CES/XML : An XML-based Standard for Linguistic Corpora

Nancy Ide, Patrice Bonhomme, Laurent Romary

XCES:
An XML-based Encoding Standard for Linguistic Corpora

Nancy Ide*, Patrice Bonhomme†, Laurent Romary†

*Department of Computer Science, Vassar College
Poughkeepsie, NY 12604-0520 USA
ide@cs.vassar.edu
†LORIA (CNRS, INRIA)
Campus Scientifique - BP 239
54506 Vandoeuvre-lès-Nancy FRANCE
{bonhomme, romary}@loria.fr

Abstract

The Corpus Encoding Standard (CES) is a part of the EAGLES Guidelines developed by the Expert Advisory Group on Language Engineering Standards (EAGLES) that provides a set of encoding standards for corpus-based work in natural language processing applications. We have instantiated the CES as an XML application called XCES, based on the same data architecture comprised of a primary encoded text and "standoff" annotation in separate documents. Conversion to XML enables use of some of the more powerful mechanisms provided in the XML framework, including the XSLT Transformation Language, XML Schemas, and support for inter-rescue reference together with an extensive path syntax for pointers. In this paper, we describe the differences between the CES and XCES DTDs and demonstrate how XML mechanisms can be used to select from and manipulate annotated corpora encoded according to XCES specifications. We also provide a general overview of XML and the XML mechanisms that are most relevant to language engineering research and applications.

Introduction

The Corpus Encoding Standard (CES) (Ide, 1998a, b) is a part of the EAGLES Guidelines developed by the Expert Advisory Group on Language Engineering Standards (EAGLES). The CES is an application of SGML (ISO 8879:1986, Information Processing--Text and Office Systems--Standard Generalized Markup Language) compliant with the specifications of the *TEI Guidelines for Electronic Text Encoding and Interchange* of the Text Encoding Initiative. The CES is designed to be optimally suited for use in language engineering research and applications, in order to serve as a widely accepted set of encoding standards for corpus-based work in natural language processing applications. The standard specifies a minimal encoding level that corpora must achieve to be considered standardized in terms of descriptive representation (marking of structural and typographic information), provides a suite of DTDs for encoding basic document structure and linguistic annotation, and specifies a corresponding data architecture for linguistic corpora.

The eXtensible Markup Language (XML) is the emerging standard for data representation and exchange on the World Wide Web (Bray, Paoli, & Sperberg-McQueen, 1998). Although at its most basic level XML is a document markup language directly derived from SGML (i.e., allowing tagged text (elements), element nesting, and element references), various features and extensions of XML make it a far more powerful tool for data representation and access. For example, the eXtensible Style Language (XSL) provides a powerful transformation language (XSLT) (Clark, 1999) that can be used to convert any XML document into another document (either another XML document or a document marked with HTML, etc.) by selecting, rearranging, and adding information to it, in order to serve any application that relies on part or all of its contents. Also, XML's provision for accessing part or all of multiple DTDs in a single document provides an elegant means to represent and manipulate documents encoded according to the CES data architecture.

We have instantiated the CES as an XML application called XCES[1]. A primary motivation for this effort is to provide a state-of-the-art representation and access framework for the American National Corpus (see Macleod, Ide, & Grishman, 2000) as well as to serve the language engineering community as a whole. In this paper, we describe the differences between the CES and XCES, and demonstrate how XML mechanisms can be used to select from, manipulate, and transform corpora encoded according to XCES specifications. We also provide a general overview of XML and the XML mechanisms that are most relevant to language engineering research and applications.

XML Conversion of the CES

Minimally, conversion of the CES to XML requires the following:

- adaptation of the DTDs for XML compliance, principally by eliminating inclusion exceptions and making mixed-content models XML-compliant;
- adaptation of the CES mechanism for inter-document reference to meet the specifications of XML pointer and linking mechanisms..

However, we further exploit the capabilities of the XML framework to accomplish the following:

- validate the CES data architecture, in which linguistic annotations are maintained in separate documents that point back to the original, yielding a "hyper-document" composed of the original text and all annotations. This will enable us to ensure that the architecture and pointing mechanisms are conformant

[1] http://www.cs.vassar.edu/XCES.

to the specifications of mechanisms for manipulation of and access to XML documents, such as the XSL Transformation Language, XQL (Robie, et al., 1998), etc.

- exploit XML mechanisms for combining all or part of documents described by different DTDs, in order to create, for example, a new document containing only certain types of annotation and structural markup (e.g., markup for paragraphs, sentences, etc.). In particular, we will rely on XML "namespaces" and the ability to reference DTD fragments to retain the integrity (and validity *vis à vis* the original DTDs) of the newly formed documents.
- instantiate the DTDs using XML schemas (Thompson, et al., 1999), which, among other things, provides a means to limit element content by type (e.g., text only, numbers, etc.). In addition, XML schemas enable definition of data types (for example, a "part of speech" type, a "lemma" type, etc.) and specification of legal values; thus, precise values for tag contents describing linguistic phenomena can be defined and validated, thus reducing the need for extensive manual checking.

2.1 Adaptation of the CES DTDs

XML has been designed to eliminate some of the arcane and/or redundant syntax of SGML, in an effort to streamline SGML and enable easier parsing and processing. Conversion of the CES DTDs from SGML to XML conformance is relatively trivial, involving only a few minor syntactic changes, none of which affect the definitions of legal element content. For example,

- Attributes with types such as NAME and NAMES are changed to NMTOKEN and NMTOKENS.
- Default values for attributes must be quoted, e.g., `complete (y|n) "y"`.
- In mixed content models (i.e., elements whose content descriptions allow free mixture of text and elements) #PCDATA (meaning, in effect, text) must be the first in the list of allowed elements; e.g., a content model that allows text mixed with elements for numbers and abbreviations would be: `(#PCDATA | num | abbr)*`.
- The '&' connector is disallowed in content models; for the CES this meant simplifying the content model for the header element `respStmt` to `((respType | respName)+)`.

XML also disallows "inclusions" and "exclusions" in content models, i.e., specification of elements that either *must* or *must not* appear nested within the defined element in the document itself. The CES makes use of exclusions to disallow recursive nesting of certain elements; in particular, the elements `hi`, `foreign`, `distinct`, `mentioned`, and `title` are not allowed to be nested. Each of these elements is defined to be a member of a class of "phrase-level" elements (e.g., `foreign`, `mentioned`, `distinct`, `title`, `hi`, `list`, `corr`, `gap`, `reg`, `ptr`, `ref`) and their content models are defined to consist of members of this class inter-mixed with text. SGML DTD syntax provides a shorthand notation for indicating that a given element or elements in the content model may not appear, even though it is listed; XML requires an exact listing of allowed elements. The XCES DTDs therefore had to be modified to explicitly list allowed elements in the content models for `hi`, `foreign`, `distinct`, `mentioned`, and `title`.

2.2 Linking with XPointer and XLink

In the CES, primary documents (encoded using the cesDoc DTD) and annotations (encoded using the cesAna and cesAlign DTDs) are linked using a mechanism that enables identification of the elements and/or text content to be referenced and the document that contains these elements.

XML, like SGML, offers a mechanism for identifying pointer targets known as ID/IDREF. The mechanism works by including an ID attribute specifying a unique identifier on the element that is the target of the reference (i.e., the element is "localized"); an IDREF attribute on the source of the reference source the same identifier, thus providing a pointer to the target. However, the ID/IDREF mechanism presents certain problems for linking elements in linguistic corpora. First, the ID mechanism can be used only to point to another *tagged* element. Therefore, its use demands inserting ID attributes--and tags as well, if necessary--on every item that may possibly be a target. In linguistic corpora, it is not uncommon to require reference to parts of the text that may not be tagged; for example, if only sentences are tagged, the ID/IDREF mechanism does not enable referring to a specific word within the sentence unless the word itself is tagged and provided with an ID attribute. The addition of tags and IDs can be a substantial task, and may be impractical if the document will be modified frequently.

Another problem arises from the fact that the ID/IDREF mechanism allows references only within the same SGML/XML document. Because the data architecture of the CES provides for maintaining annotations and other related information (e.g., different versions of the text) in separate SGML/XML documents with different DTDs, the ID/IDREF mechanism is inappropriate for our use.

To answer these problems, XML provides an extended addressing syntax called the XML Path Language (XPath) (Clark & DeRose, 1999), which defines a concise notation for element localization in the document tree (as defined by the nesting of elements in the document itself). For example, the XPath expression `/div/p[2]/s[3]` specifies the third `<s>` (sentence) element within the second `<p>` (paragraph) element within each `<div>` (text division) element; `/descendant::p` specifies all `<p>` elements in the document. In addition, XPath allows addressing text fragments within a particular element by providing predicates for manipulating chains of characters. For example, the expression

```
substring(/p/s[2]/text(),6)
```

selects the string "one would expect that the whole sky would be as bright as the sun, even at night." from the following text:

```
<p><s id="d3p13s4">The difficulty is
that in an infinite static universe
nearly every line of sight would end
on the surface of a star.</s><s
id="d3p13s5">Thus one would expect
that the whole sky would be as bright
as the sun, even at night.</s></p>
```

Similarly, the expression

```
substring(/p/s[2]/text(), 10, 12)
```

selects "would expect". Thus the reference is made by specifying (1) the address (absolute or relative) of the element closest to the substring to be referred to, and (2) the substring within this element. Another XML mechanism, XPointer (DeRose, Daniel, & Maler, 1999) extends XPath syntax to allow addressing points and ranges as well as nodes, locating information by string matching, and use of addressing expressions in URI-references as fragment identifiers.

The pointer mechanisms in the SGML version of the CES are based on HyTime (ISO, 1992; DeRose & Durand, 1994) and TEI extended pointers (DeRose & Durand, 1995), the latter of which provided the basis for the development of XPath. The CES reference mechanism for identifying specific strings of characters utilizes two attributes, *from* and *to*, to identify the beginning and end points of the string, as well as a third attribute, *doc*, to specify the target document, if necessary; for example, :

```
<tok from="1.2\10" to="1.2\22">
```

This is shorthand for the HyTime/TEI expression:

```
<tok from="CHILD (1) (2) STRLOC (10)"
     to="CHILD (1) (2) STRLOC (22)">
```

XML's mechanism is more explicit and requires only one attribute; for example:

```
<tok
   xlink:href=
    "substring(/p/s[2]/text(),10,12)">
```

As this example shows, XML also provides a powerful mechanism for specifying a link (uni-directional or more complex linking structures) between two or more resources or portions of resources, called XLink (DeRose, et al., 2000). In XCES, this mechanism is used for alignment in cesAlign documents, to link corresponding segments of two or more primary texts. It is also used to link annotation documents to a base document containing the primary text, as in the example above where annotation information (e.g., morpho-syntactic information) about a specific token (`<tok>`) is linked to the string of characters in the original text to which it applies.[2] In addition to specifying the target location for information in the same or external documents, XLink attributes can be used to specify the role of the link, i.e., how the link should be activated (by hand, or automatically by the browser) and what to do with the target fragment (replace it or insert it into the source document).

Two of the CES DTDs use links extensively: the cesAna DTD for segmentation and morpho-syntactic annotation, and the cesAlign DTD for alignments between parallel texts. In these DTDs, the link element has the following attributes:

- *doc* for the address (URL) of the target resource. By the definitions in the CES, if *doc* is given on a parent element, it is inherited by all children elements, thereby avoiding repetition of the attribute. However, inheritance is not defined for SGML attributes and therefore not implemented in any SGML parser.
- *to* for the beginning of the annotated fragment, in terms of the ID on the <s> (sentence) tag and a token.
- *from* for the end of the annotated fragment.

The conversion of the CES into XML has modified this architecture. In XML, annotated fragments are referenced by the URI (remote or local) of the target resource, and an extended pointer identifying a element and, where necessary, the selected substring of that element's content, as in the following:

```
<tok
  xlink:href=
  "http://www.loria.fr/doc.xml#xptr
  (substring(/p/s[2]/text(), 10, 12))">
```

Annotation resulting from automatic processing (marking of sentence boundaries, tokens, links between parallel texts, etc.) often includes thousands of links to the same external document. Repetition of the document name on, for example, every `<tok>` element in a cesAna annotation document would obviously significantly multiply its size. XML includes an attribute *xml:base* (Marsh, 2000) that builds in to XML the inheritance specified for the CES *doc* attribute. For example, in the following text:

```
<chunk
  xml:base=
  "http://www.loria.fr/doc.xml#">
  <tok
      xlink:href="xptr(substring
       (/p/s[2]/text(), 10, 12))"/>
  <tok
      xlink:href="xptr(substring
       (/p/s[2]/text(), 24, 4))"/>
</chunk>
```

the value of the attribute *xml:base* specified on the `<chunk>` element is inherited by the two `<tok>` elements that are its children, and therefore need not be re-specified. The inclusion of `xml:base` in the XML specification ensures that conformant XML processors will handle it (unlike SGML).

Manipulating and Extracting from XCES Documents

The Extensible Style Language (XSL) is a part of the XML framework, consisting of two parts: the best known is the XSL formatting or "style sheet" language; and a powerful tree-traversal language, XSLT (Clark, 1999), that can be used to convert any XML document into another document in any form (e.g., XML, well-formed HTML, plain text, etc.). The transformed documents may or may not be intended for rendering data on a computer screen, but may be used simply to move data from one computer system or program to another (e.g., to transduce between encoding and/or annotation formats, etc.).

XSLT supports the following kinds of document manipulation:

- selection of elements or portions of element content using the XPath syntax;
- rearrangement or transformation of extracted information (including not only text content but also element names, etc.) in the target document;
- addition of information in the target document.

Thus, a suite of documents representing a base text (or texts) and its annotations can be manipulated to serve any application that relies on part or all of its contents. Thus

[2] Although at present we link only text, the mechanism provides for linking resources in any medium (audio, video, etc.), which in later versions of XCES will allow for linking speech, external images, video, applets, form-processing programs, style sheets, etc.

XSLT is likely to have the most to offer for manipulation of and access to annotated corpora.

XSLT is relatively complex and will not be described in detail here.[3] A short example can provide some idea of the possibilities. Using as input a cesAna document containing morpho-syntactic information (e.g., a document containing the fragment in Figure 1[4]), the XSLT document in Figure 2 can be used to create an HTML document that displays a text in "word | lemma | pos" form. When the resulting HTML document is loaded into a browser, it will display the following:

```
It|it|PPER3 was|be|PAST3 a|a|DINT
bright|bright|ADJE cold|cold|ADJE
day|day|NN…
```

The XSLT script in Figure 2 could be modified to produce output in any desired form, or to produce another XML document containing the merged text and annotation documents. Similarly, XSLT can be used to produce concordances, paired sentences or words from a parallel text, or even a web document that displays the orthographic representation of a text and provides the audio rendition when the word is clicked on, etc. The XCES web page[5] provides additional examples of XSLT scripts and their output. Also, Ide, Kilgarriff, & Romary (2000) describe an XML format for encoding lexical information (primarily drawn from dictionaries) and demonstrate how XSLT can be used to implement an inheritance mechanism over the document tree.[6]

We include in our presentation a demonstration using the publicly-available XT tool (Clark, 1999), showing how combination of and selection from base and annotation documents can be accomplished, and the various presentation and formatting options XSLT can effect. In particular, we demonstrate application of XCES and the use of XT and other XML tools to corpora including extensive morpho-syntactic information, and aligned documents (potentially either text or speech).

```
<?xml version="1.0">
<chunk type="BODY" lang="en"
 xml:base=
"http://www.cs.vassar.edu/~ME/Oen.xcesDoc#">
 <par xlink:href="xptr(substring(//p[1]">
  <s xlink:href="xptr(substring(//p/s[1]">
   <tok type="WORD"
     xlink:href=
     "xptr(substring(//p/s[1]/text(),1,2">
      <orth>It</orth>
      <disamb>
        <base>it</base>
        <msd>Pp3ns</msd>
        <ctag>PPER3</ctag></lex>
      <lex>
        <base>it</base>
        <msd>Pp3ns</msd>
        <ctag>PPER3</ctag></lex></tok>
    <tok type="WORD"
      xlink:href=
      "xptr(substring(//p/s[1]/text(),4,2">
      <orth>was</orth>
      <disamb>
        <base>be</base>
        <msd>Vmis3s</msd>
        <ctag>PAST3</ctag></lex>
      <lex>
        <base>be</base>
        <msd>Vais1s</msd>
        <ctag>AUX1</ctag></lex>
      <lex>
    <base>be</base>
    <msd>Vais3s</msd>
    <ctag>AUX3</ctag></lex>
  <lex>
    <base>be</base>
    <msd>Vmis1s</msd>
    <ctag>PAST1</ctag></lex>
  <lex>
    <base>be</base>
    <msd>Vmis3s</msd>
    <ctag>PAST3</ctag></lex></tok>…
```

Figure 1 : Fragment of a cesAna document

```
<xsl:stylesheet version="1.0"
   xmnls:xsl=
   "http://www.w3.org/1999/XSL/Transform">

<xsl:template match= "/">
     <html>
       <body>
           <xsl:apply-templates/>
       </body>
     </html>
</xsl:template>

<xsl:template match="//par"/>

  <xsl:for-each select="//tok"/>
     <xsl:value-of select="orth"/>
     <xsl:text>|</xsl:text>
     <xsl:value-of select="disamb/base"/>
     <xsl:text>|</xsl:text>
     <xsl:value-of select="disamb/ctag"/>
  </xsl:for-each>
</xsl:template>

</xsl:stylesheet>
```

Figure 2 : XSLT document to create HTML output

XML Schemas

The XML Schema definition language (Thompson, et al., 2000; Biron & Malhotra, 2000) enables document creators to constrain and document the meaning, usage and relationships of the constituent parts of XML documents: datatypes, elements and their content, and attributes and their values. Schemas can also be used to provide default values for attributes and elements. As such, XML schemas provide means to define an abstract data model for a class of documents. While duplicating (or making explicit) some of the capabilities provided by XML DTDs, they significantly extend their power and provide for much tighter validation of document form and content.

XML schemas have considerable implications for development of XCES and for corpus encoding and

[3] Full documentation is available at http://www.w3.org/TR/xslt.

[4] Note that this cesAna document contains full sementation and annotation information, including full morpho-syntactic specifications for all potential annotations and the results of automatic disambiguation.

[5] http://www.cs.vassar.edu/XCES

[6] See also Erjavec *et al.* in this volume.

annotation in general. The following lists only a few possibilities for the application of XML schemas in XCES:

- different attribute declarations and/or content models can apply to elements with the same name in different contexts. This allows for more tightly constrained content models than possible with DTDs. For example, names in headers (names of authors, etc., consisting of the usual "first name", "last name" elements) and names in the text ("named entities") should have different content models and attributes in order to provide for tight validation of form in each context. In the TEI, upon which the CES is based, the element `<name>` is used in both headers and text, and its content model is necessarily broad enough to encompass the variety of forms it may have in these contexts. In the CES, header elements are prefixed with "h." so that names in headers are tagged with `<h.name>`, whose content model is different from that of the `<name>` element that can appear in the body of the text. This strategy is effectively a "kludge" to overcome the fact that SGML provides no scoping capabilities. XML schemas, building on definitions using XML Namespaces (Bray, Hollander, & Layman, 1999), solves this problem. Thus in XCES, we avoid the invention of variant element names while retaining the ability to constrain content and attributes based on context.
- equivalence classes can be defined for groups of elements and/or attributes, indicating that they may be used in the same ways as defined for a particular named element ("the exemplar"). The CES makes extensive use of parameter entities to group together elements that behave identically. For example, phrase-level elements (i.e., elements that can appear within but not outside paragraphs or paragraph-like elements, such as name, num, etc.) are grouped using the parameter `%phrase.seq`, so that all paragraph-level elements can include this class in their content models. Again, this is a work-around for the fact that equivalence and inheritance of properties is not expressable in SGML. Similarly, groups of attributes are defined in all CES DTDs, as in the cesANa DTD fragment given in Figure 3. In XCES, this is replaced by the schema in Figure 4.
- attribute or element values, or combinations of attribute and element values, can be constrained to be unique. That is, it is possible to indicate in a computational lexicon that only one entry can be defined with the value of a given word form as its content (or the content of one of its child elements), that only one paragraph can have an attribute indicating that it is the 23rd, or in general that a given key appears only once in a document. Similarly, we can ensure that only one disambiguated form is given for each token in a cesAna document, or only one correspondence for a given sentence in a cesAlign document. Obviously, this is useful for error detection and prevention.
- dependencies can be established based on values of elements or attributes. This has similar benefits for error detection in creating annotated corpora: nouns can be prevented from being assigned a tense, tokens whose *type* attribute has the value PUNCT can be specified to include only `<orth>` elements containing specific characters, etc. More generally, annotation labels (e.g., pos indicators) used in an annotation document can be specified elsewhere, and element content can be constrained to these values only; for example, to constrain the values of the `<msd>` element in an XCES annotation document to the EAGLES morphosyntactic specifications (Monachini & Calzolari, 1996), the following could be specified:

```
<element name="msd">
    <simpleType base="string">
      <enumeration value="Pp3ns"/>
      <enumeration value="Vmis3s"/>
      <enumeration value="Vais3s"/>
      <enumeration value="Vmis1s"/>
      ...
    </simpleType>
  </element>
```

```
<!ENTITY % a.global '
        id              ID          #IMPLIED
        n               CDATA       #IMPLIED
        xml:lang        CDATA       #IMPLIED
        lang            IDREF       #IMPLIED'  >

<!ENTITY % a.ana '%a.global;
        type            CDATA       #IMPLIED
        wsd             CDATA       #IMPLIED'
        >

<!ELEMENT cesAna        (cesHeader?, chunkList)>
<!ATTLIST cesAna        %a.ana;
        doc             CDATA           #IMPLIED
        version         CDATA           #REQUIRED>
```

Figure 3 : cesAna DTD fragment for global attributes

```
<schema>
   <attributeGroup name="globalAtt">
     <attribute name="id" type="ID"
           maxOccurs="1" minOccurs="0"/>
     <attribute name="n" type="NMTOKEN"
           maxOccurs="1" minOccurs="0"/>
     <attribute name="lang" type="IDREF"
           maxOccurs="1" minOccurs="0"/>
   </attributeGroup>
   <attributeGroup name="anaAtt">
     <attribute name="type" type="string"
           maxOccurs="1" minOccurs="0"/>
     <attribute name="wsd" type="string"
           maxOccurs="1" minOccurs="0"/>
   </attributeGroup>
   <element name="cesAna">
     <complexType>
      <element name="cesHeader" minOccurs="0"/>
      <element name="chunkList" minOccurs="1"/>
      <attributeGroup ref="globalAtt"/>
      <attributeGroup ref="anaAtt"/>
      <attribute name="doc" maxOccurs="1"
           minOccurs="0"/>
      <attribute name="version" maxOccurs="1"
           minOccurs="1"/>
    </complexType>
  </element>
</schema>
```

Figure 4 : XCES schema for global attributes

XML Schemas have been developed for all three of the current XCES DTDs, and are available through the XCES web site.

Conclusion

The XML framework provides search, extraction, and transformation capabilities that answer most, if not all, of the current and foreseen needs for corpus-based language engineering. In particular, XML provides mechanisms for easily implementing the CES and XCES data architecture, which calls for modularization of resources by putting different kinds of annotation, different versions of the text and annotations, etc. in separate, linked documents. In addition, processing tools for the various XML recommendations (XPath, XPointer, XLink, etc.) are generally freely distributed, thus eliminating the need for costly and time-consuming tool development.

The CES and XCES encoding specifications have been developed for and by the language engineering community, and their coverage will continue to evolve. At present, XCES provides guidelines for encoding various features in written text, morpho-syntactic annotation, and alignment information, all of which are relatively stable and agreed-upon within the community. We are currently working with several different groups to implement encoding guidelines for additional written text features, computational lexicons, discourse and dialogue, and co-reference, as well as speech and its various levels of annotation and representation. Our development effort will continue to be based on the principle of collaborative and distributed development in a bottom-up fashion, building up the specifications as need and agreement within the community dictate. We welcome all input to the continuing development of the XCES.

Acknowledgments

This work was partially funded by the National Science Foundation and the Centre National de la Recherche Scientifique, under the NSF/CNRS International Collaborative program.

References

Biron, P. & Malhotra, A., 2000. XML Schema Part 2: Datatypes. W3C Working Draft, 25 February 2000. http://www.w3.org/TR/xmlschema-2/.

Bray, T., Hollander, D., Layman, M., 1999. Namespaces in XML. World Wide Web Consortium Recommendation, 14 January 1999. http://www.w3.org/TR/REC-xml-names/.

Bray, T., Paoli, J., Sperberg-McQueen, C.M. (eds.), 1998. Extensible Markup Language (XML) Version 1.0. W3C Recommendation. http://www.w3.org:TR/1998/REC-xml-19980210.

Clark, J. (ed.), 1999. XSL Transformations (XSLT). Version 1.0. W3C Recommendation. http://www.w3.org/TR/xslt.

Clark, J. and DeRose, S., 1999. XML Path Language (XPath). Version 1.0. W3C Recommendation. http://www.w3.org/TR/xpath.

Clark, J., 1999. XT Version 1991105. http://www.jclark.com/xml/xt.html

DeRose, S & Durand, D., 1994. *Making HyperMedia work: A users's guide to HyTime.* Boston., MA: Kluwer Academic Publishers.

DeRose, S & Durand, D., 1995. The TEI Hypertext Guidelines. In N. Ide & J. Véronis (eds.), *The Text Encoding Initiative: Background and Context.* Dordrecht: Kluwer Academic Publishers, 181-90.

DeRose, S, Maler, E., Orchard, D., Trafford, B. (eds.), 2000. XML Linking Language (XLink). W3C Working Draft, 21 February 2000. http://www.w3.org/TR/xlink.

DeRose, S., Daniel, R., & Maler, E., 1999. XML Pointer Language (XPointer). W3C Working Draft, 6 December 1999. http://www.w3.org/TR/xptr.

Erjavec, T., Evans, R., Ide, N., Kilgarriff, A., 2000. The CONCEDE model for Lexical Databases. In *Proceedings of the Second International Language Resources and Evaluation Conference*, this volume.

Ide, N., 1998a. Encoding Linguistic Corpora. In *Proceedings of the Sixth Workshop on Very Large Corpora*, 9-17.

Ide, N., 1998b. Corpus Encoding Standard: SGML Guidelines for Encoding Linguistic Corpora. In *Proceedings of the First International Language Resources and Evaluation Conference*, 463-70.

Ide, N., Kilgarriff, A., Romary, L., 2000. A Formal Model of Dictionary Structure and Content. In *Proceedings of EURALEX'00*, to appear.

ISO, 1992) ISO 10744: 1992. *Information technology -- Hypermedia/time-based Structuring Language (HyTime).* Geneva: ISO.

Macleod, C., Ide, N., Grishman, R., 2000. Progress Report on the American National Corpus. In *Proceedings of the Second International Language Resources and Evaluation Conference* (this volume). Paris: European Language Resources Association.

Marsh, J., 2000. XML Base (XBase). W3C Working Draft 21-February-2000. http://www.w3.org/TR/xmlbase.

Monachini, M. & Calzolari, N., 1996. Synopsis and Comparison of Morphosyntactic Phenomena Encoded in Lexicons and Corpora: A Common Proposal and Applications to European Languages. EAGLES Report EAG-CLWG-MORPHSYN/R. http://www.ilc.pi.cnr.it/EAGLES96/morphsyn/.

Robie, J., Lapp, J., Schach, D., 1998). XML Query Language (XQL). http://www.w3.org/TandS/QL/QL98/pp/xql.html

Thompson, H., Beech, D., Maloney, M. Mendelsohn, N., 2000. XML Schema Part 1: Structures. W3C Working Draft, 25 February 2000. http://www.w3.org/TR/xmlschema-1/.