Extreme Markup Languages 2006®

Montréal, Québec
August 7-11, 2006

Metadata Enrichment for Digital Preservation

David Dubin
University of Illinois

Joe Futrelle
National Center for Supercomputing Applications

Joel Plutchak
National Center for Supercomputing Applications

Abstract

Description of structural and semantic relationships and properties of, within, and between resources is seen as a key issue in digital preservation. But the markup languages used to encode descriptions for migration between and storage within digital repositories are subject to the same interpretive problems that complicate other uses of markup. This paper reports on a project that aims to address these problems by explicating facts that otherwise would not support automated inferencing. These facts are expressed as RDF [Resource Description Framework] triples, stored in and retrieved from a scalable RDF-based repository.[1]

A Conference of IDEAlliance
Extreme Markup
2006 Languages®
Montréal, August 7–11, 2006

Metadata Enrichment for Digital Preservation

Table of Contents

Markup and Digital Preservation....1
Steps toward metadata enrichment....4
Serializing BECHAMEL Knowledge as RDF....5
Footnotes....7
Bibliography....7
The Authors....8

Metadata Enrichment for Digital Preservation

David Dubin, Joe Futrelle, and Joel Plutchak

§ Markup and Digital Preservation

There are many open questions that complicate the problem of preserving digital information resources, including a lack of consensus on precisely what it means to "preserve" a digital resource, and how one would know whether efforts aimed at preservation are succeeding. There seems to be a general understanding, however, that to the extent a resource encodes meaningful content information, that continued access to that information over time is a necessary condition for preservation. This access should be robust against transformations of the expressions that embody that content, and changes to the environment on which that access depends (computing hardware, application software, etc.). An example of an attempt to come to terms with these issues more formally can be found in the model of Cheney, Lagoze, and Botticelli [cheney01:report]. For a proposal on how such principles would be put into practice, see the "Born Again Bits" migration framework by Liu et al [liu05]

The lack of a full theory of information preservation has not stifled proposals for preservation strategies. The focus of these is typically either emulation of earlier hardware and software environments, or the migration of resources to new systems in which it is hoped the content will be equally accessible. *Description* of structural and semantic relationships and properties of, within, and between resources is seen as key to the second strategy. The following quote, for example, is taken from Section 2 of the *Reference Model for an Open Archival Information System (OAIS)*:

> In order for this information object to be successfully preserved, it is critical for an OAIS [Open Archival Information System] to clearly identify and understand the data object and its associated representation information. For digital information, this means the OAIS must clearly identify the bits and the Representation Information that applies to those bits. This required transparency to the bit level is a distinguishing feature of digital information preservation, and it runs counter to object-oriented concepts which try to hide these implementation issues. This presents a significant challenge to the preservation of digital information [OAISRM02].

Not every recommendation insists on description as fine-grained as the OAISRM [Open Archival Information System Reference Model] would seem to demand, but the association of the resource with descriptions at some level is usually called for — whether that's at the bit level, relationships among files, or at the higher level of domain objects, topics, authorship relations, etc. For some preservation efforts, *Digital Object Repository* systems (such as DSpace [dspace] and Fedora [fedora] play a key role in managing both the metadata descriptions and the files encoding the resources that are described. These repositories are a kind of digital asset management architecture—the same class of information systems that includes digital library and content management systems. Such systems include a storage or database layer, a layer dedicated to resource description, record import and export capabilities, often some form of version control or transaction logging, and the ability to create web-based interfaces for access by users. Typically, information is exchanged between repositories in the form of XML [Extensible Markup Language] records.

The listings below show parts of a Dublin Core metadata record exported from an installation of the DSpace repository system. The resource being described is an aerial photograph expressed as a JPEG [Joint Photographic Experts Group] file, and the record reflects several archival transactions that have already taken place.

Figure 1

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<dublin_core>
  <dcvalue element="contributor" qualifier="none">Scanning, indexing, and description
sponsored by the Illinois State Library and the University of Illinois at Urbana-Champaign
Library. Geo-referencing sponsored and performed by the Geographic Modeling Systems
Laboratory, University of Illinois at Urbana-Champaign.</dcvalue>
  <dcvalue element="contributor" qualifier="author">United States. Agricultural Adjustment
```

```
Agency.</dcvalue>
  <dcvalue element="contributor" qualifier="author">Aerial Photographs</dcvalue>
```

DSpace Export file

The first and most noticeable puzzle in this description is that “Aerial Photographs” is listed as an author. An obvious interpretation is that this is simple tag abuse or human error, but the history of this description reveals it to be an example of a more general and complicated problem. This is the latest in a series of descriptions each derived from an earlier version:

1. A paper description accompanied the original photograph, which had been taken in 1938.
2. In 1998 the photograph was scanned for inclusion in an image database made available on the web [grainger99]. A metadata record for the photograph was entered into a relational database. The fields for that database were derived from the FGDC [Federal Geographic Data Committee] Content Standard for Digital Geospatial Metadata [fgdc98].
3. In May of 2005 an OAI 2.0 metadata record was derived from that database entry, via a mapping from the database fields into Dublin Core.
4. Several months later the OAI record was transformed via XSLT [XSL Transformations] into a form suitable for ingestion into a DSpace installation.
5. When the record was exported from DSpace, additonal DC [Dublin Core] metadata statements had been automatically added.

The relational database at the second stage included three fields, each derived from the FGDC “originator” element: “originator-federal,” “originator-flight,” and “originator-flight-subcontractor.” The distinction among these elements is not part of the FGDC standard, but they were used by indexers to record the contracting government agency (e.g., the Agricultural Adjustment Agency) separately from the aerial survey company that conducted the flight and took the picture. The string “Aerial Photographs” may have been used as a placeholder when no information was available about the survey company, although it's possible that there was a company with that name. If it was just a placeholder, then that demonstrates a certain sloppiness, but none of these “originator” fields were designed as an access points, and the indexer certainly did not intend to assert that the values encoded represented “authorship” as such.

The mapping at the third stage from the RIB'S to Dublin Core was based on the semantics of “originator” in FGDC, not on any distinction between Federal agency and contractor, and so all three fields were mapped to **DC:creator**. The change from creator to **contributor.author** was made because authorship is an important access point for browsing and searching in DSpace, and because the library application profile that informed DSpace's default indexing advised against qualifying **DC:creator** with a role like “author” [dclib] .

A spurious authorship attribution will not prevent future software from correctly displaying a JPEG, and would therefore seem to pose little in the way of a preservation risk. But this example is illustrative of how transformations of this kind, executed over time on large groups of metadata descriptions, can introduce and compound errors. The independent decisions that guided the field and element mappings are representative of how records like this one are managed.

A problem more directly relevant to preservation can be seen at another point in the exported Dublin Core record:

Figure 2

```
  <dcvalue element="format" qualifier="none">image/jpeg</dcvalue>
  <dcvalue element="format" qualifier="extent">23179 bytes</dcvalue>
  <dcvalue element="format" qualifier="extent">209151 bytes</dcvalue>
  <dcvalue element="format" qualifier="mimetype">text/xml</dcvalue>
  <dcvalue element="format" qualifier="mimetype">image/jpeg</dcvalue>
```

Orphan file sizes

Both the JPEG expression of the photograph and the original OAI Dublin Core record were ingested into DSpace, and this export record includes facts about both streams. But nothing in the syntax of the record makes it clear that it's the XML text file that is 23179 bytes and the JPEG file that is 209151 bytes. In fact, there's nothing to indicate that the jpeg stream referred to by the unqualified format element is the same stream as the jpeg referred to in the format statement qualified with the mimetype. DSpace has generated a **description** element that makes all these facts clear in natural language, but we imagine trying to make file size inferences over a large, heterogeneous collection of records and natural language explanations are of limited use in such cases.

The authorship and file size examples are among several interpretive problems we can point to:

1. There are a variety of ways that one can express metadata statements such as these using XML syntax. For example, each metadata element can have its own XML element, instead of an attribute value as in the current example. A human interprets the XML syntax without conscious effort, but brings knowledge to that inference that isn't explicit in the XML. Even if names are introduced via an XML namespace, it's still an interpretive problem to recognize that the role of the namespace is to introduce metadata elements (as opposed to some other role).
2. As mentioned earlier, this export record includes information on both an image (e.g., creator) and the file that express it (a JPEG). In general, a record such as this describes entities at any number of levels of abstraction—it can even, as we see in this example, describe an earlier version of itself. The same metadata element qualified in exactly the same way can describe different levels in different contexts. Extent, for example, can be in inches, pages, or bytes.
3. In theory, each metadata element ought to correspond to a property of the resource (at some level of abstraction). But in this example we find some resource properties expressed using DC elements (e.g. language), some expressed via the qualifier mechanism (e.g., extent), and other examples where the resource property is identified only in the textual content of a metadata statement. This record, for example includes several unqualified date elements where the identification of the event (creation, issuance, scanning, etc.) appears next to the date string in the content of the dcvalue element.

Although XML files as problematic as the DSpace export example are common, it's tempting to say they're simply examples of markup used incorrectly, and to insist that the solution is to promote more careful encoding practice, rather than trying to reconstruct the meaning of badly structured descriptions (as we propose in this paper). One reason we resist this view is that even very carefully and deliberately constructed metadata records can have interpretive problems that, although more subtle than those discussed above, pose similar preservation problems. Two examples of markup used in the METS standard for metadata encoding and transmission illustrate the character of these problems [mets]. These examples are drawn from the METS [Metadata Encoding and Transmission Standard] documentation on the Library of Congress web site.

In METS there are several different ways one can infer the existence of a particular file or stream. For example, a stream can be pointed to with a file name or URI. File contents can be base64 encoded and stored as text in the METS record itself. But one of the most interesting (and least documented) cues for the existence of a stream is shown below. In this example, a Dublin Core description is expresed as XML within an **mdWrap** element. Note the **MIMETYPE** attribute on that element—what is it, precisely that has the property of a MIME [Multipurpose Internet Mail Extensions] type? Clearly, the subtree of nodes under the **mdWrap** is supposed to be understood as a stream in its own right. But in what sense does that subtree consist of an XML stream?

Figure 3

```
<METS:dmdSecFedora ID="DC" STATUS="A">
 <METS:descMD ID="DC1.0" CREATED="2002-05-20T06:32:00">
  <METS:mdWrap MIMETYPE="text/xml" MDTYPE="OTHER"
  LABEL="DC Record for Exhibit Intro: Pavillion III Architectural
  image object">
   <METS:xmlData>
     <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
     xmlns:dc="http://purl.org/dc/elements/1.1/">
        <dc:title>Exhibit Intro: Architectural drawings, Pavilion III,
         University of Virginia</dc:title>
```

METS subtree as a stream

A similar problem can be seen in the example below. In this case, the stream's existence is cued by a **file** element, and identified by an XLink [XML Linking Language] attribute. Once again the file is identified as having the MIME type **text/xml**. But the URI points to a text node *within* an XML document. As with the previous example, it would be a mistake to infer on the basis of the MIME type that (for example) the contents of either stream will validate as XML.

Figure 4

```
<mets:fileGrp ID="FILEGROUP_PRELUDE1.2">
   <mets:file MIMETYPE="text/xml" ID="FILE1.2">
      <mets:FLocat LOCTYPE="URL"
            xlink:href="file:doctxt.xml#xpointer(/Section[1]/
            Section[2]/text())"/>
   </mets:file>
 </mets:fileGrp>
```

fragment identified as XML

Ambiguities and difficulties like those we've discussed arise from a variety of causes. One problem is that typical uses of XML semantically overload a small number of generic syntactic relationships (such as parent/child or element/attribute) [renear02:doceng]. Other problems may occur at an intersection of institutional practices and system features that seem perfectly reasonable when considered separately from each other. Either way, the situation is messy enough to rule out the possibility of a simple, reliable solution. What is needed instead are tools that support a multifaceted strategy for improving metadata records such as these.

§ Steps toward metadata enrichment

Our experimental work on these problems is part of ECHO DEPository [Exploring Collaborations to Harness Objects in a Digital Environment for Preservation], a digital preservation research and development project at the University of Illinois in partnership with OCLC [Online Computer Library Center] and funded by the Library of Congress under their National Digital Information Infrastructure Preservation Program. There are two main dimensions to this part of the project:

- Systems for automatically inferring facts and identifying knowledge gaps.
- The encoding and storage of that knowledge in forms that can be used by a range of different tools, thus supporting diverse approaches to ameliorating the problems identified in the previous section.

As with previous work we have reported, our tools for reasoning over XML markup are developed in the context of the BECHAMEL [Bergen, Champaign, Espanola] markup semantics framework, a research environment for proposing and testing theories of the meaning of markup [dubin03:llc]. BECHAMEL is specifically intended to serve as a workbench for building systems that take conventional markup as input and from that infer the objects that readers would interpret the markup to represent, as well as those objects' properties and the relationships they stand in with respect to each other. Descriptions of our success in disambiguating overloaded markup have been reported elsewhere [dubin03:extreme].

Two important drawbacks of the BECHAMEL system have limited its applicability to digital preservation problems. First, BECHAMEL uses its own native, Prolog [programmation en logique]-based representation of objects, classes, properties, syntactic and relations, and rules of inference. Therefore the only interface between BECHAMEL and other software has been through BECHAMEL's capabilities to process and interpret markup that those systems produce. The discoveries that a BECHAMEL application would make have not been easily ported to other more conventional XML processing tools.

The second limitation concerns scalability: we have developed applications in which inferences are drawn over two or more documents together, and BECHAMEL's Prolog-based architecture can efficiently handle thousands of facts representing, parse tree nodes, domain objects, properties, relations, and so on. But prior to the developments reported in this paper, all such facts were assumed to be resident in memory, ruling out inferences over a large corpus of documents or metadata records.

Our current work on overcoming these limitations is focused on two developments: mapping BECHAMEL's native knowledge representation framework into RDF triples, and the construction of an interface between BECHAMEL and the Kowari metastore.

§ Serializing BECHAMEL Knowledge as RDF

Our earlier papers describe the architecture of BECHAMEL applications, and the methods such systems employ for mapping markup constructs to object instances and properties. Rather than review those details in yet another application context, in this paper we will describe how even the simplest inferences result in explicating facts that are hardly ever recorded as part of a metadata expression, but rather are left as an exercise for human brains. The task in this example is to discover that the **dcvalue** elements in the Dublin Core export record above are metadata statements. In BECHAMEL, this process proceeds as follows:

1. An ontology is loaded into memory, consisting of class declarations for metadata elements, element sets, and the statements in which those element references occur.
2. A metadata record (such as the DSpace export example) is parsed, and a tree of nodes similar to the XML DOM [Document Object Model] is asserted into the native Prolog database.
3. A series of blackboard agents, responding to evidence in the parse tree, begin constructing a network of object, property, and relationship assertions. Each such change to the blackboard may trigger the action of one or more other agents.
4. One such agent discovers that an **element** attribute on a certain **dcvalue** XML element encodes a name known to belong to a metadata element in the Dublin Core element set. This triggers a proposal that this attribute may encode DC element names in every case.
5. A second agent confirms the first agent's hypothesis by retrieving every instance of that attribute on a **dcvalue** element, and checking them against the names of all known DC elements.
6. A third agent then begins constructing *metadata statement* instances for each such **dcvalue** element, and asserting that they are *occurrences* of the metadata elements named in the attribute.
7. A fourth agent recognizes that these statement instances can take a property *qualifier* that corresponds to the name of an attribute on the **dcvalue** elements. It begins assigning the values on that attribute to the values of the corresponding statement instance property.

When the process is complete, the resulting graph of objects, properties, and relations encodes the kind of knowledge that the original record lacks, and which would be too tedious for an indexer to tag by hand. Specifically, it encodes a declarative account of the relationships between the XML syntactic structures and the metadata statements and element references which those XML structures express. These are much more direct announcements of what the XML elements and attributes in the record are doing. The fact that (for example) “contributor” is the name of a Dublin Core element is known ahead of time, but the relationship between the XML syntax and the element names is discovered through the execution of general rules.

To be sure, these discoveries are one or more levels of abstraction removed from those that first motivated our interest, namely ambiguities with respect to resource authorship and file size. But as we saw, those ambiguities sometimes arose from structural problems with the metadata record. It's our view that highlighting those for human intervention and correction is a more promising avenue to pursue than pinning our hopes on fully-automatic disambiguation driven by domain knowledge. Our system does reason about resource properties like mimetype, file size and authorship. But it seems doubtful that we would have a sufficiently deep and robust base of domain knowledge to automatically draw the inference that “Aerial Photographs” isn't an organization's name (since, after all, it could be). In contrast, we are confident that our system could call analysts' attention to a pair of file sizes and a pair of mimetypes that cannot be matched on the available structural evidence. Thus we begin our analysis at the general level of metadata elements and descriptions.

The next issue to be considered is how knowledge of this kind can be expressed in a form that can be used by tools other than those constructed in the BECHAMEL framework. Each of the knowledge structures to which we've alluded (classes, property/value pairs, etc.) has a counterpart in the RDF, RDF Schema or OWL [Web Ontology Language] models, and so exporting our knowledge base in the form of RDF triples would seem a worthwhile exercise.

In translating to RDF, one of the first questions to be addressed is how to assign URIs [Uniform Resource Identifier] to objects and properties that may not suggest obvious candidates. In our running example, some of the objects (such as DC metadata elements) have been assigned URIs by an authority, and if an object discovers itself to have such an “official” URI, then it ought to be possible for that URI to be reported (or constructed if necessary). BECHAMEL's property/value predicates allow the value of a property to be dynamically generated on an as-needed basis, and in that way a metadata element can construct its URI based on the URI of a set in which it stands in a membership relationship.

Figure 5

```
?- exists(E, metadata_element,_), property_applies(E,uri,U), describe(U).
o50
Class= uri
Models: []
  value = http://purl.org/dc/elements/1.1/title

E = o1
U = o50

Yes
```

BECHAMEL transcript

On the other hand, the inference that some object or property exists may come from the distributed action of a large number of blackboard agents, and it can be difficult for BECHAMEL to report from whence it came. In those situations we may resort to unique TAG URIs generated on the fly and impossible to reconstruct if the object or property is ever rediscovered on another run of the system. More often, though, BECHAMEL has maintained a link from a knowledge structure back to the evidence that licensed its assertion, and this trace can help produce a serviceable (if not very human-readable) URI:

Figure 6

```
?- exists(S,metadata_statement,_),property_applies(S,uri,U), describe(U).
o65
Class= uri
Models: []
  value = tag:eprg@isrl.uiuc.edu,2006-03-16:/bechamel/markup/
dublin_coreExport.xml#+object+ECHODEPRMO+0.1+
metadata_statement+xpointer(/dublin_core[1]/dcvalue[1])

S = o18
U = o65

Yes
```

Generated URI

That URI is read as the instance of a particular class belonging to a particular version of a particular ontology, said instance having been inferred from a particular node in the tree produced by parsing the file “dublincoreExport.xml.” The aim is to be able to distinguish separate URIs for an element node, the metadata statement encoded by that node, the resource property expressed by that statement, the value assigned to that property, the resource exhibiting the property, and the metadata element naming the property, each of which have a separate identity.

The goal of our project is to store and retrieve RDF triples to a remote installation of the Kowari metastore. It is possible to use SWI [Sociaal-Wetenschappelijke Informatica] Prolog's SemWeb [Semantic Web] module to serialize a set of RDF statements in RDF/XML. But such automatically-generated serializations are much more awkward to read than expressions crafted by hand. It's much easier to interpret the statements expressed as triples of URIs, as shown in the transcript below:

Figure 7

```
?- rdf(A,B,C), print(A), print('\n'), print(B), print('\n'), print(C),
print('\n\n'), fail.

http://purl.org/dc/elements/1.1/title

http://www.w3.org/1999/02/22-rdf-syntax-ns#type

tag:eprg@isrl.uiuc.edu,2006-03-16:/bechamel/ontology/EDRDV/0.1#+
class+metadata_element
[...]
tag:eprg@isrl.uiuc.edu,2006-03-16:/bechamel/markup/dublin_coreExport.xml#+
object+EDRDV+0.1+metadata_statement+xpointer(/dublin_core[1]/dcvalue[1])
```

```
http://www.w3.org/1999/02/22-rdf-syntax-ns#type

tag:eprg@isrl.uiuc.edu,2006-03-16:/bechamel/ontology/EDRDV/0.1#+
class+metadata_statement
[...]
```

Inferred RDF triples

Once RDF statements are produced by BECHAMEL, they can be made persistent using any of a number of RDF databases. Managing RDF statements in a database enables them to be used in ways that significantly augment BECHAMEL's capabilities. Firstly, an RDF database provides a point of interoperability between statements generated by BECHAMEL and statements produced by other applications. For instance, if BECHAMEL infers that a domain object was created in 1993 and records a statement to that effect in the database, the domain object can be associated with other statements relevant to that year using operations on the RDF database, regardless of which application produced the statements. Secondly, most RDF database technologies scale well beyond what can fit in physical memory on a typical computer, enabling very large amounts of descriptive information to be aggregated and analyzed, including statements generated by BECHAMEL.

It should be noted that we don't claim our approach to be an answer to the very general scalability problems associated with large graphs. We simply aim at any time to be working with manageable pieces of large graphs, storing and retrieving those pieces as needed.

SWI Prolog provides some support for RDF databases in its Semweb package, but does not yet support some of the most recent innovations in RDF storage. For instance the Kowari store uses AVL [Adelson-Velsky and Landis] trees and other unusual indexing strategies to enable the efficient traversal of RDF graphs, exceeding the performance of virtually all other RDF database implementations for large numbers of RDF statements. Kowari makes it feasible to decompose collections of complex objects into many statements, store all the statements, and retrieve relevant subsets for typical use cases in a reasonable amount of time. Emerging standards such as SPARQL, [SPARQL Protocol and RDF Query Language] once married with scalable implementations, will create a new class of standards-based, high performance RDF databases that can receive and manage metadata generated by applications such as BECHAMEL. To this end, we have developed an SWI Prolog interface to Kowari enabling Prolog applications to issue queries and updates against a Kowari instance and receive the results as simple Prolog data structures.

Notes

1. This material is based upon work supported by the Library of Congress under the grant “Exploring Collaborations to Harness Objects in a Digital Environment for Preservation”, award number A6075. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the awarding agency, or the University of Illinois.

   The authors are grateful to the participants in the University of Illinois GSLIS [Graduate School of Library and Information Science] Research Writing Group and to the anonymous reviewers of this paper for their helpful comments and suggestions.

Bibliography

[cheney01:report] Cheney, J., Lagoze, C., and Botticelli, P. Towards a theory of information preservation. Tech. Rep. TR2001-1841, Cornell University, Ithaca, NY, 2001.

[dclib] Clayphan, R., and Guenther, R. DC-Library application profile (DC-Lib). Published on the World Wide Web at http://dublincore.org/documents/2004/09/10/library-application-profile/, September 2004. DCMI Working Draft.

[dspace] Smith, M., Barton, M., Bass, M., Branschofsky, M., McClellan, G., Stuve, D., Tansley, R., and Walker, J. H. DSpace: an open source dynamic digital repository. *D-Lib Magazine 9*, 1 (2003).

[dubin03:extreme] Dubin, D. Object mapping for markup semantics. In *Proceedings of Extreme Markup Languages 2003* (Montreal, Quebec, August 2003), B. T. Usdin, Ed.

[dubin03:llc] Dubin, D., Sperberg-McQueen, C. M., Renear, A., and Huitfeldt, C. A logic programming environment for document semantics and inference. *Literary and Linguistic Computing 18*, 2 (2003), 225-233. (This is a corrected version of an article that appeared in 18:1 pp. 39-47).

[fedora] Fedora Development Team. Fedora open source repository software: White paper. Published on the World Wide Web at http://www.fedora.info/documents/WhitePaper/FedoraWhitePaper.pdf

[fgdc98] Federal Geographic Data Committee Metadata Ad Hoc Working Group. *Content Standard for Digital Geospatial Metadata*. US Geological Survey, Reston, VA, 1998. FGDC-STD-001-1998.

[grainger99] University of Illinois Grainger Engineering Library. Historic aerial photo imagebase - an Illinois aerial photograph search and retrieval system. Published on the World Wide Web at http://images.library.uiuc.edu/projects/aerial_photos/, 1999.

[liu05] Liu, A., Durand, D., Montfort, N., Proffitt, M., Quin, L. R. E., Réty, J. H., and Wardrip-Fruin, N. Born-again bits: A framework for migrating electronic literature. Published on the World Wide Web at http://www.eliterature.org/pad/bab.html, August 2005. version 1.1.

[mets] United States Library of Congress. METS: An overview and tutorial. Published on the World Wide Web at http://www.loc.gov/standards/mets/METSOverview.v2.html, May 2005.

[OAISRM02] Consultative Committee for Space Data Systems. *Reference Model for an Open Archival Information System (OAIS)*. Washington, DC, January 2002.

[renear02:doceng] Renear, A., Dubin, D., Sperberg-McQueen, C. M., and Huitfeldt, C. Towards a semantics for XML markup. In *Proceedings of the 2002 ACM Symposium on Document Engineering* (McLean, VA, November 2002), R. Furuta, J. I. Maletic, and E. Munson, Eds., Association for Computing Machinery, pp. 119-126.

The Authors

David Dubin
University of Illinois, Graduate School of Library and Information Science
501 E. Daniel Street
Champaign
IL
61820
USA
ddubin@uiuc.edu
tel: 217-244-3275
fax: 217-244-3302

David Dubin is a senior research scientist on the staff of the Information Systems Research Lab at the University of Illinois Graduate School of Library and Information Science. He is a member of the Electronic Publishing Research Group.

Joe Futrelle
National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign
1205 W. Clark Street
Champaign
IL
61820
USA
futrelle@ncsa.uiuc.edu
tel: 217-265-0296
fax: 217-244-1987

Joe Futrelle is a senior research coordinator at the National Center for Supercomputing Applications.

Joel Plutchak
National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign
1205 W. Clark Street
Champaign
IL
61820
USA
plutchak@ncsa.uiuc.edu
tel: 217-244-5355
fax: 217-244-4393

Joel Plutchak is a software developer at the National Center for Supercomputing Applications.

Extreme Markup Languages 2006®
Montréal, Québec, August 7-11, 2006
This paper was formatted from XML source via XSL
by Mulberry Technologies, Inc.