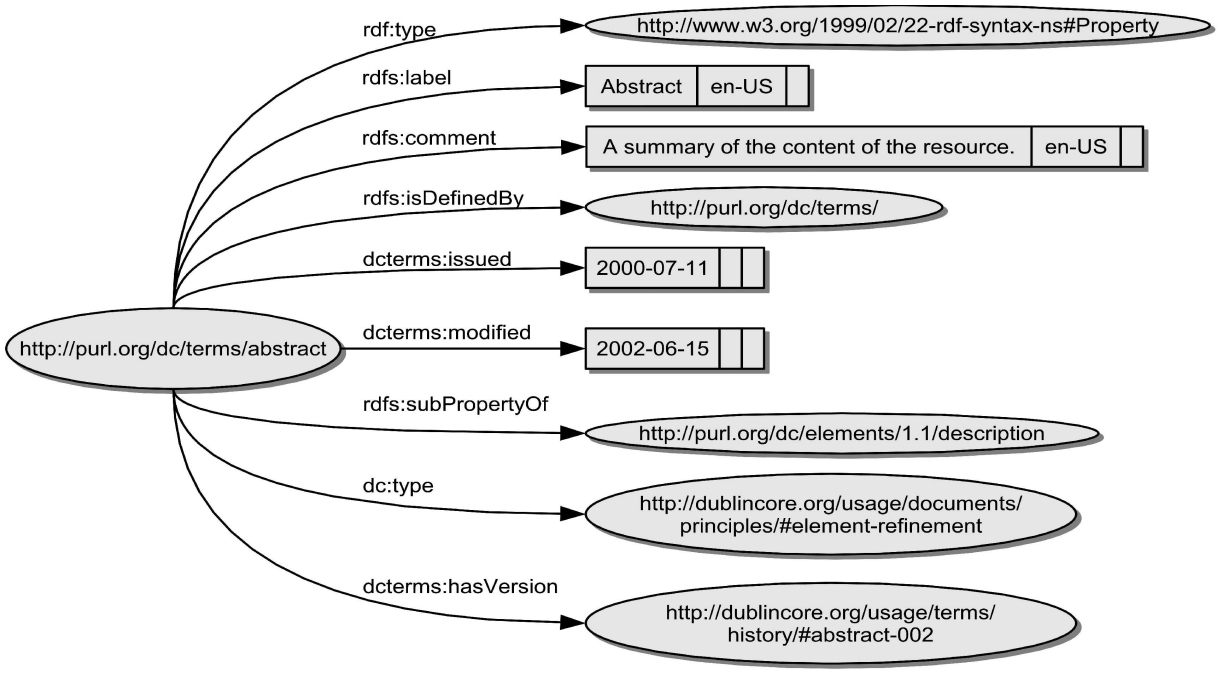

Figure 10. The RDF schema description of the Dublin Core term "dcterms:abstract".

encompasses both property refinement and sub-classing. The following table gives some examples of what can be expressed in RDF Schema, and using what construct.

In order to express	***Use this construct***
This resource **is a** Person	rdf:type
Student is **a kind of** Person	rdfs:subClassOf
"creator" is a **Property**	rdf:Property
"hasBirthday" **can only be used to describe** a Person	rdfs:domain

Another promising RDF-based framework for defining value vocabularies, especially in the form of hierarchical taxonomies or thesauri is SKOS, Simple Knowledge Organization System (Miles and Brickley, 2005).

For more advanced semantics, *ontologies* using the Web Ontology Language OWL, provide a foundation for expressing complete conceptual models of a domain, allowing for a dramatically higher level of automation that allows computer systems to operate at a conceptual level much closer to the human level. As described in Heflin (2004), OWL can express that the Person and Car classes are disjoint, or that a string quartet has exactly four musicians as members, something that RDF Schema cannot do.

Another important benefit of ontologies is that they allow for the automatic deduction of additional information about resources based on existing information. For example, if the metadata of a certain learning object states that it requires support for a specific set of standards, such as CSS2 and XHTML, and it is separately known which web browsers support those standards, an inference engine can infer that a certain browser works with that learning object

without being explicitly told so. In the same way, ontologies provide support for semantic mappings between vocabularies that partially overlap, so that users may ask questions in terms of one vocabulary and receive answers that are described using a separate vocabulary.

From this short introduction to the notion of metadata semantics, we can conclude that a metadata framework built on the foundation of a solid abstract model, with support for machine-processable semantics, is a desirable goal for future learning object metadata standards. The metadata standards in current use go some way towards the fulfilment of this goal, but they operate mostly in isolation from each other, and one important component is missing: a metadata standardization framework. Throughout this chapter, we have gathered enough requirements to be able to put together a vision of such a framework.

Towards an Interoperability Framework for Metadata Standards

If we try to look forward into the future of learning object metadata standards, it seems clear that an improved approach to metadata standardization is needed in order to fulfil the metadata interoperability requirements we set forth early in this chapter.

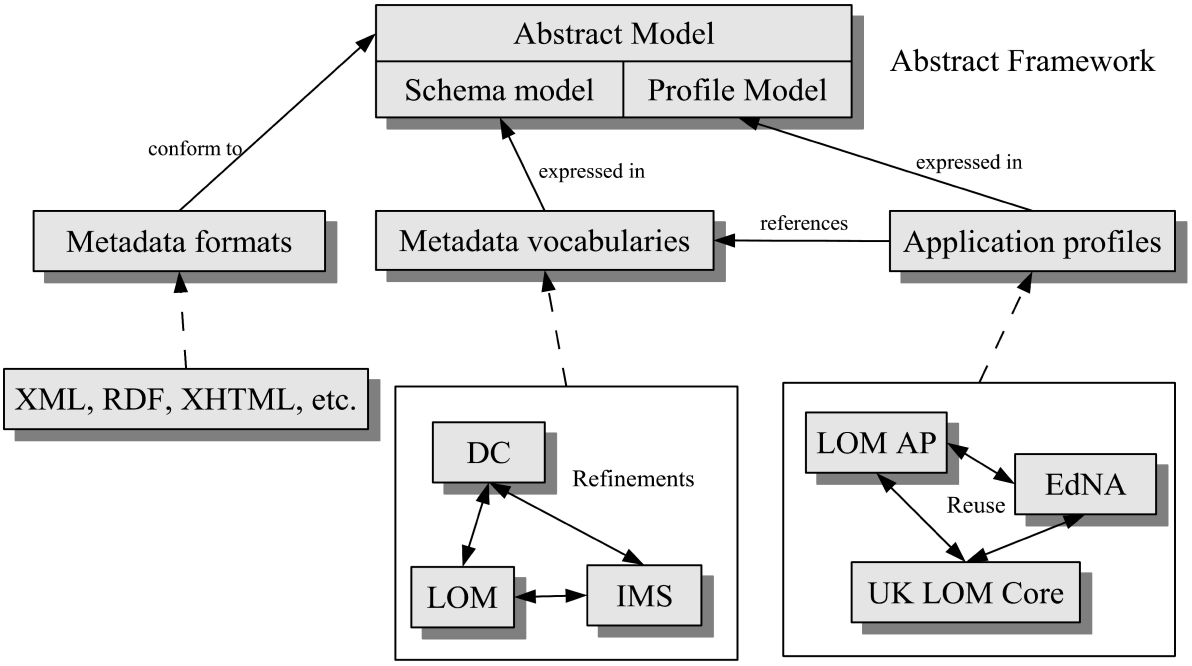

Figure 11. A possible structure of a future metadata standardization framework.

We have already seen examples of the four central components that are needed to create such a framework. The framework rests on the basis of an *abstract model* for metadata. A *schema language* allows for the definition of metadata element and value vocabularies that fill the abstract model with metadata terms and relationships between terms. A *framework* for definition of machine-processable *application profiles* use the abstract model to constrain vocabularies to fit the requirements of specific communities. Finally, *metadata formats* encode metadata descriptions using any combination of vocabularies and application profiles.

Thus, the current use of the term “metadata standard” or “metadata schema” will need refinement, resulting in at least four different kinds of metadata standards:

- The overarching abstract model standard. This will also include a specification for how to express the semantics of vocabularies adhering to the abstract model as well as a specification for how to express application profiles in a machine-processable way.
- Metadata format specifications. These will include bindings of the abstract model to a set of formats and systems, including XML, RDF, database layouts, programming languages, etc., and will replace current format bindings for the different metadata standards.
- Metadata vocabularies. These will include metadata terms from different communities. The Dublin Core terms, the LOM elements and so on are examples of metadata element vocabularies, and a large set of value vocabularies also fit into this category.
- Application profiles. These will specify usages of metadata vocabularies in complex combinations. As we have noted, the LOM standard contains a basic application profile, and this aspect of LOM will be separated from the definition of the element vocabulary consisting of the LOM elements.

Common Abstract Model

The basis of the envisioned learning object metadata standardization framework is the abstract model. As we have seen, the incompatibilities of abstract models are the most significant stumbling blocks for metadata interoperability. The development of a common abstract model for metadata is therefore of central importance if we are ever going to experience true metadata interoperability.

Developing such an abstract model is a major undertaking, not so much because of the technical difficulties, but because of the lack of coordination between the major standardization organizations involved. Still, the process is necessary and will give a number of tangible benefits, including:

- Clear guidelines on how to create and maintain customized metadata vocabularies. There is currently some confusion on how to best produce vocabularies, much due to the differing fundamental principles for vocabularies in the different metadata standards.
- Fine-grained control over relationships between terms from different standards, including refinement and partial mappings. Automation of interoperable metadata management will be greatly improved, and metadata vocabularies will be able to build upon each other.
- A single set of format bindings. Contrast this with the current situation, which requires every metadata standard to have its own set of format bindings. This will make life easier not only for metadata standardization bodies, but also for applications that will only need to support one format.
- A single framework for extending and combining metadata from different standards. This will enable standardized principles for the construction of interoperable application profiles.
- A single storage and query model for very different types of data and schemas. For example, storing metadata from different specifications in the same database is straightforward. Implementing searching that includes dependencies between metadata expressed in different schemas is simplified.

Thus, the development of a common abstract model leads the way towards support for all our metadata interoperability principles: extensibility, modularity, refinement and machine-processability.

Interoperable Vocabularies

In a metadata standardization framework supported by a common abstract model, the work of defining new metadata terms is much reduced. As the "grammatical structure" of metadata descriptions is already laid down, the only thing needed is to fill the abstract model with specific terms. In order to do so, we need a language for describing metadata vocabularies. We will call this a *schema language*. RDF Schema is one such language.

The main benefit of developing vocabularies in a common framework is that reuse across standards will be much simpler. As an example, many elements in the LOM standard are not specific to learning, and have similar counterparts in other standards. In a common framework, the LOM elements will be made into a fully-fledged element vocabulary capable of being extended, refined and semantically annotated. The semantic relationships to terms in these other standards can be made explicit and machine-processable.

One interesting consequence of a common element and value vocabulary framework is the possibility of unexpected collaboration. That is, as others specify relationships to your vocabulary, new relations between resources will start to appear, and you will be able to process metadata that you have not explicitly declared semantic relationships to.

Modular Application Profiles

A common framework for expressing application profiles will be a necessary building block for the construction of reusable application profiles. The framework must not be tied to a specific metadata format, but must operate at the level of the abstract model, so that the application profile can be reused in all metadata formats.

By separating the specification of application profiles from the declaration of metadata vocabularies, we will reach a partial solution to the differences between the LOM and Dublin Core approaches to application profiles. One reason for the LOM approach to application profiles of restricting the base standard seem to be that the LOM standard specifies four separate things in a single standard: an abstract model, an element vocabulary, a set of value vocabularies and a basic application profile. By using the metadata standardization approach proposed here, these components would be split into separate specifications, leading to a much higher incentive for mixing and matching, while still retaining all the advantages of the combined approach in terms of validation and conformance testing.

Promising work on machine-processable application profiles can be seen in, e.g., "Guidelines" (2005). There are also other initiatives for such frameworks, but none are yet in widespread use.

Implications

We have seen clear evidence that RDF family of specifications provide an abstract framework of the kind discussed here, including a vocabulary description framework (RDF Schema) and support for ontologies through OWL. However, it remains to be seen if using RDF will be acceptable as a foundation for the wide set of applications that use LOM, Dublin Core, the IMS standards, etc. RDF was designed primarily for the Web, while the metadata that are of concern to us are important for a wide range of systems that are not restricted to web-oriented systems.

Dublin Core has proven that simple metadata formats such as HTML meta tags are popular and useful, and have contributed immensely to the spread of metadata tagging. LOM has a relatively complex structure, but it is similar enough to the structure of XML documents to be simple to use. RDF provides no such simple syntax, and the apparently steep learning curve remains a major obstacle to the acceptance of RDF. For example, while several versions of the RSS news syndication format have tried to use RDF, new versions seem to always step away from RDF in favour of a more predictable XML approach.

This points to a general observation: in any given application it is always easier to devise a custom XML language with custom semantics than to use a complex metadata framework. The extra work involved in being compatible with such a metadata framework does not become evident until the amount of metadata interactions increases beyond a certain threshold. It seems more and more systems are reaching that threshold and are looking for such a framework.

On the other hand, if the RDF specifications are not reused for such a framework, there is a real risk of reinventing much of what has already been achieved within the Semantic Web. Dublin Core is one example, as its abstract model closely resembles that of RDF. Dublin Core on the one hand uses its own abstract model and metadata formats, while on the other hand relies on RDF Schema to specify the machine-processable semantics of its terms.

Will RDF be the future of learning object metadata standards? Only time will tell. But it seems certain that many of the features of RDF are destined to become part of learning object metadata standards, whatever form they will take.

Conclusion

We have demonstrated that true metadata interoperability is still, to a large extent, only a vision, and that metadata standards still live in relative isolation from each other. The modularity envisioned in application profiles is severely hampered by the differences in abstract models used by the different standards, and efforts to produce vocabularies often end up in the dead end of a single framework. In order to enable automated processing of metadata, including extensions and application profiles, the metadata will need to conform to formal metadata semantics.

To achieve this, there is a need for a radical restructuring of metadata standards, modularization of metadata vocabularies, and formalization of abstract frameworks. RDF and the Semantic Web provide an inspiring approach to metadata modelling: it remains to be seen whether that framework will be reusable for learning object metadata standards.

References

Baker, T. (2003), DCMI Usage Board Review of Application Profiles. Retrieved July 1 2005, from http://dublincore.org/usage/documents/profiles/

Baker, T. & Dekkers, M., (2002), CORES Standards Interoperability Forum Resolution on Metadata Element Identifiers. Retrieved July 1, 2005, from http://www.cores-eu.net/interoperability/cores-resolution/

Bearman, D., Miller, E., Rust, G., Trant, J. & Weibel, S. (1999), A Common Model to Support Interoperable Metadata, *D-Lib Magazine*, January 1999. Retrieved July 1, 2005, from http://www.dlib.org/dlib/january99/bearman/01bearman.html

Brickley, D. & Guha, R. V. (2004), RDF Vocabulary Description Language 1.0: RDF Schema, *W3C Recommendation 10 February 2004*. Retrieved July 1, 2005, from

http://www.w3.org/TR/rdf-schema/

Cover, R. (1998), XML and Semantic Transparency. Retrieved July 1, 2005, from http://www.oasis-open.org/cover/xmlAndSemantics.html

Day, M. & Cliff, P. (2003), RDN Cataloguing Guidelines, Version 1.1. Retrieved 1 July, 2005, from http://www.rdn.ac.uk/publications/cat-guide/

Department for Education and Skills (in association with Simulacra and Schemeta) (2003a), Curriculum Online: Metadata Guide for Tagging, Version 1.11. Retrieved 1 July, 2005, from http://www.curriculumonline.gov.uk/SupplierCentre/Metadataguides.htm

Department for Education and Skills (in association with Simulacra and Schemeta) (2003b), Curriculum Online: A Technical Guide, Version 1.07. Retrieved 1 July, 2005, from http://www.curriculumonline.gov.uk/SupplierCentre/Metadataguides.htm

Dublin Core Application Profile Guidelines (2003), CEN Workshop Agreement CWA 14855. Retrieved July 1, 2005, from ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/MMI-DC/cwa14855-00-2003-Nov.pdf

Duval, E., Hodgins, W., Sutton, S. & Weibel, S. L. (2002), Metadata Principles and Practicalities, *D-Lib Magazine*, April 2002. Retrieved July 1, 2005, from http://www.dlib.org/dlib/april02/weibel/04weibel.html

Duval, E. & Hodgins, W. (2003), A LOM Research Agenda. *In Proceedings of WWW2003 - Twelfth International World Wide Web Conference*, 20-24 May 2003, Budapest, Hungary. Retrieved July 1, 2005, from http://www2003.org/cdrom/papers/alternate/P659/p659-duval.html

Friesen, N., Mason, J. & Ward, N. (2002), Building Educational Metadata Application Profiles, *Dublin Core - 2002 Proceedings: Metadata for e-Communities*: *Supporting Diversity and Convergence.* Retrieved July 1, 2005, from http://www.bncf.net/dc2002/program/ft/paper7.pdf

Godby, C. J., Smith, D. & Childress, E. (2003), Two Paths to Interoperable Metadata, *Proceedings of DC-2003: Supporting Communities of Discourse and Practice – Metadata Research & Applications,* Seattle, Washington (USA). Retrieved July 1, 2005, from http://www.siderean.com/dc2003/103_paper-22.pdf

Guidelines for machine-processable representation of Dublin Core Application Profiles (2005), CEN Workshop Agreement CWA 15248. Retrieved July 1, 2005, from ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/MMI-DC/cwa15248-00-2005-Apr.pdf

Hayes, P. RDF Semantics (2004), *W3C Recommendation 10 February 2004*. Retrieved July 1, 2005, from http://www.w3.org/TR/rdf-mt/

Hazaël-Massieux, D. & Connolly, D., (2005) Gleaning Resource Descriptions from Dialects of Languages, *W3C Team Submission 16 May 2005*. Retrieved July 1, 2005, from http://www.w3.org/TeamSubmission/grddl/

Heery, R. & Patel, M. (2000), Application Profiles: mixing and matching metadata schemas, *Ariadne Issue 25*, September 2000. Retrieved July 1, 2005, from http://www.ariadne.ac.uk/issue25/app-profiles/

Heflin, J. (2004), OWL Web Ontology Language – Use Cases and Requirements, *W3C Recommendation 10 February 2004*. Retrieved July 1, 2005, from

http://www.w3.org/TR/webont-req/

IMS Global Learning Consortium (2004), IMS Meta-data Best Practice Guide for IEEE 1484.12.1-2002 Standard for Learning Object Metadata. Retrieved July 1, 2005, from http://www.imsglobal.org/metadata/mdv1p3pd/imsmd_bestv1p3pd.html

Johnston, P., (2005a), XML, RDF, and DCAPs. Retrieved July 1, 2005, from http://www.ukoln.ac.uk/metadata/dcmi/dc-elem-prop/

Johnston, P., (2005b), Element Refinement in Dublin Core Metadata. Retrieved July 1, 2005, from http://dublincore.org/documents/dc-elem-refine/

Klyne, G. & Carroll, J. J. (2004), Resource Description Framework (RDF): Concepts and Abstract Syntax, *W3C Recommendation 10 February 2004*. Retrieved July 1, 2005, from http://www.w3.org/TR/rdf-concepts/

Lagoze, C,. Van de Sompel, H., Nelson, M., & Warner, S. (2002), The Open Archives Initiative Protocol for Metadata Harvesting, Protocol version 2.0 of 2002-06-14. Retrieved July 1, 2005, from http://www.openarchives.org/OAI/openarchivesprotocol.html

Manola, F. & Miller, E. (2004), RDF Primer, *W3C Recommendation 10 February 2004*. Retrieved July 1, 2005, from http://www.w3.org/TR/rdf-primer/

Miles, A. J. & Brickley, D. (2005), SKOS Core Guide, *W3C Working Draft 10 May 2005*. Retrieved July 1, 2005, from http://www.w3.org/TR/swbp-skos-core-guide

Memorandum of Understanding between the Dublin Core Metadata Initiative and the IEEE Learning Technology Standards Committee (2000). Retrieved July 1, 2005, from http://dublincore.org/documents/2000/12/06/dcmi-ieee-mou/

Nack, F., van Ossenbruggen, J. & Hardman, L. (2005), That obscure object of desire: multimedia metadata on the Web, part 2, *IEEE Multimedia 12* (1) 54-63. Retrieved July 1, 2005, from http://ieeexplore.ieee.org/iel5/93/30053/01377102.pdf?arnumber=1377102

Nilsson, M. (2001), The Semantic Web: How RDF will change learning technology standards, Feature article, Centre for Educational Technology Interoperability Standards (CETIS). Retrieved July 1, 2005, from http://www.cetis.ac.uk/content/20010927172953/viewArticle

Nilsson, M., Palmér, M. & Naeve, A. (2002), Semantic Web Metadata for e-Learning - Some Architectural Guidelines, *Proceedings of the 11th World Wide Web Conference (WWW2002)*, Hawaii, USA. Retrieved July 1, 2005, from http://kmr.nada.kth.se/papers/SemanticWeb/p744-nilsson.pdf

Nilsson, M., Palmér, M. & Brase, J. (2003), The LOM RDF Binding - Principles and Implementation, *Proceedings of the Third Annual ARIADNE conference*. Retrieved July 1, 2005, from http://kmr.nada.kth.se/papers/SemanticWeb/LOMRDFBinding-ARIADNE.pdf

Powell, A. (2003), RDN OAI rdn_dc XML Schema(s). Retrieved 1 July, 2005, from http://www.rdn.ac.uk/oai/rdn_dc/

Powell, A. (2005), RDN/LTSN LOM application profile (RLLOMAP). Retrieved 1 July, 2005, from http://www.rdn.ac.uk/publications/rdn-ltsn/ap/

Powell, A., Nilsson, M., Naeve, A., Johnston, P. (2005), DCMI Abstract Model, *DCMI Recommendation*. Retrieved July 1, 2005, from http://dublincore.org/documents/abstract-model/

The International Press Telecommunications Council (2005), News Metadata Framework Requirements specification. Retrieved July 1, 2005, from http://www.iptc.org/dev

UK LOM Core. Retrieved July 1, 2005, from http://www.cetis.ac.uk/profiles/uklomcore

Uschold, M. & Gruninger, M. (2002), Creating Semantically Integrated Communities on the World Wide Web, Invited Talk, Semantic Web Workshop, Co-located with WWW 2002, Honolulu, HI, May 7 2002. Retrieved July 1, 2005, from http://semanticweb2002.aifb.uni-karlsruhe.de/USCHOLD-Hawaii-InvitedTalk2002.pdf

van Ossenbruggen, J., Nack, F. & Hardman, L. (2004), That obscure object of desire: multimedia metadata on the Web, part 1, *IEEE Multimedia 11* (4) 38-48. Retrieved July 1, 2005, from http://ieeexplore.ieee.org/iel5/93/29587/01343828.pdf?arnumber=1343828

Biographies

Mikael Nilsson

Mikael Nilsson has a background in Mathematics and Computer Science. Since 2001, he is working towards a PhD degree in a Swedish National Research School in Mathematics Education. Mikael is based at the Knowledge Management Research (KMR) Group at the Royal Institute of Technology, where his research focuses on ICT-enhanced mathematics education based on semantic web technologies.

During his time at the KMR group, he has been a leading architect and developer of Edutella, an RDF-based P2P network for educational metadata, the metadata management system SCAM and the fully flexible RDF metadata editor SHAME, as well as the knowledge management tool Conzilla. He is actively involved in several international metadata standardisation efforts within technology-enhanced learning, including Dublin Core, IEEE LOM and IMS, and he is the technical editor for the development of an RDF binding for the IEEE LOM metadata standard. He was a co-author of the DCMI Recommendation “DCMI Abstract Model”.

Pete Johnston

Pete Johnston worked as an applications programmer and later as a systems programmer supporting transaction-processing software on mainframe computer systems during the 1980s. He subsequently pursued interests outside the IT sector and in 1994 obtained a first class Honours degree in Latin American Studies from the University of Liverpool. He went on to work in the areas of special collections and archives at the University of Liverpool and digital records management at the University of Glasgow. During this period, he was involved in some of the early implementation of the Encoded Archival Description (EAD) standard in the UK, and developed a broader interest in markup languages and metadata.

At the start of 2001, Pete took up the post of Research Officer at UKOLN, at the University of Bath. Pete's work at UKOLN has been divided between the work of the Interoperability Focus on promoting strategies for the effective exchange and reuse of information, and a number of research projects on metadata registries, including a current project to develop a metadata registry for the JISC Information Environment. Pete is an active contributor to the standards development work of the Dublin Core Metadata Initiative: he is currently a member of the DCMI Advisory Board and chair of the DCMI Collection Description Working Group. He is a co-author of two DCMI Recommendations: “Guidelines for implementing DC in XML”, and the “DCMI Abstract Model”. He is also contributing to the work of the NISO Metasearch Initiative on

standards to facilitate cross-searching in digital library services.

Ambjörn Naeve

Ambjörn Naeve has a background in mathematics and computer science and earned his PhD in computer science from KTH in 1993. He is presently the coordinator of research on interactive learning environments and Semantic Web technology at the School of Computer Science and Communication (NADA) at the Royal Institute of Technology (KTH) in Stockholm, where he heads the Knowledge Management Research group. He has been involved with research and development of interactive learning environments since he initiated the "Garden of Knowledge" project at NADA in 1996. He has also taught mathematics at KTH since 1967 and in the last two decades he has headed the development of several tools for ICT-enhanced mathematics education.

Under the lead of Ambjörn, the KMR group has developed the information architecture called the Knowledge Manifold, the SHAME framework and a number of tools (Formulator, Meditor and Conzilla). The group has also been leading in the design and development of the Edutella infrastructure, the SCAM framework and the tools VWE and Confolio. These items should be considered as contributions towards a publicly accessible Knowledge and Learning Management Environment, based on open source and open international ICT standards as well as on Semantic Web technology. Ambjörn is active within several international networks for technology-enhanced learning and Semantic Web research, notably WGLN, Prolearn and SIGSEMIS.

Andy Powell

Andy Powell received a first class honours degree in Software Engineering from the University of Birmingham in 1984. After some time as a Computer Officer in Bath University Computing Services working on a variety of development and support activities, Andy moved to UKOLN in 1996. Andy is now Assistant Director, Distributed Systems and Services. His main focus of work is to coordinate the technical aspects of the development of the Resource Discovery Network – a cooperative network of subject gateways funded by the JISC. He also acts as technical consultant for the JISC Development Team, developing the architecture of the JISC Information Environment and providing technical support and advice for the various JISC development programmes.

His previous research and development work focused on the use of metadata in resource discovery, and he was involved in a number of projects including: ROADS, DESIRE, PRIDE, BIBLINK, TF-CHIC and NewsAgent. Andy has been active in the development of the Dublin Core - he is a member of the Dublin Core Advisory Board, the Dublin Core Usage Board and is chair of the DC Architecture Working Group. He developed the metadata generator DC-dot, which has been widely used internationally, the metadata help utility DC-assist and OpenResolver, a demonstrator OpenURL resolver. He was co-editor of the "DCMI Namespace Policy" and co-author of the "Guidelines for implementing DC in XML", "Expressing Qualified Dublin Core in HTML/XHTML meta and link elements" and the "DCMI Abstract Model" DCMI recommendations. He was also a member of the Open Archives Initiative technical committee and the OAI-rights technical committee.