Action Plan Background: XML 1.0

Author: Andrea Goethals, FCLA
Release Date: 06/09/03

Change History:
07/07/2003 Added to the locations within an XML document that references to external files can appear in Section *3.8.2 External References*.
02/01/2008 Added section 2.2 “Internal Technical Metadata”.

Preface: The W3C (World Wide Web Consortium) publishes different types of documents differentiated by their level of consensus within the W3C and whether or not they have the Director's stamp of approval. For a specification to be considered ready for implementation, it must be of type 'Recommendation'. A W3C document can progress from 'Working Draft', to “Last Call Working Draft', to 'Candidate Recommendation', to 'Proposed Recommendation', and finally to Recommendation. All W3C document types, except Recommendations, are open to revision [W3C, 2001]. The XML 1.0 specification is a W3C Recommendation. The XML 1.0 specification describes XML documents and partially how computer programs must process them.

1 General Description

1.1 Format Name: XML (eXtensible Markup Language)

1.2 Version: 1.0

1.3 MIME media type name: text or application
According to [Murata et.al, 2001], the mime-type application/xml is applicable to all XML documents. The mime-type text/xml is only applicable to XML documents that can be treated as plain text. These latter documents are restricted to particular text encodings: i.e. UTF-8, but not UTF-16 (except in the case of HTTP).

1.4 MIME subtype: xml
Note: XHTML has its own registered mime type: application/xhtml+xml

1.5 Short Description: A text markup language, a meta-language and a file format; a subset of SGML (Standard Generalized Markup Language) [ISO-8897]

1.6 Common Extensions: .xml, .xsd (XML Schema), .xsl or .xslt (XML Stylesheets), .xhtml or .xht or .html (XHTML documents)

1.7 Color depth: Not applicable

1.8 Color Space: Not applicable

1.9 Compression: Not applicable

1.10 Progressive Display: Not applicable

1.11 Animation: Not applicable

1.12 Magic number(s):

There is no guarantee that an XML 1.0 document can be identified as an XML 1.0 document. [Murata, et.al., 2001] states that:

> **Although no byte sequences can be counted on to always be present, XML MIME entities in ASCII-compatible charsets (including UTF-8) often begin with hexadecimal 3C 3F 78 6D 6C ("<?xml"), and those in UTF-16 often begin with hexadecimal FE FF 00 3C 00 3F 00 78 00 6D 00 6C or FF FE 3C 00 3F 00 78 00 6D 00 6C 00 (the Byte Order Mark (BOM) followed by "<? xml").**

XML files usually start with either an XML declaration (also called XML Prolog), which unfortunately is not required by the XML 1.0 specification, or an encoding signature, also known as the byte order mark (BOM) - (see section *7 Glossary of Terms*). If an XML file is in UTF-16, the BOM precedes the XML declaration. If an XML document uses a character encoding other than UTF-8 or UTF-16, and no encoding was determined by a higher-level protocol (MIME Header, etc.), the XML declaration is required [W3C, 2002].

The XML declaration may optionally include an XML version attribute (<?xml version="1.0"?>), which is also not required by the specification. If the XML version number is not specified it can be assumed to be XML 1.0 because there are not any other versions of XML yet.

<u>Recognizing XHTML files</u> [Baker and Stark, 2002] states that "while there is no single initial byte sequence that is always present for XHTML files", there are some guidelines that can be followed that help in the identification:

> **All XHTML documents will have the string "<html" near the beginning of the document. Some will also begin with an XML declaration which begins with "<?xml", though that alone does not indicate an XHTML document. All conforming XHTML 1.0 documents will include an XML document type declaration with the root element type 'html'. XHTML Modularization provides a naming convention by which a public identifier for an external subset in the document type declaration of a conforming document will contain the string "//DTD XHTML". And while some XHTML based**

languages require the doctype declaration to occur within documents of that type, such as XHTML 1.0, or XHTML Basic (http://www.w3.org/TR/xhtml-basic), it is not the case that all XHTML based languages will include it. All XHTML files should also include a declaration of the XHTML namespace. This should appear shortly after the string "<html", and should read 'xmlns="http://www.w3.org/1999/xhtml"'.
Source: [Baker and Stark, 2002]

2 Content and Features

2.1 Essential and Distinguishing Characteristics

XML is a text format usually encoded in Unicode. All 'XML processors' (see section *7 Glossary of Terms*) must accept the UTF-8 and UTF-16 encodings of ISO/IEC 10646. An XML document can be encoded in any character encoding as long as the character set is identified in the XML declaration (<?xml encoding='EUC-JP'>), otherwise it is assumed to be UTF-8 or UTF-16 (See section *1.12 Magic Numbers* and section *7 Glossary of Terms* for distinguishing between UTF-8 and UTF-16 in XML document).

Logically, an XML document is composed of units called entities that may refer to other entities. The document contains exactly one document entity, or “root”, which contains all other entities in the document. XML documents must be 'well-formed', which means that the document must adhere to three requirements (See Section 2.1 of [Bray et. al., 2000]). Additionally, a well-formed document may or may not be 'valid'. A valid XML document is one that has an associated Document Type Declaration (DTD) and the XML document complies with the DTD's constraints on the document [Bray et. al., 2000]. Note that the XML 1.0 specification does not mention XML Schemas. Some authors use the phrase 'Schema-valid' to denote an XML document that complies to the constraints of an associated XML Schema.

Physically, an XML document is composed of text: either character data or markup. The markup takes the form of start-tags, end-tags, empty-element tags, entity references, character references, comments, CDATA section delimiters, document type declarations, processing instructions, XML declarations, text declarations and whitespace outside the document entity. Character data is anything else that isn't markup. The valid characters in XML documents are any Unicode character, excluding the surrogate blocks (see [Gillam, 2003]), FFFE, and FFFF. The function of the markup in an XML document is to describe its storage and logical structure and to associate attribute-value pairs with its logical structures [Bray et. al., 2000].

2.2 Internal Technical Metadata

Technical metadata element (G = general file metadata, F = format specific metadata)	*Obligation (R = required by spec., O = Optional but described in spec., D=derived from spec.)*
Character set used in the document (G)	O
Character set origin - the institution determining the character set used in the document (G)	D

Technical metadata element (G = general file metadata, F = format specific metadata)	*Obligation (R = required by spec., O = Optional but described in spec., D=derived from spec.)*
Name of the schema used for the root element (F)	D
Type of schema used by the XML (Ex: W3C_XML_SCHEMA, DTD, UNKNOWN) (F)	D
Whether or not the XML document was validated and if so, whether it is valid (F).	D

3 Usefulness

3.1 Version Duration: 5 years, 2 months and continuing

3.2 History of Prior Versions Duration:
This is the first version of XML, although the current XML 1.0 specification (published on October 6, 2000) is the second edition of the specification. The first edition of the XML 1.0 specification (published on February 10, 1998), is now obsolete. The second edition is nothing more than the errata to the first edition combined with the first edition for reading convenience.

3.3 Expected Newer Versions:
The XML 1.1 specification (Oct. 15, 2002) is currently a W3C "Candidate Recommendation". It will update XML to use Unicode 3, support checking of normalization, and to follow the Unicode line ending rules more closely.

3.4 Existence of Publicly Available Complete Specifications: The XML Specifications are viewable and freely downloadable in several formats from the W3C website (http://www.w3.org/XML/Core/#Publications).

3.5 Specifications-controlling Body: World Wide Web Consortium (W3C), which is composed of a 'Team' (about 60 researchers and engineers, most of whom work at the US MIT, the French ERCIM headquarters, or the Japanese Keio University), 'Members' (408 corporations and institutions), a Technical Architecture Group (TAG), an Advisory Board, and a director (Tim Berners-Lee).

3.6 Related Legal Issues:
There are no known legal issues related to the use of the formats and technologies described in the W3C's XML-related Recommendations. On May 20, 2003, the W3C

published the "W3C Patent Policy". It is meant to ensure that the W3C's Recommendations can be implemented on a royalty-free basis. It requires W3C members participating in the development of W3C Recommendations to disclose related patent claims. It should help ensure that patents do not slow down or prevent implementations of the W3C's XML Recommendations.

A different legal issue emerges when digital archives ingest documents (such as schema) that are referenced by the digital objects that they are archiving. It is somewhat straightforward to obtain copying and format migration permissions for the original digital objects but more complicated to get permission for the related files that are possibly owned by a different individual or organization. While it is not necessary to include a copyright statement in XML documents (such as XML Schema) to make them copyrighted, some authors are already doing this. An example of an existing copyright statement in an XML Schema document is shown in Figure 1. For digital archives that ingest schema associated with XML documents that they are archiving, it is unclear what the implications of these copyrights are and what if any permission is needed by the archive. In any case, all attempts should be made by a digital archive implementation to respect the intellectual rights of schema authors. The FCLA digital archive will seek copy and format migration permission from authors of schema for the schema that it is known will be frequently associated with digital objects ingested into the archive, so that these schema can be archived. This is probably a more difficult issue for archives that allow public access to the digital objects. The FCLA's digital archive is a 'dark archive' - there is no public access to the archived files.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.2 U (http://www.xmlspy.com) by Jerome McDonough (private) -->
<!-- METS: Metadata Encoding and Transmission Standard -->
<!-- Copyright © 2001, 2002, 2003 Digital Library Federation -->
...
```

Figure 1: A copyright statement appearing in an XML Schema document. Source: The METS 1.2 Schema from the Library of Congress' METS webpage (http://www.loc.gov/standards/mets/).

3.7 Application and Platform Support: Many databases can now import and export XML, and it is becoming the native format of choice for office applications (StarOffice, OpenOffice, Microsoft Office 11). PDF 1.4 contains some XML internally. There are many proprietary and open-source XML parsers, processors, editors and application servers.

Web browsers provide varying support for XML and its related technologies. A popular web browser that supports many XML technologies is Mozilla (and Netscape 6 and later which is built on top of the Mozilla engine). Mozilla 1.3 fully supports core XML, XHTML, XPointer Framework, XSLT and XPath; and partially supports XML Base, and XLink[1] [Toivonen, 2003]. Microsoft Internet Explorer 6.0 SP 1 supports XPath 1.0 and XSLT 1.0 [Microsoft, 2002] [Skonnard, 2000]. Opera 7.x supports XHTML, XML display,

---

1For the full list of XML technologies supported by Mozilla see http://www.mozilla.org/newlayout/xml/

and XML namespaces, but intentionally does not support XSLT or XSL Formatting Objects [Opera, 2003].

3.8 Limitations:

3.8.1 Complexity. The XML 1.0 specification started out as a 25-page 'stripped down' version of the SGML standard. Because it was made so 'simple', its functionality was limited. Because of this, it was extended by many other specifications and technologies, including XLink, XSL, Namespaces, Infoset, XML Linking, XPointer Framework, XPointer namespaces, XPointer xptr(), XSLT, XPath, XSL FO, DOM, Sax, stylesheet linking PI, XML Schema, XQuery, XML Encryption, XML Canonicalization, XML Signature, DOM Level 2 and DOM Level 3. The consequences of these extensions is that XML technologies are now described in separate documents, totaling hundreds of pages. This makes XML confusing and complex and XML tools more difficult to implement. The original designers of XML are aware of the complexity problem but have not come up with solutions yet [Hollander and Sperberg-McQueen, 2003]. Table 1 shows the continuing evolution of the XML infrastructure. There are several current XML-related Candidate Recommendations, including XML Events, Namespaces in XML 1.1, XForms 1.0, XML 1.1, and XML Inclusions (XInclude) 1.0. The recommendation to use 'IRIs' instead of URIs (from Namespaces in XML 1.1), and XInclude (an alternative to the external entities of DTDs) will be of particular interest to digital archive applications. It is expected that many of these XML Recommendations will become part of the next major XML Recommendation [Wilde and Lowe, 2003].

XML Recommendation	***Date Published***
XML 1.0	February 1998
XML Namespaces	January 1999
Associating Stylesheets with XML Documents	June 1999
XSLT 1.0	November 1999
XPath 1.0	November 1999
XML Schema	May 2001
XLink 1.0	June 2001
XML Base	June 2001
XML Information Set	October 2001
XML-Signature	February 2002
XML Encryption	December 2002
XML Pointer	March 2003

Table 1: The Publication Date of XML's Related Recommendations.

3.8.2 External References. XML files can internally contain references to other external files (schema, etc.) in the form of URIs and entity references (see section *7 Glossary of Terms*). At the present there is no mechanism to ensure that these references will always be able to access the files they refer to, because files move, change or become otherwise unattainable. URNs have

been suggested as the solution to this persistence problem [Wilde and Lowe, 2003], but there is little support for them yet.

For an archive to ensure that current XML documents are usable in the future, the externally-referenced files must be obtained and stored locally, if possible, because of the persistence problem. Once the files are local to the archive, care must be taken to keep track of these other files as well as the original XML file. In addition, the URIs of the external files as specified in the XML file must be made consistent with the URIs/names of the now local files, at least when displaying the XML file. Finally, care needs to be taken upon dissemination to ensure that the original XML file (or its migrated form) will be usable with the migrated forms of its externally-related files.

Because users of XML can create their own schema, it is not possible for an automated program to be able to 'recognize' all references to external files within an XML document. There are infinite ways that a schema creator could invent to specify externally-related files. However, it is relatively easy for a program to recognize the finite number of methods that the XML 1.0 Recommendation and its related XML Recommendations suggest for specifying references to external links. These include methods to specify DTDs, XML Schema, and non-schema files.

If the schema for an XML document is an external DTD, the DTD's address (URI reference) will be referred to in the XML document's document type declaration, which must appear before the first element in the document. The document type declaration syntax is partially described in Figure 2. Two examples of its use are shown in Figure 3. Note that if the XML declaration contains a *standalone='yes'* attribute name/value pair, then any references to an external DTD must be ignored.

```
document type declaration:
doctypedecl ::= '<!DOCTYPE'  S Name (S  ExternalID)?  S?
                ('[' (markupdecl  |  DeclSep)* ']'  S?)? '>'

white space:
S ::= (#x20 | #x9 | #xD | #xA)+

externalID:
ExternalID ::= 'SYSTEM'  S SystemLiteral |
                'PUBLIC'  S PubidLiteral S SystemLiteral
```

Figure 2: Some of the productions in EBNF (Extended Backus-Naur Form) notation needed to recognize the address of an external DTD. The position in the declaration where the DTD URI would be located is underlined. For the complete production set refer to [Bray et. al., 2000].

```
<?xml version="1.0"?>
<!DOCTYPE greeting SYSTEM "hello.dtd">
<greeting>Hello, world!</greeting>

---------------------------------------------------------------

<?xml version="1.0"?>
```

```
<!DOCTYPE greeting PUBLIC  "-//Examples//Hello world example//EN"
      "http://www.examples.com/hello.dtd">
<greeting>Hello, world!</greeting>
```

Figure 3: Examples showing the use of the document type declaration tag to specify the address of an external DTD. The underlining was added for emphasis and would not actually appear this way in an XML document.

There are two attributes used to specify the address of an XML Schema document from within an XML document, *noNamespaceSchemaLocation* and *schemaLocation*. These attributes are in the XMLSchema instance namespace (http://www.w3.org/2001/XMLSchema-instance). This namespace is usually prefixed with 'xsi', by convention, but could be prefixed other ways. These attributes are usually located in the root element of the XML document, although these attributes can occur on almost any elements [Thompson et. al., 2001]. Figures 4 and 5 show the use of xsi:noNamespaceSchemaLocation and xsi:schemaLocation, respectively.

```
<book isbn= "0836217462"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="file:library.xsd">
```

Figure 4: An example from [Thompson et. al., 2001] showing the use of the noNamespaceSchemaLocation attribute in the XMLSchema instance namespace to specify the address of an XML Schema document.

```
<book isbn= "0836217462" xmlns="http://example.org/ns/books/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://example.org/ns/books/
 file:library.xsd">
```

Figure 5: An example from [Thompson et. al., 2001] showing the use of the schemaLocation attribute in the XMLSchema instance namespace to specify the address of an XML Schema document.

XML Linking Language (XLink) is an XML technology developed specifically for creating and describing associations between XML documents. It can be used to create simple uni-directional links like those used in HTML ('simple'), or much more complex types of links ('extended'). There is not much support yet for XLink's extended links, but XLink's simple links are commonly used and are supported by several web browsers, including Mozilla (and Netscape 6 and higher), the W3C's Amaya, and IE 6.0 SP2. The Library of Congress' METS schema uses XLink's simple links (see http://www.loc.gov/standards/mets/).

To use XLink's elements and attributes, an XML document declares the XLink namespace. The prefix 'xlink' is used by convention to represent the XLink namespace, but other prefixes could be used. The XLink attribute that specifies the address of an external file is 'href'. An example of its use is shown in Figure 6.

```
<myElement xmlns:xlink="http://www.w3.org/1999/xlink">
```

```
    <relatedFile xlink:type="simple"
                 xlink:href="http://www.examples.com/otherFile.xml">
        <title>Example XML file</title>
        <description>A file used as an example of XLink.</description>
        <author>Joe Doe</author>
    </relatedFile>
    <relatedFile xlink:type="simple"
                 xlink:href="http://www.examples.com/image.jpg">
        <title>Example image</title>
        <description>A picture taken by me.</description>
        <author>Joe Doe</author>
    </relatedFile>
</myElement>
```

Figure 6: An example showing the use of the href attribute in the XLink namespace to specify the address of an externally-related file.

Entity references refer to entities named in an associated DTD (external or internal DTD). They do not pose a problem for archives because it is assumed that any external DTDs will be obtained by reading the DTD address in the document type declaration tag. The entity associated with the entity reference in the XML document can then be read from the DTD. Similarly, XPath and XPointer do not pose any additional file retrieval burdens on an archive because XPath and XPointer refer to specific parts of XML documents that are located by other XML technologies, usually XLink.

When a DTD associated with an XML document is internal (included within the document type declaration), there can be external references within the internal DTD's general or parameter entity declarations, or within notation declarations. Non-XML documents, such as images, can also be linked to as external unparsed entities, using a combination of the notation and entity declarations. An example of an XML document linking to an image using this technique is shown in Figure 7.

```
<?xml version="1.0"?>
<!DOCTYPE dog [
<!NOTATION JPEG SYSTEM "Joint Photographic Experts Group">
<!ENTITY colliepic SYSTEM "lassie.jpg" NDATA JPEG>
<!ELEMENT dog EMPTY>
<!ATTLIST dog picfile ENTITY #REQUIRED>
]>
<dog picfile="colliepic"/>
```

Figure 7: An example of an XML document that links to a non-XML document (lassie.jpg). The example is taken from [ISUG, 2002]. This example does not use the notation declaration syntax the way that the W3C intended, as "Joint Photographic Experts Group" is not a URI.

XML Schema documents can refer to other XML Schemas using three elements in the XMLSchema namespace: import, include and redefine. All three use the attribute

schemaLocation to point to the location of the Schema. An example taken from the METS schema (http://www.loc.gov/standards/mets/mets.xsd) is shown in Figure 8.

```
<xsd:schema targetNamespace="http://www.loc.gov/METS/"
        xmlns="http://www.loc.gov/METS/"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:xlink="http://www.w3.org/TR/xlink" elementFormDefault="qualified"
        attributeFormDefault="unqualified">
        <xsd:import namespace="http://www.w3.org/TR/xlink"
                schemaLocation="xlink.xsd"/>
...
```

Figure 8: An example of the use of the schemaLocation attribute within the import element in the XMLSchema namespace to link to an external XML Schema document.

An XML document can associate one or more external Cascading Styles Sheets with the XML document by using processing instructions whose target is xml-stylesheet. An href attribute points to the Style Sheet's URI. This processing instruction must come before the first tag of the document. The XSLT 1.0 Recommendation does not explicitly specify a means to refer to an external XSLT stylesheet from within an XML document. Instead it recommends that XSL processors support the same mechanism described here for CSS, for XSLT stylesheets. An example using CSS is shown in Figure 9.

```
<?xml-stylesheet href="common.css" type="text/css"?>
<?xml-stylesheet href="modern.css" title="Modern" media="screen" type="text/css"?>
<?xml-stylesheet href="classic.css" alternate="yes" title="Classic" media="screen, print"
type="text/css"?>
<ARTICLE>
<HEADLINE>Fredrick the Great meets Bach</HEADLINE>
        <AUTHOR>Johann Nikolaus Forkel</AUTHOR>
        <PARA>
                One evening, just as he was getting his
                <INSTRUMENT>flute</INSTRUMENT> ready and his musicians were
                assembled, an officer brought him a list of the strangers who had arrived.
        </PARA>
</ARTICLE>
```

Figure 9: An example showing the recommended way to reference an external Cascading Style Sheet from an XML document. The example is taken from http://www.w3.org/Style/styling-XML.

XSLT Stylesheets may refer to other XSLT Stylesheets using the href attribute of the import or include elements, within the XSLT namespace. The prefix 'xsl' is the common prefix for the XSLT namespace, but others could be used. The *xsl:include* and *xsl:import*

elements are only allowed as top-level elements (see [Clark, 1999b] for more information on this). An example of using the *xsl:import* element is shown in Figure 10. The use of the *xsl:include* element could be illustrated by substituting the word 'import' with 'include' in Figure 10.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
        <xsl:import href="article.xsl"/>
        <xsl:import href="bigfont.xsl"/>
        <xsl:attribute-set name="note-style">
                <xsl:attribute name="font-style">italic</xsl:attribute>
        </xsl:attribute-set>
</xsl:stylesheet>
```

Figure 10: An example of an XSLT stylesheet referring to another XSLT stylesheet by using the import element. The example is taken from [Clark, 1999b].

XPath is very different from the other XML technologies because it is not written in XML. It is primarily used to address parts of an XML document, similar to the way fragment identifiers and the NAME attribute of the HTML 'A' element are used to create links within an HTML page. XPath includes a core set of functions. Adding to the already complex set of XML technologies, the XSLT 1.0 recommendation adds functions to XPath's core functions. One of these functions, the document function, allows XSLT stylesheets to access external XML documents. An example of its use is shown in Figure 11.

```
<xsl:variable name=”xmlDoc” select=”document('data.xml')”/>
```

Figure 11: An example of using the document function to include an XML document in an XSLT stylesheet.

XInclude 1.0 is still a W3C Candidate Recommendation, although it already has many full and partial implementations (see http://www.w3.org/XML/2002/09/xinclude-implementation). XInclude specifies how to include an XML (or plain text) document within another XML document. It provides a way to merge XML documents. It differs from XLink because XLink can be used to refer to an external file of any media type, not just XML. The href attribute of the include element within the XInclude namespace refers to the URI of the included XML file. See Figure 12 for an example of its use.

```
<?xml version='1.0'?>
<document xmlns:xi="http://www.w3.org/2001/XInclude">
        <p>120 Mz is adequate for an average home user.</p>
        <xi:include href="disclaimer.xml"/>
</document>
```

Figure 12: An example of XInclude's method of referring to external XML files from within an XML document. The example is taken from [Marsh and Orchard, 2002].

3.9 Perceived Popularity:

XML use has become pervasive in the information technology sector for data sharing, especially by database developers, transaction designers, system engineers, and B2B (business-to-business) developers. XML is only now being explored as a medium for data preservation [ICTU, 2002].

4 Related Formats

4.1 Specification Variations:

There are many 'companion' specifications to XML, including XML namespaces, the XML Information Set, XSL Transformations (XSLT), XML Schema, and XML Linking. The W3C's website (http://www.w3.org/) is the authoritative site housing these specifications.

4.2 Other Specifications used by this Format:

Unicode and ISO/IEC 10646 for characters, Internet RFC 1766 for language identification tags, ISO 639 for language name codes, and ISO 3166 for country name codes

5 Summary and Conclusions

XML is system-independent and non-proprietary. The creators of the XML 1.0 specification designed XML so that data owners would not be locked into proprietary software for creating or manipulating XML.

For the purposes of preservation, XML documents are essentially plain text files, with a few differences. The character set may be explicitly stated. This is an advantage over plain text files, for being able to correctly interpret its contents in the future. However, unlike plain text files, XML documents may contain references to external files that must be accessed in order to fully interpret the XML document's contents. Because of the persistence problem that currently exists for files on the Internet, archives should retrieve and archive locally any files associated with an XML document that would cause information loss if they couldn't be accessed in the future. Not all associated files will be accessible, even at the time when the original XML file is ingested, either because the file is no longer available at the specified address, or because the reference is not easily recognized as a reference to a file by an automated program. Some associated files can be retrieved for the archive by an automated program by recognizing conventional ways to reference external files from within an XML document, as discussed in *Section 3.8.2 External References*.

6 References

[Alvestrand, 1995] Alvestrand, H., ed. “RFC 1766: Tags for the Identification of Languages”, IETF (Internet Engineering Task Force), 1995.

http://www.ietf.org/rfc/rfc1766.txt

[Baker and Stark, 2002] Baker, M.; and P. Stark. “The 'application/xhtml+xml' Media Type”, Request for Comments: 3236, The Internet Corporation for Assigned Names and Numbers (IANA), January 2002.
http://www.rfc-editor.org/rfc/rfc3236.txt

[Bray et. al., 1998] Bray, Tim; Jean Paoli, C. M. Sperberg-McQueen. “Extensible Markup Language (XML) 1.0”, W3C Recommendation; February 10, 1998.
http://www.w3.org/TR/1998/REC-xml-19980210

[Bray et. al., 2000] Bray, Tim; Jean Paoli, C. M. Sperberg-McQueen, and Eve Maler. “Extensible Markup Language (XML) 1.0 (Second Edition)”, W3C Recommendation; October 6, 2000.
http://www.w3.org/TR/2000/REC-xml-20001006

[Bray et. al., 2002] Bray, Tim; Dave Hollander, Andrew Layman, and Richard Tobin. “Namespaces in XML 1.1”, W3C Candidate Recommendation, December 18, 2002.
http://www.w3.org/TR/2002/CR-xml-names11-20021218/

[Clark, 1999a] Clark, James. “Associating Style Sheets with XML Documents Version 1.0”, W3C Recommendation, June 29, 1999.
http://www.w3.org/1999/06/REC-xml-stylesheet-19990629/

[Clark, 1999b] Clark, James. “XSL Transformations (XSLT) Version 1.0”, W3C Recommendation, November 16, 1999.
http://www.w3.org/TR/1999/REC-xslt-19991116

[Cowan, 2002] Cowan, John. “Extensible Markup Language (XML) 1.1”, W3C Candidate Recommendation. October 15, 2002.
http://www.w3.org/TR/2002/CR-xml11-20021015/

[Cowan and Tobin, 2001] Cowan, John and Richard Tobin. “XML Information Set”, W3C Recommendation; October 24, 2001.
http://www.w3.org/TR/2001/REC-xml-infoset-20011024/

[DeRose, et. al., 2001] DeRose, Steve; Eve Maler, and David Orchard. “XML Linking Language (XLink)”, W3C Recommendation; June 27, 2001.
http://www.w3.org/TR/2001/REC-xlink-20010627/

[Fallside, 2001] Fallside, David C. (editor). “XML Schema Part 0: Primer”, W3C Recommendation, May 2, 2001.
http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

[Gillam, 2003] Gillam, Richard. “Unicode Demystified: A Practical Programmer's Guide to the Encoding Standard”, Addison-Wesley: Boston; 2003.

[Hollander and Sperberg-McQueen, 2003] Hollander, Dave; and C. M. Sperberg-McQueen. “Happy Birthday, XML!”, On-line article, W3C website, February 10, 2003.
http://www.w3.org/2003/02/xml-at-5.html
[ICTU, 2002] ICTU, “XML and Digital Preservation”, Digital Preservation Testbed White Paper, ICTU: Den Haag, Netherlands; September 2002.

[ISUG, 2002] International SGML/XML Users' Group. “Graphics in SGML and XML”, 2002.
http://www.isgmlug.org/graphics.html

[Marsh, 2001] Marsh, Jonathan. “XML Base”, W3C Recommendation, 2001.
http://www.w3.org/TR/2001/REC-xmlbase-20010627/

[Marsh and Orchard, 2002] Marsh, Jonathan; and David Orchard. “XML Inclusions (XInclude) Version 1.0”, W3C Candidate Recommendation, September 17, 2002.
http://www.w3.org/TR/2002/CR-xinclude-20020917/

[Microsoft, 2002] Microsoft Corporation. “Microsoft Internet Explorer Features”, September 9, 2002.
http://www.microsoft.com/windows/ie/evaluation/features/default.asp

[Murata et.al., 2001] Murata, M.; S. St. Laurent, and D. Kohn. “XML Media Types”, Request for Comments: 3023, The Internet Corporation for Assigned Names and Numbers (IANA), January 2001.
http://www.rfc-editor.org/rfc/rfc3023.txt

[Opera, 2003] Opera Software ASA. “Web Specifications Supported in Opera 7”, Website, 2003.
http://www.opera.com/docs/specs/

[Skonnard, 2000] Skonnard, Aaron. “The XML Files: MSXML 3.0 supports XPath 1.0, XSLT 1.0, XDR, and SAX2”, MSDN Magazine, Microsoft Corporation, September 2000.
http://msdn.microsoft.com/msdnmag/issues/0900/xml/default.aspx

[Thompson et. al., 2001] Thompson, Henry S.; David Beech, Murray Maloney, and Noah Mendelsohn. “XML Schema Part 1: Structures”, W3C Recommendation, May 2, 2001.
http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[Toivonen, 2003] Toivonen, Heikki. “XML in Mozilla”, Webpage, The Mozilla Organization, April 3, 2003.
http://www.mozilla.org/newlayout/xml/

[Unicode, 2002] Unicode, Inc. “UTF & BOM”, Webpage, Unicode, Inc., 2002.
http://www.unicode.org/unicode/faq/utf_bom.html

[W3C, 2001] W3C, “W3C Process Document: 5 Technical Reports”, W3C; July 19, 2001.
http://www.w3.org/Consortium/Process-20010719/tr
[W3C. 2002] W3C HTML Working Group, “XHTML 1.0 The Extensible HyperText Markup Language (Second Edition): A Reformulation of HTML 4.0 in XML 1.0”, W3C Recommendation; January 26, 2000, revised August 1, 2002.
http://www.w3.org/TR/2002/REC-xhtml1-20020801/

[Weitzner, 2003] Weitzner, Daniel J., ed. “W3C Patent Policy”, W3C Policy, May 20, 2003.
http://www.w3.org/Consortium/Patent-Policy-20030520.html

[Wilde and Lowe, 2003] Wilde, Eric; and David Lowe. “XPath, XLink, XPointer, and XML: A Practical Guide to Web Hyperlinking and Transclusion”, Addison-Wesley: Boston; 2003.

7 Glossary of Terms

See [Bray et. al., 2000] and [Cowan and Tobin, 2001] for additional terms.

BOM - byte order mark; it is a mechanism for determining the 'endianness' of an XML document. It is not really necessary for UTF-8, but it is necessary for UTF-16 and UTF-32. If an XML document is in UTF-16 encoding, then its BOM specifies whether it is in UTF-16BE (big-endian) or UTF-16LE (little-endian). The BOM is the Unicode code point FE FF (ZERO WIDTH NO-BREAK SPACE character). If an application reads the BOM as FF FE (an illegal Unicode character) then the endianness is the opposite of the operating system's endianness that the application is running on. Some applications (i.e. Microsoft Notepad) write the BOM for UTF-8 files. In all cases, the BOM is FE FF translated to the transformation format, so the BOM is EF BB BF for UTF-8. See http://www.w3.org/TR/REC-xml#sec-guessing for more on interpreting BOMs.

entity reference - Found in an XML document, refers to the content of a named entity; delimited by a starting ampersand ('&') and a closing semicolon (';'). The named entity matches a name in an entity declaration of an associated DTD.

meta-language - A language used to create other vocabularies, i.e. SGML and XML

URI - Short for Uniform Resource Identifier, the generic term for all types of names and addresses that refer to objects on the World Wide Web. A URL is one kind of URI. [Source; webopedia.com]

URN - Short for Uniform Resource Name, refers to the subset of URIs that are required to remain globally unique and persistent even when the resource ceases to exist or becomes unavailable. [Wilde and Lowe, 2003]

XML processor - A software program that reads XML documents and provides access to their content and structure