The International Journal of Digital Curation
Issue 1, Volume 6 | 2011

What Constitutes Successful Format Conversion? Towards a Formalization of 'Intellectual Content'

C.M. Sperberg-McQueen,

Black Mesa Technologies LLC

Abstract

Recent work in the semantics of markup languages may offer a way to achieve more reliable results for format conversion, or at least a way to state the goal more explicitly. In the work discussed, the meaning of markup in a document is taken as the set of things accepted as true because of the markup's presence, or equivalently, as the set of inferences licensed by the markup in the document. It is possible, in principle, to apply a general semantic description of a markup vocabulary to documents encoded using that vocabulary and to generate a set of inferences (typically rather large, but finite) as a result. An ideal format conversion translating a digital object from one vocabulary to another, then, can be characterized as one which neither adds nor drops any licensed inferences; it is possible to check this equivalence explicitly for a given conversion of a digital object, and possible in principle (although probably beyond current capabilities in practice) to prove that a given transformation will, if given valid and semantically correct input, always produce output that is semantically equivalent to its input. This approach is directly applicable to the XML formats frequently used for scientific and other data, but it is also easily generalized from SGML/XML-based markup languages to digital formats in general; at a high level, it is equally applicable to document markup, to database exchanges, and to ad hoc formats for high-volume scientific data.

Some obvious complications and technical difficulties arising from this approach are discussed, as are some important implications. In most real-world format conversions, the source and target formats differ at least somewhat in their ontology, either in the level of detail they cover or in the way they carve reality into classes; it is thus desirable not only to define what a perfect format conversion looks like, but to quantify the loss or distortion of information resulting from the conversion.[1]

[1] This paper is based on the paper given by the authors at the 6th International Digital Curation Conference, December 2010; received December 2010, published March 2011.

The *International Journal of Digital Curation* is an international journal committed to scholarly excellence and dedicated to the advancement of digital curation across a wide range of sectors. ISSN: 1746-8256 The IJDC is published by UKOLN at the University of Bath and is a publication of the Digital Curation Centre.

SOME RIGHTS RESERVED

Introduction

It is widely (and plausibly) believed that preservation of digital objects over long periods will typically require repeated format conversions.[2] In many cases, as Lesk (1992) points out, these will simply involve copying the digital content from one type of storage medium to another, in a permanent attempt to outrun the obsolescence of one generation after another of data carriers and their associated hardware. I will call these *media conversions*. In other cases, less frequent but more dangerous to the integrity of the information, the format conversions will involve translation from one data format (or file format) to another, to outrun the obsolescence of the format and its associated software.[3] I will call these *substantive conversions* when it is necessary to distinguish them clearly from media conversions.

When either form of conversion is performed, it is desirable to ensure that the conversion is performed correctly and retains, as far as possible, the 'intellectual content' of the object. For media conversions (e.g., from floppy disk to CD-ROM), standard file-comparison routines typically suffice.[4] The conversion is successful if, and only if, the new copy is byte-for-byte similar to the original copy.[5]

But what constitutes a successful format conversion in the more complex case of substantive conversions, where the data format itself is being changed to keep the object accessible in a new hardware/software context? How do we test whether the intellectual content of the object has been preserved? How do we detect cases where it has not been preserved? Currently, the state-of-the-art appears to be: it's complicated, and each case involves human judgement. And in cases where it's not feasible for a human to examine each digital object before and after conversion, we check what we can and hope for the best.[6]

---

[2] Michael Lesk writes "Reformatting, instead of being a last resort as material physically collapses, will be a common way of life in the digital age" (Lesk, 1992).

[3] Alternatives are possible, of course. Format conversions may be avoided by keeping a digital object's entire hardware and software infrastructure in service indefinitely, or by periodically replacing portions of the infrastructure with emulators. The argument of this paper addresses issues which arise in migration, not emulation. The problems posed by obsolescence of emulators are similar, but harder to solve since emulation involves dynamic behavior and many systems have delicate timing dependencies.

[4] File comparison routines suffice only for objects whose digital realization takes the form, or can be regarded as taking the form, of a set of files in a file system. This is common enough to count as the usual case, even though historically many database management systems have not been stored in files accessible through the usual operating system file primitives. When the digital object's natural form does not consist of, or cannot be reduced to, a set of files, then either special arrangements must be made for testing the integrity of copies or else some file-based representation of the object must be used for purposes of preservation. For example, standard database management systems typically provide facilities for dumping a database to files, or loading a database from files, even when the live database is not file-based.

[5] Conversions that involve a change to the object's character set or other fundamental conventions of its representation will not produce byte-for-byte similar results, but they are best regarded as being substantive conversions, not media conversions.

[6] Jantz and Giarlo (2005) formulate the explicit question: "How does one know that the digital object viewed on the computer screen is a faithful reproduction of the original artifact?" but can only suggest visual inspection of the result. Moore et al. (2000) suggest encoding all digital objects in XML so as to be able to re-create equivalent objects in new technologies; this exploits the device- and application-independent semantics of XML, but leaves open the question asked by Jantz and Giarlo. This paper may be regarded as an attempt to fill that gap.

Would it be possible to provide a crisp operational definition of successful substantive format conversion? Could we automate the process of checking for degradation of information in substantive conversions, in the same way that we can automate the process of checking for failures in media conversions?

Recent work in the semantics of markup languages suggests that the answer to these questions is "yes", at least in principle; this paper describes that work and describes its relevance for the long-term preservation of digital objects. The following section proposes an operational definition of format conversion; the next discusses some of its obvious applications. The penultimate section describes some complications which arise when the simple idea in its pure form is confronted with real-world situations; the final section mentions some natural directions for further development of the idea.

Noise-Free Lossless Conversion

This section proposes an operational definition of correct format conversion, using conversions from one XML vocabulary to another as the frame of reference. Although not all digital formats are XML vocabularies, every XML vocabulary effectively defines a distinct digital format, so the terms *format* and *vocabulary* are used nearly interchangeably in the discussion that follows. Non-XML formats are described further below.

Markup-Language Semantics as Sets of Inferences

In a number of papers published over the last ten years (e.g., Sperberg-McQueen et al., 2001; Renear et al., 2003; Marcoux, 2006; Marcoux et al., 2009), students of markup theory have proposed to treat the meaning of markup in XML documents as the set of inferences licensed by occurrences of the markup in the document. This means, in practice: 1) that for each markup construct (e.g., for each element type in an XML vocabulary), one or more sentence schemata are specified; and 2) that for each instance of the construct, all the relevant sentence schemata are instantiated.

For example, the OAI-PMH *GetRecord* element is used in an OAI-PMH request to request a metadata record for a particular item in a particular format:

> "This verb is used to retrieve an individual metadata record from a repository. Required arguments specify the identifier of the item from which the record is requested and the format of the metadata that should be included in the record." (Lagoze et al. 2008)

In a response message, the presence of a *GetRecord* element indicates several things:

- The request which elicited this response used the verb *GetRecord*;
- The request raised no error and no exception (if an error had been raised, the response would contain an *error* element, not a *GetRecord* element);
- The item requested exists within the repository;

- The metadata format requested exists, or did exist, for the item in question (if the *record* element, which appears as the child of *GetRecord*, has `status="deleted"`, then the format did exist until the record was deleted; otherwise, it exists still and is given as the content of the *record* element).

These inferences might be expressed in first-order predicate logic,[7] as shown below, using *request-id*, *item-id*, *repository-id*, and *m-prefix* to denote identifiers assigned to the request, the item for which metadata was requested, the repository to which the request was directed and the metadata prefix specified in the request:

```
request_verb(request-id, "GetRecord")
∧errorfree(request-id)
∧isin_repository_item(repository-id, item-id)
∧hasformat_repository_item(repository-id, item-id,
m-prefix)
```

For each *GetRecord* element occurring within a normal OAI-PMH response, a sentence can be instantiated from this schema by replacing *request-id*, *repository-id*, etc., with suitable identifiers. Given a suitable semantic description of the vocabulary, it is possible to generate such instance sentences automatically from XML documents which use the vocabulary. In this way it is possible to make explicit the information conveyed indirectly or implicitly by conventional XML vocabularies.

For technical reasons, it is necessary to distinguish between the meaning of the XML markup, that is the set of inferences licensed by the markup, and the inferences actually enumerated in expressions like the one shown above. For any logic complete enough to be interesting, the full set of inferences licensed by any proposition P is infinite,[8] but for practical reasons it is undesirable to undertake to enumerate in full the members of any infinite set. It is preferable, therefore, to enumerate a finite set of sentences, such that every sentence in the infinite set follows logically from the sentences of the finite set. Note that any such finite set of sentences uniquely determines the infinite set of sentences, which contains all and only those sentences which follow logically from the sentences of the finite set. I will use the terms *licensed inferences* and *enumerated inferences* when it is necessary to stress the distinction.

A Definition of Intellectual Content and Lossless Conversion

I propose to define the intellectual content of a digital document, for preservation purposes, as the meaning of the document, which in turn is taken as the set of inferences licensed by the markup in that document.

If it is feasible to identify the meaning of a given document with the inferences licensed by the markup in that document, then it becomes possible to use mechanical methods to compare the meanings of documents. In particular, it is possible to compare the meaning of the document before format conversion with the meaning of the document produced as a result of format conversion.

---

[7] First-order predicate calculus is a convenient choice because it is well understood, reasonably powerful, and suitable for formal reasoning, but the essential idea is to use sentence schemata to describe the meaning of markup construct and use instance sentences to describe the meaning of instances of the markup construct in *some* convenient language. Alternatives to predicate calculus include other logical formalisms, RDF, and natural language.

[8] It includes, among others, the sentences $\neg\neg P$, $P \vee P$, $P \wedge P$, $\neg\neg\neg\neg P$, $P \vee P \vee P$, $P \wedge P \wedge P$, etc.

In the simplest imaginable case, where both the source vocabulary and the target vocabulary are described in terms of the same set of primitive notions (specifically the same sets of objects, relations and predicates), it might be possible simply to compare the sentences produced from the two documents: if the two documents produce different sets of inferences, then the meaning has changed.

In the more general case, however, the semantic descriptions of the two vocabularies may use predicates which differ either in the spelling of the name or in semantics. If the idea of converting an object from the source format to the target format makes any sense at all, however, the two must share at least part of their way of thinking and talking about the world, and it will be possible, with more or less effort and with more or less satisfactory results, to describe the primitives of each vocabulary in terms of the other. In simple cases, it may be possible to do so by simply stating that the *x* of the source vocabulary is exactly equivalent to the *y* of the target vocabulary, or that every *z* (in source-vocabulary terms) is (in terms of the target vocabulary) a *w*. More generally, such equivalences and subset/superset relations between vocabularies will depend, at least in part, on context; we can define the relation between vocabularies, as well as it is possible to do so, by specifying a set of inference rules which indicate when a particular state of affairs described in the source vocabulary licenses a particular inference in the target vocabulary, and vice versa. I will call the inference rules that map from the source to the target vocabulary the *ST inference rules*, and those that map in the other direction the *TS inference rules*. The enumerated inferences of the source and result documents, respectively, will be called the *S enumeration* and the *T enumeration*.

The comparison between the semantics of the source document and the semantics of the result document can then be performed as follows: for each sentence in the T enumeration, we ask “Does this sentence follow from the S enumeration, together with the ST inference rules?” If the answer is “no” for any sentence, then that sentence conveys information in the result document which was not present in the source document; the digital object has been contaminated (to a greater or lesser degree, depending on the importance of the information in question). If the answer is “yes” for each such sentence, then the format conversion has not introduced any spurious new information into the document; the conversion may be said to be *noise-free* (or equivalently *noiseless* or *non-noisy*).

The same question is then asked in the converse direction: for each sentence in the S enumeration, we ask “Does this sentence follow from the T enumeration, together with the TS inference rules?” If the answer is “no” for any sentence, then the conversion has lost information (specifically the information conveyed by the sentence in question). If the answer is “yes” for all sentences, then the conversion has lost no information and may be described as *lossless* or *non-lossy*.

It is now possible to state concisely the ideal goal for any format conversion: the conversion should be noise-free (it should introduce no new information) and lossless (it should lose no information). And equally, it is possible to test empirically whether a given conversion is, or is not, noiseless and or lossless.[9]

[9] Becker and Beck (2010) describe a very similar process they call “XML essence testing”. It differs from the procedure proposed here by using a carefully selected subset of the information in the document and by being handled by ad hoc transformations.

Applying the Definition

The primary advantage of the definition offered in the previous section is perhaps the clarity it tries to give to the notion of successful format conversion. It also suggests some operations which may usefully be undertaken (or at least considered) in connection with format conversions.

Proving the Correctness of a Transformation Process

Given the definition of noise-free, lossless transformation, it is possible to specify formally the desired behavior for a transformation and may be possible, in principle, to prove that a particular procedure correctly implements the specification. In the current state of knowledge, such proofs are probably not feasible for standard methods of XML processing. The problem is analogous to proving that an XSLT stylesheet will always produce schema-valid output if provided with schema-valid input; or analysing the input/output dependencies of XSLT stylesheets, problems which have thus far resisted solution.[10] This problem may be more tractable with regard for a particular transformation than with regard to the general problem of proving correctness for any arbitrary transformation, and of course it is possible that weaker methods of specifying transformations may make appropriate proofs easier to achieve. Proofs of transformation correctness will remain a topic of interest for the foreseeable future.

Testing the Conversion of a Digital Object

To test a given source document/result document pair for correctness of the conversion, two main processes are needed: 1) a process for generating a set of enumerated inferences for each document, and 2) a process for comparing two sets of enumerated inferences.

Ideally, a process for sentence generation will start from appropriate semantic descriptions of the vocabularies involved (see, for example, Marcoux et al., 2009) and use a general-purpose tool to apply those semantic descriptions to XML document instances and generate the enumerated inference sentences as a result. A less general approach (but one which requires less up-front investment) is to write ad-hoc transformations to transform XML documents in specific vocabularies into lists of sentences (for example, W3C, 2007; Sperberg-McQueen & Miller, 2004; and Dubin et al., 2003).

The process of comparing sets of sentences for logical equivalence will be simplest if the sentences generated by the enumeration process are simple ground facts about named individuals and objects in the world.[11] Whether this will always be so, for normal XML vocabularies, is an empirical question to which the answer is not known.

---

[10] Fokoue (2005), for example, reports on a straightforward context-insensitive analysis of XSLT stylesheets, which produces accurate results for a non-recursive stylesheet but “very inaccurate” results for recursive stylesheets on the same vocabulary. A more careful context-sensitive analysis produced better results, but suffers from worst-case exponential behavior. Møller and Schwartzbach (2004) also report some success with the typing problem.

[11] By “ground fact” I mean an assertion consisting of a simple predicate whose arguments are all literal values or identifiers, not references to quantified variables. For example, if *r1, r2,* and *i1* are identifiers denoting an OAI-PMH request, an OAI-PMH repository, and an item in that repository, respectively, and if “`dc`” was specified by the repository as a known metadata prefix, then the expression schema given above may yield the ground facts request_verb(r1, “`GetRecord`”), errorfree(r1), isin_repository_item(r1, i1), and hasformat_repository_item(r1, i1, “`dc`”).

But experience suggests that for many colloquial XML vocabularies, most or all enumerated inferences can indeed take the form of ground facts.[12]

For example, a Prolog system can be used to check a transformation in the following way:[13]

1. Generate the two enumerations as lists of Prolog facts;
2. Formulate the ST and TS inference rules as Prolog rules allowing new facts to be inferred. Since Prolog is a Turing-complete language, this does not limit the power of the inference rules;
3. To check purity, load the S enumerations and the ST inference rules, and treat each fact in the T enumeration as a Prolog goal (that is, ask the Prolog system to try to prove the fact from the information at its disposal). If the system succeeds for all such facts, the conversion is noise-free;
4. Check losslessness in the analogous way.

The formulation of strategies for correctness-checking using other technologies for representing logical assertions (e.g., SQL or W3C's Resource Definition Framework, RDF) is left as an exercise for the reader.

In all of these approaches, the comparison will be much simpler if the S and T enumerations use the same identifiers for objects whose existence can be inferred from the markup. Otherwise, a set of correspondences between the S identifiers and the T identifiers will have to be found, which in the general case will be expensive, as the number of possible mappings grows very quickly with the number of identifiers.

In the normal case, a file containing the enumerated inferences for a document is likely to be several times the size of the original document, and the comparison of the S and T enumerations tends to be correspondingly time consuming, at least for straightforward implementations of the comparison strategy. When digital objects are large or numerous, therefore, it may be thought impractical to test every transformation of every object in this way; in such cases, the comparison of enumerations may be a helpful testing procedure during the development of the transformation. During production use, the testing procedure may be applied to a random sample of objects from the larger population as a routine quality assurance measure.

Some Complications

In the interests of clarity and simplicity, the discussion above has left some real-world complications out of account. The following paragraphs touch briefly upon some of these complications.

[12] For the vocabularies for which formal semantic descriptions have thus far been attempted, the main obstacle to expressing all enumerated inferences as ground facts is the need to mint identifiers for otherwise anonymous entities mentioned in the formal expression of the facts.

[13] Prolog is a well known programming language intended to enable processes to be described in a purely or mostly declarative way; the name is formed from the French phrase "programmation en logique" ("programming in logic"). Since it is widely supported, Prolog offers a convenient way to operationalize at least some purely logical descriptions. Nothing in the argument, however, depends on any unique properties of Prolog; other computational logic systems can be used to perform the tasks described.

Not All Digital Objects are XML Documents

Even if one accepts the proposal to equate the intellectual content of a marked-up document with the inferences licensed by its markup, the fact remains that not all digital objects are XML documents, let alone XML documents in vocabularies for which suitable semantic descriptions exist. Can the basic concept be applied to other forms of digital objects? To proprietary formats? To relational databases?

Yes, it can.

For any digital object in any data format, it is possible to ask repeatedly "What does this mean? How does it mean it?" until, at length, an account of the propositional content of the data format begins to emerge. The methods described above have been developed first for declarative markup languages in the SGML/XML family, because the lack of pre-defined semantic primitives in SGML and XML (and in particular the systematic avoidance of a purely operational semantics based on software behavior) has given particular prominence to the topic in the XML context. However, they can be applied equally well to relational databases, or to proprietary data formats of arbitrary kinds.

For proprietary data formats, of course, it will be very difficult in practice to obtain reliable or authoritative answers to the questions "What exactly does this mean? Why?" It is correspondingly difficult, in practice, to be confident that any format conversion from, or into, a proprietary data format has been successful in preserving all the information in the source while avoiding adding spurious additional information.

Some Inferences Should Not Carry Forward

It is not unusual, in colloquial markup vocabularies, for some markup constructs to have metalinguistic significance: to identify the vocabulary used in the document, for example, or to specify which particular version of the vocabulary is in use (and thus just what semantic rules are to be applied in interpreting the markup). This complicates the story somewhat.

In HTML documents, for example, the document type declaration (and in particular the public identifier of the vocabulary) is conventionally used by those who wish to specify the particular version of HTML being employed, and the *meta* element allows the document to carry information about itself and about the HTTP context in which it is served. The enumerated inferences for an HTML document may thus contain the information that the document itself is represented in HTML 2, or served in a particular character set, or that it was last modified on a particular date in the past. But if we are converting a document last changed in 1996 from HTML 2 using a now elderly 7-bit national character set, for example, into a more current version of HTML, using UTF-8 encoding, then the enumerated inferences for the result document must either include the statement that the document is encoded in HTML 2 (which will be false), or else not include the statement (in which case the transformation is, strictly speaking, a lossy one).

In practice, therefore, strictly noise-free, strictly lossless transformations are not always desirable. There are certain kinds of information which, strictly speaking, we will almost always want to filter out of the source document, while injecting corresponding but different information into the result document. Some obvious categories of such information are:

- Metalinguistic information identifying the markup vocabulary (or version) used;
- Information about the processing history of the document (and in particular when it was last touched).

More generally, any self-referential information (i.e., any information provided by the document about the document itself) may be rendered false or incomplete by the action of converting the document to another format. False information will need to be suppressed, and incomplete information will need to be augmented, in the process of conversion. In practice, then, the desired goal for a formation conversion is not that *no information at all* be lost or added in the process of conversion, but that information be lost or added *only to the extent and in the manner intended*.

Different Levels of Detail

A second reason for a source document and result document to vary is that the source and target vocabularies may provide different levels of detail for some concepts.

If we are translating bibliographic descriptions from one vocabulary into another, for example, we may find that one vocabulary provides a detailed inventory of ways in which individuals or organizations may be involved with an item: author, editor, translator, illustrator, composer, performer, conductor, director, producer, publisher, distributor, etc., while the other makes do with the categories of creator, contributor, and publisher.

In such cases, it is inevitable that information will be lost in moving from the more complex to the simpler vocabulary, and in some cases (if the richer vocabulary has no generic term for contributor that does not specify the kind of contribution in more detail) that information will, at least apparently, be added when moving in the other direction. In some cases, up-conversions of this kind will choose some default type of contributor and translate all the occurrences of *contributor* in the simpler vocabulary into (for example) *editor* in the richer vocabulary. In some cases, some ad hoc annotation or other is added to indicate that these instances of *editor* are semantically deviant in that they really mean something more general; this is sometimes the only way to avoid abuse of the target vocabulary. If the target vocabulary provides no way of annotating particular instances of a classification or element type or field (here, the *editor* element) as being problematic or special in some way, then normal occurrences of the concept will be indistinguishable from cases where the prescribed semantics are being stretched.

Designers of data formats can help make these problems more manageable by:

- Systematically providing generic fallback elements, fields, or classes to handle cases where the finer-grained distinctions usually called for cannot be made, for some reason or another;[14]

[14] The TEI's *div*, *seg*, and *ab* (abstract block) elements provide this kind of fallback for textual material, as do the HTML *div* and *span* elements.

- Always providing ways to annotate portions of the digital object in unconstrained ways.

Those responsible for format conversion may not be in a position to influence the design of the target format. But those who choose which formats to target may be well advised to consider whether the target format makes it easier, or harder, to produce acceptable results, given the input at hand.

When a generic annotation mechanism is used to mark certain portions of the material as problematic, it becomes possible, if resources permit, to route the material to a special workflow designed to take care of the problem (e.g., by determining whether each contributor in the up-translated material was an author, an editor, a translator or something else); if resources do not permit the addition of the required extra information (and when large volumes of data are involved, they may not), then marking the material as problematic can allow automatic processes to work around the problem, or avoid making it worse. As the long-established principle of computer programming holds, the only thing worse than cutting a corner is cutting the corner and not marking the spot so that it can be fixed later, if appropriate, or so that later maintainers of the system at least know to tread carefully.

Different Ways of Carving the World

When vocabularies differ only in level of detail, as described in the previous paragraphs, the categories of the one vocabulary are simply supersets, or proper subsets, of the other's. Far more common are cases of the kind familiar from natural-language translation, where the concepts of one language do not map cleanly onto the concepts of the other, but overlap in idiosyncratic ways. It may be impossible to specify the ST or TS inference rules described above in such a way that automated systems can reliably produce correct results; sometimes error is inevitable in at least some cases, unless human intervention is feasible.

Here, too, the design of the target vocabulary can help. Broad, generic classes may be more likely to be strict supersets of classes in the source vocabulary, and thus safe target translations for problematic concepts. And the annotation of possibly problematic cases is equally important when the matchup between the world view of the source vocabulary and that of the target vocabulary differ in the way they carve reality up into categories.

Conclusions

It is possible, though currently still rather laborious, to provide formal semantic descriptions of important data format, to use these to define exactly what is meant by noise-free lossless format conversion, and to test the purity and lossiness of a conversion empirically. While noise-free, lossless conversion does not become any easier or more likely owing to this way of conceiving of the process, it does at least become easier to consider concretely the specific kinds of information loss and the specific kinds of information distortion to which substantive format conversions are prey.

Explicit semantic descriptions of XML vocabularies, relational database schemas and other data formats can make possible a wide variety of tools for making format conversions safer and more reliable. Tools for checking the conversion were described above. It may also be possible to create tools for identifying individual documents or other objects which are likely to need human intervention, or conversely, those which present no apparent problems for a purely automatic process and which can therefore be checked more quickly, allowing human resources to be spent on cases where they are more likely to be valuable.

Acknowledgements

The author thanks Professor Yves Marcoux and Professor Allen Renear for helpful discussions of the subject matter of this paper.

References

Becker, A., & Beck, J. (2010). XML essence testing. *Proceedings of Balisage: The Markup Conference 2010. Balisage Series on Markup Technologies, 5*. Retrieved August 9, 2010, from http://www.balisage.net/Proceedings/vol5/html/Becker01/BalisageVol5-Becker01.html.

Dubin, D., et al. (2003). A logic programming environment for document semantics and inference. *Literary & Linguistic Computing 18, (2).* Oxford, UK: Oxford University Press.

Fokoue, A. (2005). Extracting input/output dependencies from XSLT 2.0 and XQuery 1.0. *Proceedings of Extreme Markup Languages.* Montréal, Canada. Retrieved August 9, 2010, from http://conferences.idealliance.org/extreme/html/2005/Fokoue01/EML2005Fokoue01.html.

Jantz, R., & Giarlo, M.J. (2005). Digital preservation: Architecture and technology for trusted digital repositories. *D-Lib Magazine, 11, (6)*. Retrieved August 9, 2010, from http://www.dlib.org/dlib/june05/jantz/06jantz.html.

Lagoze, C., Van de Sompel, H., Nelson, M., & Warner, S., ed. (2008). *The Open Archives Initiative Protocol for Metadata Harvesting - Protocol Version 2.0 of 2002-06-14*. Open Archives Initiative. Retrieved July 28, 2010, from http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm.

Lesk, M. (1992). *Preservation of new technology: A report of the Technology Advisory Committee to the Commission on Preservation and Access.* Washington, D.C.: Commission on Preservation and Access. Retrieved August 1, 2010, from http://www.lesk.com/mlesk/cpa/cpa2.html and from http://www.clir.org/pubs/reports/lesk/lesk2.html.

Marcoux, Y. (2006). A natural-language approach to modeling: Why is some XML so difficult to write? *Proceedings of Extreme Markup Languages.* Montréal, Canada. Retrieved August 1, 2010, from http://conferences.idealliance.org/extreme/html/2006/Marcoux01/EML2006Marcoux01.html.

Marcoux, Y., Sperberg-McQueen, C. M., & Huitfeldt, C. (2009). Formal and informal meaning from documents through skeleton sentences: Complementing formal tag-set descriptions with intertextual semantics and vice-versa. *Proceedings of Balisage: The Markup Conference.* Retrieved August 1, 2010, from http://balisage.net/Proceedings/vol3/html/Sperberg-McQueen01/BalisageVol3-Sperberg-McQueen01.html.

Moore, R., et al. (2000a). Collection-based persistent digital archives. *D-Lib Magazine, 6, (3) and 6, (4)*. Retrieved August 9, 2010, from http://www.dlib.org/dlib/march00/moore/03moore-pt1.html and http://www.dlib.org/dlib/april00/moore/04moore-pt2.html.

Møller, A., & Schwartzbach, M.I. (2004). *The design space of type checkers for XML transformation languages.* BRICS (Basic Research in Computer Science) Report Series RS-04-34. Aarhus, Denmark: BRICS. Retrieved August 9, 2010, from http://www.brics.dk/RS/04/34/BRICS-RS-04-34.ps.gz.

Renear, A., Dubin, D., Sperberg-McQueen, C.M., & Huitfeldt, C. (2003). XML semantics and digital libraries. *JCDL '03: Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital Libraries.* Retrieved August 1, 2010, from http://portal.acm.org/ft_gateway.cfm?id=827192&type=pdf&coll=GUIDE&dl=GUIDE&CFID=99020226&CFTOKEN=85858933.

Sperberg-McQueen, C.M., Huitfeldt, C., & Renear, A. (2001). Meaning and interpretation of markup. *Markup Languages: Theory & Practice, 2, (3)*. Cambridge, MA: MIT Press.

Sperberg-McQueen, C.M., & Miller, E. (2004). On mapping from colloquial XML to RDF using XSLT. *Proceedings of Extreme Markup Languages.* Montréal, Canada. Retrieved August 9, 2010, from http://conferences.idealliance.org/extreme/html/2004/Sperberg-McQueen01/EML2004Sperberg-McQueen01.html.

W3C (World Wide Web Consortium). (2007). *Gleaning Resource Descriptions from Dialects of Languages (GRDDL) W3C Recommendation.* ed. D. Connolly. Cambridge MA, Sophia-Antipolis, & Tokyo: W3C. Retrieved August 9, 2010, from http://www.w3.org/TR/grddl/.