# Two Paths to Interoperable Metadata

Carol Jean Godby, Devon Smith, and Eric Childress
{godby, smithde, childress}@oclc.org
OCLC
Online Computer Library Center, Inc.
6600 Frantz Road
Dublin Ohio, USA
43017

## Abstract

*This paper describes a prototype for a Web service that translates between pairs of metadata schemas. Despite a current trend toward encoding in XML and XSLT, we present arguments for a design that features a more distinct separation of syntax from semantics. The result is a system that auomates routine processes, has a well-defined place for human input, and achieves a clean separation of the document data model, the document translations, and the machinery of the application.*

**Keywords:** *metadata schema translation, interoperability, Web services, communities of practice, XML, XSLT*

## 1. Introduction

Our goal is to develop software that performs metadata schema transformations. By *metadata schema*, we mean a formal specification for encoding knowledge about an object—such as a bibliographic record, a description of a learning object, or perhaps the descriptive data enclosed in an HTML <meta> tag. Specifications that are familiar to managers of digital libraries include Dublin Core[1], EAD [2], LOM [3], MARC [4], and ONIX [5]. We omit from the scope of our study the semi-structured data formats that represent proprietary conventions for encoding descriptions, such as the familiar *Author:* and *Title:* fields in Dialog records. Also excluded are purely structural transforms of digital objects that usually do not contain metadata, such as conversions from one word processor format to another, or from HTML to plain text. As members of the library community, we are interested in improving methods for managing sets of resource descriptions that conform to public standards but originate from a variety of sources—libraries, vendors, museums, educators, and government agencies. Software that effectively translates among the common metadata schemas promises to neutralize the logistical problem of creating or searching heterogeneous databases, an issue facing all digital library projects of at least moderate size.

Given that metadata translation is a common and even critical problem in the library environment, it is reasonable to wonder why it continues to be an object of study. Unfortunately, the problem is far from solved. The evidence is easy to find, both inside OCLC and in the digital library projects that have reported the details of their systems. The functionality of metadata translation may be embedded in large, complex applications that may be inaccessible to small libraries or libraries with specialized needs. The source code in these applications often contains long sections of *if-then* statements that are difficult to read and are rendered obsolete whenever the input data or a standard changes. A program may be unfinished because it was written for a standard that is no longer needed. Nearly identical programs may be duplicated because different project groups have slightly different needs. Metadata schema transformations are more complex than purely structural transforms because they require a set of equivalences identified by human experts—Dublin Core *title* can be mapped to MARC *245*, Dublin Core *author* can be mapped to MARC *100* and so on—but this important knowledge is recorded in a multitude of ways that are not standardized and not always machine-processable, including Web pages, databases, spreadsheets, PDF documents, and the source code of many computer languages.

With these problems in mind, we have defined the following design goals:

- The development of a self-contained metadata translation service,
- The clean separation of the document data model, the schema translations, and the machinery of the application,
- The automation of routine processes and a well-defined place for human input,

– Support for current metadata creation practice and for foreseeable innovation.

We are revisiting the problem of metadata schema translation as part of a larger research project currently in progress at OCLC. The goal of the Metadata Switch project [6] is to create a collection of Web services that perform key tasks required for the management of the digital library—such as Web page harvesting, automatic subject assignment, and the standardization of terminology and proper names—and share many of the technical problems we have identified. The Web services protocol [7], defined by the World Wide Web Consortium, promises to impose a common standard on a set of modular functions and increase their accessibility to clients whose data management problems cannot be solved by expensive turnkey systems.

## 2. A scenario

In one potential application of the services in the Metadata Switch project, a digital library project collects metadata records from dozens of sources to present a coherent Web-accessible picture of a given state's or country's cultural heritage to the rest of the world. Volunteers from historical societies contribute Dublin Core records that describe photographs of now-demolished buildings. University and public libraries contribute MARC records describing related books, journals, unpublished notes, and sound recordings. But the records are inconsistent because the public libraries classify their materials with the Dewey Decimal Classification, while the university libraries use the Library of Congress Classification, and the historical societies use homegrown subject schemes. This scenario quickly grows complex, but the root problems are already apparent. The digital library must handle multiple data streams, some of which may need to be harvested from local sources. Records must be enhanced or corrected because they were created for different audiences by staff with different professional standards.

A search interface that effectively locates the records and presents them to users in a meaningful context requires some minimal level of interoperability, which is accomplished by a metadata schema translation service like the one we have created. If properly designed, it may be invoked at more than one level in the process flow. For example, a service that links roughly equivalent title and author fields in the most common metadata schemas could be accessed as databases of different materials are searched. More extensive translations are possible at an earlier stage, when databases are created. A translation service can standardize records to a common format, giving database designers the flexibility to base their decisions about which metadata standards to adopt on non-technical criteria, such as the intended audience for the resource collection, or the availability of trained cataloging staff.

## 3. The design of a metadata schema translation service

Figure 1 shows the high-level design of our translation service. It has three parts: a translator enclosed in a Web service wrapper, a record submission client, and a crosswalk registry client. A record to be translated is submitted from the record submission client, which may be a software program or an interface that accepts human input. It is translated according to lists of equivalences, or crosswalks, created by human experts. A specialized client to the Web service, the crosswalk registry client, allows users to submit and test translations.

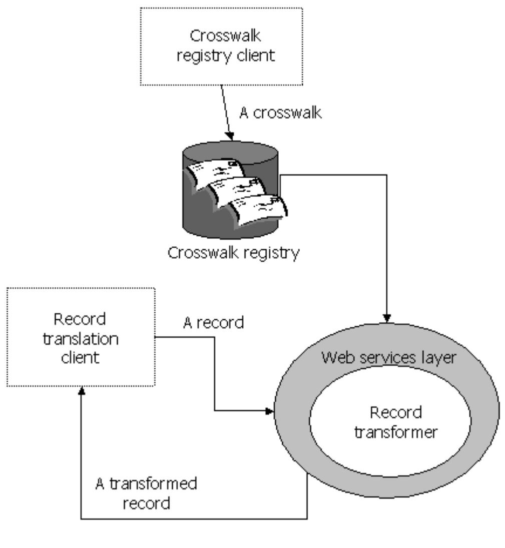

**Figure 1**. High-level design of the metadata schema translation service

The prototype crosswalk registry client maintains a list of translations that we have implemented and tested. It has also been seeded with crosswalks for important metadata standards that have been posted on the Web in such frequently visited locations as the Library of Congress [8], the Getty Museum [9], and the MIT DSpace project [10]. An accompanying Web interface [11] to our service fills in what these sites omit, providing links to the complete versions of the involved standards, to the files required for machine processing, and to the organizations that sponsor the development of the standards. Once the service achieves critical mass, the crosswalk registry will be made available to members of the metadata standards community.

Figure 2 shows our design for the translator. It accepts diverse input formats and has separate modules for syntactic transformation and semantic mapping. This design allows a minimal passthrough for the easy cases and applies more extensive processing where necessary.

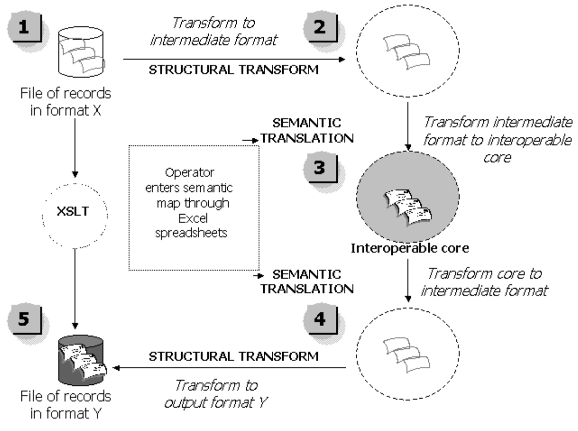
1
Transform to intermediate format
2
STRUCTURAL TRANSFORM
File of records in format X
SEMANTIC TRANSLATION
Transform intermediate format to interoperable core
3
XSLT
Operator enters semantic map through Excel spreadsheets
Interoperable core
SEMANTIC TRANSLATION
Transform core to intermediate format
5
4
STRUCTURAL TRANSFORM
File of records in format Y
Transform to output format Y

**Figure 2.** Transforms in the interoperable core

## 3.1. The short translation path

In the most straightforward case, shown in the left-hand edge of Figure 2, documents are represented in XML and transformed using XSLT in a process that starts at Step 1 and goes directly to Step 5. The correct application of XML requires the creation of *XML schemas*—which are, essentially, grammars that define well-formed documents. XSLT (Extensible Stylesheet Language Transformations) [12], a World Wide Web Consortium recommendation, operates on documents that conform to the syntax specified by XML schemas.

By prominently featuring XML, our design achieves a clean separation of the document representation and translation functions from the machinery of the application, offering a solution that makes state-of-the-art technology accessible to clients with limited technical expertise. This benefit is significant, given that XML is gaining momentum as the syntax of choice for representing Web-accessible metadata. When the complex conditions for the use of XSLT are met, application developers need only issue simple commands to identify and isolate subtrees that would have the effect of transforming statements commonly found in metadata crosswalks, such *DC:Title* to *LOM <general.title>* [13], or *DC:Description* to *EAD <archdesc> <abstract>* [14].

Unfortunately, the direct translation path from Step 1 to Step 5 has several problems that result from using XSLT to do a job it was not designed for. XSLT operates on tree-structured documents that have been successfully parsed by XML validators. For example, many crosswalks involving complex standards such as MARC or ONIX require the manipulation of a record's unique data, not just XML tags, as in the statement *ONIX <IllustrationsNote> is mapped to MARC 300$b (concatenate to existing $b)* [15] , or the statement *DC:Source is mapped to LOM <relation.resource> when the value of Relation.kind is IsBasedOn* [13]. In these cases, the translation code is not simple because XSLT has only a limited ability to extract, combine, and otherwise refer to the parts of element values.

Moreover, an XSLT statement is an instruction for a structural transform, but a mapping between two fields in a metadata crosswalk is also a statement about semantics. The above examples encode assertions that the concepts of *title*, *description,* or *illustration note* are preserved when the specified substitutions are made, but this assertion about meaning is inextricably bound to the statements about syntax. The statements do not spell out what, exactly, the meaning is, whether the equivalence is exact or approximate, or whether the equivalence is retained if the structural transform changes slightly in a new edition of the metadata standard. Is a *description* the same as an *abstract*? Is an *illustration note* a paragraph-like description, or a title-like caption?

A further problem is that the mappings are defined in pairs. In a repository of crosswalks like ours, the number of mapping pairs grows rapidly, which adds complexity but no useful knowledge. Because of the implied semantic assertions, we cannot use the mappings to create long chains that would link equivalent fields in, say, Dublin Core, MARC, GEM, and LOM because an unspecified degree of meaning would be lost with each transform. According to the W3C documentation [12], XSLT is "a language for transforming XML documents into other XML documents." XSLT statements that manipulate data below the level of XML tags, or bind semantics with syntax, are limited in their capabilities because XSLT was designed for structural manipulation.

Aside from these technical problems, a solution to the metadata schema translation problem that relies solely on XML and XSLT has a more tangible roadblock. Despite a growing number of XML metadata schemas, and crosswalks that can theoretically be represented in XSLT, XML-encoded metadata records are still scarce. Our investigation has revealed that only large research libraries currently have the resources to create and process XML-encoded metadata. The solution sketched on the left-hand side of Figure 2 might work for some of the easiest transforms, but it is too early for all clients except the most technically sophisticated.

## 3.2. The long translation path

The second path illustrated by Figure 2 expands the usability of the system and addresses the need to decouple syntactic transforms from semantic mappings.

To process documents, the long path to Step 5 shown on the right side of Figure 2 accepts a wide range of structured inputs that can be converted to XML. It incorporates some of the Open Source GRUNK package developed by the Digital Library Technologies group at the National Center for Supercomputing Applications [16]. The Grammar Understanding Kernel is a Java library that parses semi-structured text formats and converts them to a standard syntax specified by the user, which includes XML.

To process mappings, the long path capitalizes on the informal practice reported in many digital library projects and accepts spreadsheets as input. In a future version of the metadata translation service, we would like to extend the functionality of the mapping intake function by automatically interpreting the tables of correspondences that usually make up the metadata crosswalks found on the Web. Perhaps the GRUNK package can be useful here, too, since tables represent the kind of semi-structured text that it should be able to parse.

Once the input is ingested and converted to standard forms, documents are translated through the interoperable core, using the semantic maps. Once this step is complete, they are formatted in the desired syntactic format—say, XML, or a displayable version of MARC—and returned to the client.

We continue to experiment with the content of the interoperable core, but it now resembles a MARC-like ontology with additional fields for encoding information that is beyond the scope of a bibliographic record, such as those required for describing the software environments of learning objects. In later revisions, we will consider using the controlled vocabulary being developed by the INDECS [17] project, which was designed to promote interoperability among metadata standards.

Figure 3 shows a schematic representation of the relationship between the metadata standards and the interoperable core. Translations between metadata standards are achieved via mappings to and from the core. These mappings may be more-or-less complete, as when translations for closely related standards, such as the various dialects of MARC, are submitted through the registry process depicted in Figure 1 for the purpose of record conversion in an offline database creation process. Or the mappings may be partial, as might be required in a real-time search process that refers only to *author*, *title*, *date*, and *subject* elements.

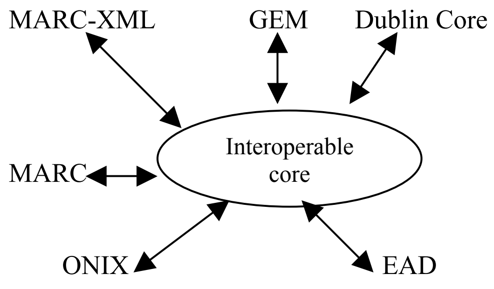

**Figure 3.** Mappings to the interoperable core

In the first stage of the project, the collection of translations is seeded by creating semantic mappings between metadata standards and the interoperable core, which are submitted by development groups at OCLC and our research partners in the digital library community. As the collection grows, the number of explicit mappings to the core can be expected to diminish because the previously submitted mappings can be leveraged to support new translations. For example, if mappings between MARC and GEM and MARC and Dublin Core have been registered, there is no need for an explicit mapping between GEM and Dublin Core because all three standards are already represented in terms of the interoperable core.

## 3.3. The long and short of metadata translation

The design of the long translation path has two desirable consequences. First, only two translation steps are ever required: from the source standard to the interoperable core, and from the core to the target standard. In the model represented by XML and XSLT transforms, inferences from GEM to Dublin Core to MARC are possible, but would require a long chain of mappings, which risks loss of information at each step. Second, the collection grows in power as more translations are submitted. In the previous model, the proliferation of mappings is a problem that leads to confusion and reduces the usability of a metadata translation service.

This problem is partly a consequence of the fact that, in the short translation path, the syntax and semantics of the mappings are inseparable. As a result, when multiple XML schemas or XSLT transforms representing the same metadata standard have been registered, it is difficult to determine whether the variants are semantically equivalent. Even in our prototype service, the user must select among Dublin Core, Dublin Core Simple, or Dublin Core XML; or between MARC, MARCSlim, and UniMARC. The differences between these variants are not obvious from visual inspection of the XML or XSLT code, and because a given document to be translated must conform to a single XML schema, it can be processed only by a single path through the system. But in the expanded translation process that separates syntactic transformation from semantic mapping, a single translation emerges for each standard. When variants are registered—for example, full and truncated versions of MARC appropriate for database creation and searching, respectively—they must be identified. The differences in outputs can be more easily verified because the same document can be translated with any of the variant mappings.

Our design for a metadata schema translator that separates syntax from semantics is consistent with the arguments made in Hunter and Lagoze [18], who proposed using XML for representing documents, augmented with Resource Description Framework (RDF) statements [19] for expressing semantic equivalences. Our work differs from theirs in technical details, such as a cleaner separation of function and a set of modules that allow users to register and test mappings. We believe that the interoperable core could be implemented in RDF, but our experience has led us to conclude that RDF representations introduce unnecessary processing bottlenecks, a result also reported by Heery and Wagner [20].

Nevertheless, we remain convinced of the need to retain two paths for translation. The XML/XSLT path may conflate syntax with semantics, with undesirable consequences, but it has some compelling advantages: it leverages current trends; it can be executed with relatively little processing overhead; and, once it accepts some circumscribed human input, the remaining processes are automated. This solution is appropriate when the transforms fall within the natural scope of XSLT statements and the transforms are essentially syntactic, as when variants of MARC are translated. The system design shown in Figure 2 also allows for a slight relaxation of the strict syntactic requirements of this path. A sequence from Steps 1 to 2 to 5 would permit users to submit structured non-XML records, with further processing in XML and XSLT.

Translation through the interoperable core is more abstract and eliminates hard-coded dependencies between documents and syntactic specifications. But this formal elegance comes at a price. As our comments suggest, the process is not fully automatic because the interoperable core must be hand-crafted. And, as we argue in Section 5, this is an ongoing process that must be constrained by the goals of the project and current best practice.

### 3.4. An example

To illustrate the steps required by the long translation path, consider the real-world task of converting a set of GEM records to MARC. Our collaborators at Syracuse University have submitted a set of test records, as well as a draft of a crosswalk from GEM to MARC [21]. Since the GEM-MARC crosswalk is represented as a table in a Web page, we converted it to a spreadsheet to automate the translation process. Our software reads the spreadsheet entries and translates them into a set of LISP-like commands that perform the semantic mapping. For example, the crosswalk entry that establishes the mapping for the *title* element—**GEM element** *title* **MARC tag** *245* **ind1** *0/1* **ind2** *0-9* **subfields** *$a $h[electronic resource]*—is converted to the following code:

```
(map  (source-element "title") (core-element "245" "a")
        (addField parent ("i1" "0"))
        (addField parent ("i2" "0"))
        (addField parent ("h" "[electronic resource]")))
```

When this code is executed, one output [22] is a conversion from an XML-encoded GEM element such as *<title>English Grammar</title>* to the corresponding XML-encoded MARC element :

```
<245><i1>0</I><i2>0</i2><a>English Grammar </a><h>
[electronic resource] </h></245>.
```

However, this short account glosses over two technical details. First, as the procedure depicted in Figure 2 suggests, the translation is not literally from GEM to MARC, but is a two-stage process that converts GEM to the interoperable core and then converts the core to MARC. Since the current implementation of the interoperable core is a working draft that closely resembles MARC, the translation of the *title* element falls within the scope of crosswalk statement and differs only in slight syntactic detail from the MARC XML output:

```
<fld tag="245" ind1=”0” ind2=”0“>
  <sfld code=”a”> English Grammar 101</sfld>
  <sfld code="h">[electronic resource]</sfld></fld>
```

The interoperable core is necessary for handling elements that are not directly translatable to MARC, such as GEM *audience*. To capture the meaning of this element, we are experimenting with a set of options that include creating a MARC-like pseudo-element or representing the entire interoperable core in an abstract data modeling language, such as RDF.

The second detail omitted from our initial description of the translation process is a syntax normalization step performed on incoming and outgoing records. For example, the *title* element in the incoming GEM record element is converted to an isomorphic form that preserves references to GEM elements, attributes, and XML namespaces:

```
<field name=”title”
namespace=http://purl.org/dc/elements/1.1>
      <value>English Grammar 101</value></field>
```

After this field is mapped to the interoperable core, it is converted to a MARC record expressed in the same syntax:

```
 <field name="245">
  <field name="a"><value>English Grammar
101</value></field>
  <field name="h"><value>[electronic
resource]</value></field>
  <field name="i1"><value>0</value></field>
  <field name="i2"><value>0</value></field></field>
```

The normalization step preserves the logic of the metadata records in a format that can be processed with a single parser and rendered in a variety of desired outputs with a few simple commands.

Thus, in the long translation path, records are normalized, translated, and normalized again before they are returned in the user's requested output. This sequence of steps gives our model the flexibility required to separate syntax from semantics and to handle records in multiple formats. But the apparent complexity is hidden from the user, who interacts with the system through familiar objects: a set of input records requiring translation, a crosswalk encoded in a table, and an output format corresponding to a published metadata standard.

## 4. The process of evaluation

Our effort is currently focused on the development of a usable prototype for translating among metadata schemas, which leverages some of the expertise now embedded in OCLC's products. The result is packaged and registered as a Web service that can work with the document harvesting, data enhancement, and classification services that comprise the OCLC Metadata Switch research project.

To test and evaluate our translation service, we constructed a ranked list of standards for correctness, which is based partly on the arguments in St. Pierre and LaPlant [23]:

1. A valid translation from the source to itself through the interoperable core—the null transform;
2. A valid translation from a source standard to a target standard, via crosswalks that have been submitted to the service and attached to the interoperable core, as implied by Figure 2;
3. A valid translation from a source to a target and back to the source—the so-called round-trip; and
4. A valid translation from a source to a target through an implied relationship among standards mapped to the interoperable core.

Of these measures, Step 1 is a sanity check and Step 4 is a goal for future work; Steps 2 and 3 constitute the essential evaluation.

The first clients for our service are development groups who convert records from third-party sources to a MARC format required for loading into OCLC's WorldCat® database. This task introduced some biases in the evaluation. Most importantly, the problem of database preparation for a product with high standards for quality control is probably the most difficult test for a metadata schema translation service because the target is a mature and complex standard. The transformation must be complete, and if possible, symmetric, because an operator should be able to submit a record to the database and retrieve it back in the original format. Though the requirements for metadata translation in the database searching problem are probably less stringent, the database preparation problem comes with some valuable resources, including OCLC's extensive library of validation routines, which largely automate Step 2.

The prototype cannot be considered mature until our system can ingest and fully process a reversible, spreadsheet-encoded mapping from MARC to Dublin Core submitted by a cataloging expert. With the achievement of this milestone, we will have succeeded in producing a software environment that allows cataloging and metadata professionals to focus on the intellectual challenges of metadata conversion, while eliminating the distractions of multiple input formats and the need to write source code, even in apparently friendly scripting languages such as XSLT.

## 5. An open issue: the design of the interoperable core

The interoperable core is currently based on MARC, despite the fact that nearly all of the publicly accessible metadata crosswalks refer to Dublin Core as a source or target. Since the Dublin Core standard was motivated in part by the need to increase interoperability among competing metadata standards, isn't it a more natural choice for the interoperable core?

The designers of metadata translation services must decide whether the interoperable core should achieve an approximate union, or intersection, of the elements in the standards that will be mapped to it. Though either option is defensible, our initial problem—data conversion—cannot be solved without a complete translation, which requires a union, if possible.

Perhaps by definition, Dublin Core is suitable for intersection. We believe it would be a reasonable choice for lightweight, real-time applications that translate portions of single records. However, our initial tests have forced us to wonder if Dublin Core would be problematic even in this role. For example, we have examined approximately 400 Dublin Core records from three data streams that were submitted to one of our test clients from a digital library project. Analysis of the records reveals that only seven of the fifteen Dublin Core elements appear in all three data sets: *Identifier*, *Title*, *Creator*, *Subject*, *Date*, *Type*, and *Format*. Of these, *Subject* and *Description* both contain subject headings and free-text descriptions; *Format* and *Type* both contain names of media types such as *photograph*; and the data in the *Language* field is ambiguous between the language of the metadata record and the language of the content. Without extensive human-mediated correction, or training that promotes more consistent application of the Dublin Core element semantics when the records are created, even the goal of limited interoperability is compromised.

### 5.1. Communities of practice

A more sophisticated answer to the question about the content of the interoperable core can be developed with reference to the concept of *communities of practice*. According to Wenger [24], a mature community of practice is a connected network of professionals who work on common problems and speak a common language. Friesen [25] argued that communities of practice are more likely to produce highly interoperable metadata because they have common goals that are formally articulated in professional publications, conferences, discussion lists, and standards committees, which are reflected in a large shared vocabulary.

For example, the learning object community represents an interesting community of practice because it is currently evolving. The goal of this community is to manage the growing number of learning materials that are being developed for dissemination on the Web. The stakeholders have developed an extensive vocabulary for use within their communities, including *Learning Resource Type*, *Typical Learning Time*, and *Intended End User Role.* The metadata for learning objects is designed to be interoperable because the communities work with mutual knowledge of the major standards such as LOM, GEM, and the Dublin Core extensions for education. It would be relatively easy to develop an interoperable core for our system based on learning object metadata because the learning object community has done most of the work.

The bibliographic community, which we represent, is an older community of practice. Thus it shouldn't be surprising that our proposed interoperable core is based on a bibliographic standard because it represents our attempt to grapple with the real but manageable differences in the stewardship of intellectual property as viewed by the library community, the publishing community, and managers of cultural heritage institutions. To put reasonable bounds on our effort, we should limit our attention to the problem of creating interoperable metadata from these related communities.

## 5.3. The limits of interoperability

Perhaps more than one interoperable core may be necessary: one for the bibliographic community, one for the learning object community, and so on. Yet if the eventual goal in a digital library project is to organize a wide range of intellectual resources, regardless of their origins, we would need to achieve interoperability across communities of practice as well as within them.

Friesen [25] argued that interoperability across communities of practice is likely to be far less comprehensive than interoperability within a single community. Following an observation by Wenger [24], we can infer that this result is a natural consequence of the fact that a day-to-day immersion in a set of problems produces a detailed vocabulary, only some of which needs to be shared with the rest of the world. Each community of practice chooses the vocabulary that is appropriate for export and educates outsiders in its use. For example, as laymen in economics, we know something about *supply and demand* and *cost-push inflation* because of our exposure to these concepts in introductory textbooks, but we can only guess at the meanings of *paradox of thrift* and *expectations-augmented Phillips curve*.

The rich concept of *community of practice* has several implications for the study of metadata and interoperability. First, each community must demonstrate a formal commitment to interoperability by identifying concepts to be exported. To some extent, this work is already being done and is documented in published crosswalks. But we have discovered that many standards are represented by multiple crosswalks with different formal encodings that may or may not produce the same results—predictable problems that arise when syntactic translation is not dissociated from semantic mapping.

This problem suggests that interoperability must be addressed at a higher level of abstraction. Just as writers of introductory economics textbooks must make decisions about how to identify and define key concepts that are precise enough to be useful, but with detail omitted that only experts can appreciate, so must the designers of metadata standards. For example, what does the world really need to know about the concepts of *title* and *learning object*? Do non-experts need to know about *Intended End User Roles* or *Varying Forms of Title*? Once the key concepts have been packaged for export, they can be mapped to Dublin Core. In other words, Dublin Core is suitable for crosswalks between communities of practice, but is less valuable for crosswalks within communities because a Dublin Core encoding risks a loss of precision.

A consequence of this view is that complete translations are possible only within a given community of practice, while only partial translations are possible between them. Thus, libraries should be able to create databases by resolving differences among multiple versions of MARC, or by accomodating related standards such as ONIX, which was developed by the publishing community. But the complete translation of a standard designed for a radically different purpose—such as SCORM, which preserves details required for creating a technical infrastructure for learning objects—is unlikely. Instead, a reasonable goal for a digital library with a database of SCORM records is to retain the native format but increase its accessibility with improvements in searching. Heterogeneous collections of databases will remain, but they can be unified through a common search language derived from a simplified crosswalk that spans communities of practice.

Finally, this discussion has implications for the design of the interoperable core in our metadata translation service. As we have hinted throughout this paper, the interoperable core must be designed to accommodate change because the problem of metadata translation is not simply technical. The most constrained solution is a Web service template with multiple instantiations. Each instance implements an interoperable core that serves a given community of practice and exports a Dublin Core view of its data to promote interoperability with other communities. Our prototype creates a flexible environment for this test of metadata schema translation.

## References

(URLs accessed July 21, 2003)

[1] Dublin Core Medata Initiative. (2003). http://www.dublincore.org

[2] Encoded Archival Description: Official EAD Version 2002 Web Site. (2002). http://www.loc.gov/ead/

[3] Draft Standard for Learning Object Metadata. (2002). http://ltsc.ieee.org/doc/wg12/LOM_1484_12_1_v1_Final_Draft.pdf

[4] MARC Standards. (n.d.). Library of Congress Network Development and MARC Standards Office. http://lcweb.loc.gov/marc/

[5] Editeur: ONIX International. (n.d.). Release 1.2. http://www.editeur.org/onixfiles1.2/onixfiles.html

[6] Metadata Switch: A project of OCLC Research. (2002). http://www.oclc.org/research/projects/mswitch/index.shtm

[7] Web Services Activity. (2003). W3C Architecture Domain. http://www.w3.org/2002/ws/

[8] MARC 21 XML Schema: Official Web Site. (2003), Library of Congress. http://www.loc.gov/standards/marcxml/

[9] Metadata Standards Crosswalks. (2000). Getty Research Institute. http://www.getty.edu/research/institute/standards/intrometadata/3_crosswalks/index.html

[10] Metadata Advisory Group crosswalk list. (n.d.). MIT Libraries. http://macfadden.mit.edu:9500/metadata/crosswalks.html

[11] Metadata switch: schema transformation services. (2002). A Project of OCLC Research. http://www.oclc.org/research/projects/mswitch/1_schematrans.shtm

[12] XSL transformations. (1999). Version 1.0. World Wide Web Consortium. http://www.w3.org/TR/xslt

[13] IMS learning resource meta-data best practices and implementation guide. (2001). Version 1.1, Final Specification. http://www.imsproject.org/metadata/mdbestv1p1.html#Dublin

[14] Dublin Core (DC) to Encoded Archival Description. (2000). Getty Museum. http://www.getty.edu/research/institute/standards/intrometadata/3_crosswalks/index.html

[15] ONIX to MARC21 mapping. (2000). Network Development and MARC Standards Office. Library of Congress. http://www.loc.gov/marc/onix2marc.html#mapping

[16] GRUNK Overview. (n.d.). Digital Library Technologies. National Center for Supercomputing Applications. http://dlt.ncsa.uiuc.edu/grunk/

[17] Rust, G. and Mark Bide, M. (2000). The <in*d*ecs> metadata framework: principles, model and data dictionary. http://www.indecs.org/pdf/framework.pdf

[18] Hunter, J. and Lagoze, C. (2001). Combining RDF and XML schemas to enhance interoperability between metadata application profiles. Proceedings of the Tenth International World Wide Web Conference, Hong Kong. http://archive.dstc.edu.au/RDU/staff/jane-hunter/www10/paper.html

[19] Resource Description Framework. (2003). W3C: Technology and Society Domain. http://www.w3.org/RDF/

[20] Heery, R. and Wagner, H. (2002). A metadata registry for the Semantic Web. *DLIB Magazine*, 8(5). http://www.dlib.org/dlib/may02/wagner/05wagner.html

[21] I-Chene Tai. (2002). DC-GEM/MARC. OCLC Office of Research. http://www.oclc.org/research/projects/mswitch/1_gem-marc.shtm

[22] A GEM-MARC translation. (2003). OCLC Office of Research. http://purl.org/net/mswitch_gem_marc

[23] St. Pierre, M. and LaPlant, W. P., Jr. (1998). Issues in crosswalking metadata content standards. *NISO*. 1998. http://www.niso.org/press/whitepapers/crsswalk.html

[24] Wenger, E. (1997). *Communities of Practice: Learning, Meaning and Identity*. Cambridge University Press, 1997.

[25] Friesen, N. (2002). Semantic interoperability and communities of practice. http://www.cancore.ca/documents/semantic.html