TeXML: Typesetting XML with TeX

Douglas Lovell
IBM Research
P.O. Box 704
Yorktown Heights, NY 10598
dcl@us.ibm.com

Abstract

XML, eXtensible Markup Language, is a simplified subset of SGML, which is fast becoming a standard for content management on the internet.

TeXML is an XML vocabulary for TeX. A processor written in the Java programming language translates TeXML-conforming XML into TeX. The processor provides a document formatting solution for XML that leverages the rich knowledge and capability built over many years in TeX.

This describes the TeXML document format and the processor, TeXMLatté, that produces TeX source from TeXML markup.

XML: The future of the Web

The World Wide Web is moving toward a future in which XML, not HTML, is the primary medium for storing and delivering documents and data.

HTML is presentation markup for browsers. Together with Cascading Style Sheets (CSS) it specifies the type sizes and fonts, and layout of a document.

XML is a standard for defining and sharing data on the Web, including Web content. Users of XML may define vocabularies — sets of tags or element names, such as "title," "section," "citation" and attributes such as "id" — to identify elements and structures in a document. With XML, organizations can develop markup which captures the structural and semantic properties of their documents and data. Instead of writing, `<H1>Typesetting XML </H1>` we can write, `<title>Typesetting XML </title>`.

Trade organizations, companies, and scientific or educational institutions may define and share XML vocabularies to freely exchange data with specific meaning. MathML (W3C, 1998), the vocabulary for writing and exchanging mathematical formulas and expressions, is one example.

In the commercial world, XML is a key technology for the widespread implementation of rapid, accurate, meaningful, automatic transactions and data exchange on the internet.

What is XSL?

XSL, eXtensible Stylesheet Language, is a W3C draft recommendation (XSLWorking Group W3C, 1998) which began life as an XML vocabulary for typesetting. XSL is influenced by the DSSSL standard (ISO/IEC, 1996).

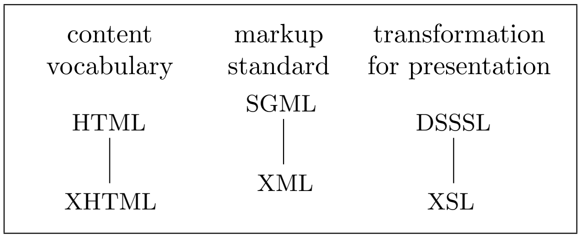

Figure 1: XML heritage

DSSSL is an ISO standard for typesetting SGML. The XSL effort began as a means for transforming XML markup into standard presentation markup, just as DSSSL did for SGML. A goal for XSL was to write a shorter, more perspicuous standard in the spirit of XML. Some of the key people who worked on DSSSL are key people working on XSL.

XSL does two things for XML:

1. It provides a language for specifying transformations from one XML vocabulary into another. The XSL specification calls this "Tree Construction."

2. It defines a set of XML elements, called "formatting objects", for encoding a typeset document specification in XML.

Figure 1 diagrams the heritage of XML and XSL from SGML and DSSSL. HTML is an SGML vocabulary which is migrating to an XML vocabulary, XHTML. XML is a content markup standard with ancestry in SGML. XSL is a formatting and transformation standard for XML with ancestry in DSSSL.

Figure 2 shows a sample of some XML markup describing an artist, taken from an application for an art museum. For more about XML see IBM/XML (1999) and W3C (1999).

What is TeXML?

TeXML is an XML vocabulary for representing TeX source. It is a medium for transforming any XML data into a document which can be typeset with the TeX program. TeXML represents the TeX commands, control symbols, and `\special`s as XML elements. Figure 3 shows the markup of Figure 2 represented in TeXML.

The potential for TeX and XML

XML makes TeX a potential universal typesetting back-end for markup in a way it never achieved for SGML. There is an opportunity for TeX to marry itself to XML, the future of the WWW, and become a predominant technology for typesetting.

TeX has many advantages. It has an established, stable implementation and user base. It has a rich body of knowledge and experience captured in its macro packages and styles. It can typeset just about anything.

The missing piece is a small implementation of TeX in the Java programming language which can be plugged into a browser or executed on a server; however, much of LaTeX can be displayed by IBM techexplorer (Sutor and Díaz, 1998).

The enabling technologies now available are TeXML and XSL, with platform-specific implementations of TeX or with the IBM techexplorer plug-in for an HTML browser.

XSL Tree Construction. The transformation part of XSL has been moving rapidly toward completion as a W3C recommendation. There are many implementations of the XSL transformation rules. In the process, people have found uses for XSL transforms far beyond producing typeset documents.

There are numerous web servers busily transforming XML into HTML using XSL style sheets. Microsoft is supplying the transform as a dynamic link library in their operating system. The Microsoft Internet Explorer Web browser (v.5) can accept XML data, apply an associated XSL transform, and display the result. Organizations use XSL style sheets to transform electronic data from another organization into a form suitable for internal use.

XSL has thus become an important tool for XML transformation. It is on the verge of becoming a *de facto* standard for XML transformations.

Figure 4 shows an XSL stylesheet that will transform the artist XML example in Figure 2 into the TeXML example in Figure 3.

XSL Formatting Objects. The formatting objects (FO) part of XSL is progressing more slowly. The specification of FO elements and the structure of those elements is far behind the transformation part as of this writing (March, 1999). This means that, while there is a "standard" way to transform XML into HTML, there is presently no standard way to produce typeset output from XML. TeXML was developed partly out of impatience to fill this gap left by the lagging FO effort. It marries the transformation function of XSL with the typesetting function of TeX. It provides a means for typesetting XML documents.

How to use TeXML

TeXML consists of two parts. The first part is the document type declaration (DTD), which defines XML that is valid TeXML. The second part is the translator program TeXMLatté, written in the Java programming language. TeXMLatté reads a valid TeXML file and writes a proper TeX file. The TeXML DTD appears as Figure 5.

The basic template for a TeXML document is:

```
<?xml version="1.0"?>
<TeXML>
  ... your content here ...
</TeXML>
```

The following sections describe how the DTD and TeXMLatté interact, and how to code TeX in TeXML.

Encoding commands. The TeXML `<cmd>` element encodes TeX commands.

1. To write a command with no parameters, such as `\par`, write `<cmd name="par"/>`.

2. To add parameters to a command, add `<parm>` children[1] to the `<cmd>` element. TeXMLatté places `<parm>` children within TeX groups, that is, curly braces.

3. To add options to a command, add `<opt>` children to the `<cmd>` element. TeXMLatté

[1] XML elements embedded within an enclosing element are often referred to as that element's "children," e.g. `<parent><child/></parent>`.

```
<?xml version='1.0'?>
<artist>
 <id>Weston</id>
 <fullName>Edward Weston</fullName>
 <lastName>Weston</lastName>
 <firstName>Edward</firstName>
 <birthYear>1886</birthYear>
 <deathYear>1958</deathYear>
 <portrait>Weston.jpg</portrait>
 <bio>Weston was a photographer who pioneered the modern use of photography
  as an art form in the United States.</bio>
 <longQuote1>One does not think during creative work, any more than one thinks
  when driving a car.  But one has a background of years - learning,
  unlearning, success, failure, dreaming, thinking, experience, all
  this - then the moment of creation, the focusing of all into the
  moment.  So I can make 'without thought,' fifteen carefully
  considered negatives, one every fifteen minutes, given material
  with as many possibilities.  But there is all the eyes have seen in
  this life to influence me.</longQuote1>
 <shortQuote1>My own eyes are no more than scouts on a preliminary search,
  for the camera's eye may entirely change my idea.</shortQuote1>
 <shortQuote2>Ultimately success of failure in photographing people depends
  on the photographer's ability to understand his fellow man.</shortQuote2>
 <soundQuote>Weston.wav</soundQuote>
</artist>
```

Figure 2: XML markup describing an artist

places <opt> children within square braces, as LaTeX style options.

As an example, the TeX code

```
\documentclass[12pt]{letter}
```

will look like this in XML:

```
<cmd name="documentclass">
  <opt>12pt</opt>
  <parm>letter</parm>
</cmd>
```

The TeXML DTD allows free-form text, commands, control symbols, and `\specials` as children of <parm> and <opt> elements. It does not allow <parm> or <opt> elements to nest, except as children of a nested <cmd>. It does not allow environments within a <parm> or an <opt>.

Encoding control symbols. The <ctrl> element encodes a control symbol, such as <ctrl ch=" "/> for a control space. Use the <cmd> element to encode control words.

Encoding specials. TeXMLatté "escapes" any and all TeX `\specials` which occur in the XML source. This means that backslashes, percent signs, dollar signs, and the like are properly escaped by the time they get to TeX. The writer of XML will not intend those characters to be special. Figure 6 gives a full list of the characters escaped by TeXMLatté, along with their replacement text.

TeXML	TeX
%	\%{}
&amp;	\&{}
{	\{
}	\}
|	$\|$
\	$\backslash$
$	\${}
#	\#{}
_	_{}
^	\char‘\^{}
~	\char‘\~{}
&lt;	$<$
&gt;	$>$

Figure 6: Characters escaped by TeXMLatté

Use the <spec> element to encode any TeX `\specials` when you really want them to occur as `\specials` in the TeX file output by TeXMLatté.

The <spec> element is always empty; that is, it never has anything within it. Encode the category

```
<TeXML>
 <cmd name="documentclass">
  <parm>article</parm>
 </cmd>
 <cmd name="title">
  <parm>Edward Weston</parm>
 </cmd>
 <env name="document">
  <cmd name="maketitle"/>
  <cmd name="section*">
   <parm>Some biographical information</parm>
  </cmd>
  Edward Weston
  was born in
  1886.
  Weston
  lived until
  1958.
  <cmd name="par"/>
  Weston was a photographer who pioneered the modern use of photography
  as an art form in the United States.
  <cmd name="section*">
   <parm>Some short quotes from Weston</parm>
  </cmd>
  <env name="enumerate">
   <cmd name="item"/>
   ‘‘My own eyes are no more than scouts on a preliminary search,
   for the camera’s eye may entirely change my idea.’’
   <cmd name="item"/>
   ‘‘Ultimately success of failure in photographing people depends
   on the photographer’s ability to understand his fellow man.’’
  </env>
  <cmd name="section*">
  <parm>A longer quote</parm>
  </cmd>
  <env name="quote">
   ‘‘One does not think during creative work, any more than one thinks
  when driving a car.  But one has a background of years - learning,
  unlearning, success, failure, dreaming, thinking, experience, all
  this - then the moment of creation, the focusing of all into the
  moment.  So I can make ’without thought,’ fifteen carefully
  considered negatives, one every fifteen minutes, given material
  with as many possibilities.  But there is all the eyes have seen in
  this life to influence me.’’
  </env>
 </env>
</TeXML>
```

Figure 3: TEXML markup derived from the XML of Figure 2

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">

<xsl:template match="artist">
 <TeXML>
  <cmd name="documentclass"><parm>article</parm></cmd>
  <cmd name="title">
   <parm><xsl:apply-templates select="fullName"/></parm>
  </cmd>
  <env name="document">
   <cmd name="maketitle"/>
   <cmd name="section*">
    <parm>Some biographical information</parm>
   </cmd>
   <xsl:apply-templates select="birthYear"/>
   <xsl:apply-templates select="deathYear"/>
   <cmd name="par"/>
   <xsl:apply-templates select="bio"/>
   <cmd name="section*">
    <parm>Some short quotes from
     <xsl:apply-templates select="lastName"/>
    </parm>
   </cmd>
   <env name="enumerate">
    <cmd name="item"/>
    ``<xsl:apply-templates select="shortQuote1"/>''
    <cmd name="item"/>
    ``<xsl:apply-templates select="shortQuote2"/>''
   </env>
   <cmd name="section*">
    <parm>A longer quote</parm>
   </cmd>
   <env name="quote">
   ``<xsl:apply-templates select="longQuote1"/>''
   </env>
  </env>
 </TeXML>
</xsl:template>

<xsl:template match="birthYear">
 <xsl:apply-templates select="//fullName"/>
 was born in
 <xsl:apply-templates/>.
</xsl:template>

<xsl:template match="deathYear">
 <xsl:apply-templates select="//lastName"/>
 lived until
 <xsl:apply-templates/>.
</xsl:template>

</xsl:stylesheet>
```

Figure 4: XSL markup to transform XML for an artist into TeXML

```
<?xml version='1.0'?>
<!--* DTD for translation to TeX *-->

<!ENTITY % content "#PCDATA|cmd|env|ctrl|spec">

<!ELEMENT TeXML (%content;)*>

<!ELEMENT cmd (opt|parm)*>
<!ATTLIST cmd name CDATA #REQUIRED>

<!ELEMENT opt (#PCDATA|cmd|ctrl|spec)*>

<!ELEMENT parm (#PCDATA|cmd|ctrl|spec)*>

<!ELEMENT env (%content;)*>
<!ATTLIST env name CDATA #REQUIRED>
<!ATTLIST env begin CDATA #IMPLIED>
<!ATTLIST env end CDATA #IMPLIED>

<!ELEMENT ctrl EMPTY>
<!ATTLIST ctrl ch CDATA #REQUIRED>

<!ELEMENT group (%content;)*>

<!ELEMENT spec EMPTY>
<!ATTLIST spec cat (esc|bg|eg|mshift|align|parm|sup|sub|comment|tilde) #REQUIRED>
```

Figure 5: The DTD for TEXML

description	ch attribute	output
escape character	esc	\
begin group	bg	{
end group	eg	}
math shift	mshift	$
alignment tab	align	&
parameter	parm	#
superscript	sup	^
subscript	sub	_
tilde	tilde	~
comment	comment	%

Figure 7: <spec> "ch" attribute values

of the special from Figure 7 in the required "ch" attribute (e.g. <spec ch="bg"/>). The translator will output the character indicated in the table. It is possible to foil this by changing the category of the character output by the translator, so beware.

End-of-line characters (TEX category 5) are treated as space by XML. TEXMLatté substitutes a space (TEX category 10) character for the end-of-line character. There is no way to encode the end-of-line character in TEXML. Paragraph breaks must be coded with `<cmd name="par"/>`.

There is no need to encode the ignored character (TEX category 9). If it is to be ignored, there is no reason to put it in. The same is true of the invalid character (TEX category 15).

Encoding environments. The element <env> is a convenience for expressing LATEX environments. To have TEXMLatté output:

```
\begin{document}
...
\end{document}
```

we write in TEXML:

```
<env name="document">
  ...
</env>
```

The <env> element is not strictly required. It is supplied because it correctly captures in XML the spirit of an environment: it opens a context which is later closed. It is also much more convenient to use than the alternative method, using the <cmd> element:

```
<cmd name="begin">
```

```
    <parm>document</parm>
  </cmd>
  ...
  <cmd name="end">
    <parm>document</parm>
  </cmd>
```

TeXMLatté supplies the default values, “begin” for the begin attribute, and “end” for the end attribute of an <env>. If you have an environment which uses a different convention for the begin and end commands you will specify the “begin” and “end” attributes. For example, the following produces an environment that begins with `\s{t}` and ends with `\e{t}`.

```
  <env name="t" begin="s" end="e">
    ET phone home!
  </env>
```

Any TeXML content may appear within an environment.

Encoding groups. The <group> element is a convenience for encoding groups.

TeXMLatté will supply an open brace at the beginning, and a close brace at the end of the group. The TeX scrap, `{\it italics}` may appear as `<group><cmd name="it"/>italics</group>`.

This is much easier to write than

```
  <spec cat="bg"/>
    <cmd name="it"/>
    italics
  <spec cat="eg"/>
```

There is some technical advantage as well as convenience. The <group> element allows the XML parser to catch any missing close brace by the absence of the close group (</group>). XML cannot detect a missing `<spec cat="eg"/>`.
Any TeXML content may appear within a group.

Conclusion

XML is taking the world by storm. IBM is committed to making XML a viable and widely adopted standard for engaging in electronic commerce and publishing on the internet.

IBM’s TeXML provides an immediate solution for typesetting any XML content. It positions TeX as a potential typesetting back-end for internet publishing and electronic commerce applications.

Figure 8 displays the TeX resulting from processing the XML of Figure 3 with TeXMLatté.

A How to get TeXML

TeXML is available for download from the IBM AlphaWorks website:

`http://www.alphaworks.ibm.com`

Look for “TeXML.” You will find full source, instructions, and examples at the site.

You will need the following in addition to the files provided there:

An XSL implementation. We have used the Lotus implementation, which you may download from the IBM AlphaWorks web site. Look for “LotusXSL.” You will find references to more implementations at the W3C XSL website:

`http://www.w3.org/Style/XSL/`

A java runtime implementation. You will find these at the JavaSoft website:

`http://www.javasoft.com`

Look for “Java runtime” under “products.”

An XML parser. TeXMLatté uses the XML4J parser available from the IBM AlphaWorks web site. Look for “xml4j.”

A LaTeX implementation. You will find references to these at the TeX Users Group website:

`http://www.tug.org/`y

References

ISO/IEC. “Information technology — Processing languages — Document Style Semantics and Specification Language (DSSSL)”. International Standard ISO/IEC 10179:1996(E), International Standards Organization, 1996. See `http://www.oasis-open.org/cover/dsssl.html`. 1015

XSL Working Group W3C. “Extensible Stylesheet Language (XSL) Version 1.0”. W3C Working Draft WD-xsl-19981216, World Wide Web Consortium, 1998. See `http://www.w3.org/TR/1998/WD-xsl-19981216.html`. 1015

IBM/XML. “IBM Answers your XML Questions”. 1999. See `http://www.ibm.com/xml/`.

Sutor, Robert S. and A. L. Díaz. “IBM techexplorer: Scientific Publishing for the Internet”. In *Proceedings of the EuroTeX’98 Conference, St. Malo, France*, volume 28/29, pages 295–308. 1998. Also avail. at: `http://www.gutenberg.eu.org/pub/GUTenberg/publications/publis.html`. 1016

W3C. "Extensible Markup Language (XML) Activity". 1999. See http://www.w3.org/XML/Activity.html.

W3C, MathML Working Group. "Mathematical Markup Language (MathML) 1.0". W3C Recommendation REC-MathML-19980407, World Wide Web Consortium, 1998. See http://www.w3.org/TR/REC-MathML-1980407.html. 1015

```
\documentclass{article}
\title{Edward Weston}
\begin{document}
 \maketitle
 \section*{Some biographical information}
 Edward Weston
 was born in
 1886.
 Weston
 lived until
 1958.
 \par
 Weston was a photographer who pioneered the modern use of photography
 as an art form in the United States.
 \section*{Some short quotes from Weston}
 \begin{enumerate}
  \item
  ``My own eyes are no more than scouts on a preliminary search,
  for the camera's eye may entirely change my idea.''
  \item
  ``Ultimately success of failure in photographing people depends
  on the photographer's ability to understand his fellow man.''
 \end{enumerate}
 \section*{A longer quote}
 \begin{quote}
  ``One does not think during creative work, any more than one thinks
 when driving a car.  But one has a background of years - learning,
 unlearning, success, failure, dreaming, thinking, experience, all
 this - then the moment of creation, the focusing of all into the
 moment.  So I can make 'without thought,' fifteen carefully
 considered negatives, one every fifteen minutes, given material
 with as many possibilities.  But there is all the eyes have seen in
 this life to influence me.''
 \end{quote}
\end{document}
```

Figure 8: The TEX produced by TEXMLatté using the XML of Figure 3