Comparison of XML to HTML for Robert E. Kennedy Library's new books database

March 15, 1999
GRC 461 (Senior Project)--Harvey Levenson
Cal Poly State University, San Luis Obispo

CHAPTER I: INTRODUCTION

This study addresses XML (eXtensible Markup Language) which will likely become the next international standard for the World Wide Web. The World Wide Web Consortium (W3C) standardized XML 1.0 as an official recommendation on February 10, 1998. Its two companions, XSL (eXtensible Stylesheet Language) and XLL (eXtensible Linking Language) became working drafts on August 18, 1998 and March 3, 1998 respectively. So far, the two primary World Wide Web browsers have announced support for XML. The Microsoft Corp. included an XML parser and XSL renderer in Internet Explorer 4.0, and promises more advanced XML tools such as expanded stylesheet support in Internet Explorer 5.0 (due at the end of 1998). Netscape Communications Corp. reports advanced XML tools in its Netscape Communicator 5.0 (due at the end of 1998). Also, Microsoft Office 2000 (due at the beginning of 1999) and Adobe FrameMaker 5.5 will feature user friendly XML utilities.1

Languages written in XML to generate broad standardization have begun emerging rapidly. Information and Content Exchange (ICE) will help syndicate data for multiple uses on the Web, RDF (Resource Definition Format) will help Web sites describe themselves, VML (Vector Markup Language) and PGML (Precision Graphics Markup Language) will address vector graphics, SMIL (Synchronized Multimedia Integration Language) will link text, audio, graphics and animations, VoxML (Voice Markup Language) from Motorola will simplify voice-driven technology, CML (Chemical Markup Language) will classify chemical elements, and MML (Mathematical Markup Language) will automate mathematic typesetting.2

Furthermore, XML has gained popularity in electronic commerce because of its focus on structured information for data transaction. The Open Financial Exchange, supported by Microsoft, banks, financial service companies, and software development firms, has begun developing on-line financial standards based on integrating XML into the client and server. Meanwhile, the Financial Services Technology Consortium began work on an XML electronic commerce messaging format that will standardize check processing via the Internet. Another consortium aimed at retail market, the Open Trading Protocol, proposed an XML-based specification aimed at retail commerce on the Internet in an initiative backed by MasterCard International, AT&T, Hewlett-Packard, IBM, and Wells Fargo.3 And UWI.Com, a Canadian development company, introduced Extensible Forms Description Language (XFDL), an XML-based language for creating online forms for electronic commerce. President and CEO of Intellitech Consulting Enterprises, Dale Dowdie, reports that "the speed with which the industry has embraced and extended the capabilities of XML has stunned even the World Wide Web Consortium XML group."4

Between XML, HTML, and SGML, what will best suit Robert E. Kennedy Library? For now, I predict XML. HTML's rigid formatting options, lack of content description, constant updating, and perpetual differences between browsers make it both inflexible and unstable. SGML's complexity and small vendor markets make it both difficult and expensive to maintain. XML combines the best from both languages by combining the simplicity of HTML with the flexibility SGML.

The purpose of this study is to prepare the Kennedy Library for an XML future by designing an experimental XML database for them. Currently, the library Gopher server houses various unsorted new book listings. I plan to tag these listings in XML and create an XSL stylesheet to automatically organize and display them on the World Wide Web. Since Web browsers under version 5.0 do not process XML, I will create a way to output the listings as HTML until versions 5.0 and above become standard.

CHAPTER II: LITERATURE REVIEW

To design a World Wide Web page, a designer or tool labels text pieces with HTML "tags" that tell a Web browser how to format the elements in the document. For example, to create a page like Figure 1, the HTML coding would need to look something like Figure 2:

The goal of XML resembles SGML's: to label the contents in a document by name rather than appearance (see Figure 3) , so that the document can output as any proprietary medium imaginable, e.g., books, CD-ROM, World Wide Web page. Thus, documents only need to be edited one time--at the source--and then output on-demand through automated conversion. Furthermore, the same XML document can be programmed to revise itself based on who the audience is. For example, a publisher can output an XML document as an American article and as a translated German CD-ROM, complete with altered cultural references and monetary units. Because of XML's flexibility with changing audiences, it especially lends itself to advanced forms of personalization and variable printing.5

XML vs. HTML

XML expands on HTML in many ways. First, XML's tags resemble fields in a database, allowing industries to customize their own database managing systems. One application is efficient online commerce. For example, a paper vendor could use special tags like basic size, color, and price for a database form--then automatically output those descriptions as a paper catalog or Web page. Another application is database publishing. David L. Zwang from *American Printer* explains that society's increasing on-demand mentality has caused product and service providers to demand immediate paper and Web publication. Instead of hiring a large staff to coordinate constant revision of both paper and Web documents, Zwang recommends a better solution by editing one source document and automatically outputting its respective versions through XML. He concludes that printers should begin moving toward XML by logically identifying recurring elements in customer jobs .6

Second, because HTML limits World Wide Web pages to rigid page-oriented instructions, designers can not control fundamental formatting like precise white spacing, kerning, hyphenation, hanging indents, and column snaking.7 Designers that use XML and stylesheet languages like XSL (eXtensible Stylesheet Language) or DSSSL (Document Style Semantics and Specification Language) can standardize exactly how they want the document to look in whatever medium they desire. To do so, the designer would create a different stylesheet for each medium, e.g., a stylesheet for World Wide Web page and a different stylesheet for a printed document.

Third, HTML suffers from instability. Aside from forcing designers to keep up with evolving versions, i.e., HTML, HTML+, HTML 2, HTML 3, and HTML 4, browsers provide proprietary HTML extensions like "blink" and "center" tags that do not work between browsers.8 Since XML is a vendor-independent standard, it is guaranteed to work between browsers that support it. Thom Gillespie of Library Journal put it best with he wrote, "...it [XML] will allow users to design their own browser tags rather than wait for Netscape and Microsoft to disagree on what we need next."9

Fourth, since HTML describes page layout rather than content, users can only search Web pages for key words. This tends to flood users with unrelated results. For example, a search for information on the "Baltimore Orioles" baseball team would probably highlight various pages about the Maryland state bird as well.10 XML uses a concept called "meta data," data within the tags that describes the content. Meta data can describe a Baltimore Orioles baseball team article and an Orioles bird report directly in the document heading. This permits a search for one type of "Oriole" to ignore the other. Meta data also improves database management. In Figure 3, "BIRD5A.EPS" can be categorized as a white dove vector picture rather than as an undescriptive file.

Finally, hyperlinks in basic HTML only offer two options: sending the user to an entirely different document or sending the user to a pre-defined line in any document. XLL (eXtensible Linking Language) would allow designers to greater customize their own hyperlink actions in XML documents. For example, clicking on a hyperlinked word could send the user to anywhere in another document or display a certain portion of text from another document. Or, the designer can specify multiple hyperlink destinations and set criteria for which one to go to, e.g., a hyperlink could send a user to introductory or advanced software documentation depending on that user's level of expertise.11 Additionally, a designer can convey the exact functions of specific hyperlinked text, e.g., a footnote, citation, or glossary entry.12

The World Wide Web Consortium deemed XML as the next generation of HTML (instead of HTML 5.0).13. However, until the browsers support XML, Web publishers will probably resort to outputting XML documents as HTML files, or simply continue working in HTML. After all, small marketing-oriented Web sites have no reason to drop HTML.14 Also, the incredible number of HTML documents and their compatibility with XML will ensure that HTML will stay supported for years to come.15 So while XML will probably not replace HTML, many believe that it will eventually become the format of choice for complex, data-driven Web sites.16

XML vs. SGML

XML sprung from SGML, an international standard (ISO 8879:1986) that used the exact same concepts of intelligent tagging. SGML served certain high-end publishing niches such as aircraft maintenance manuals and federal government reports.17 Various industries including the U.S. Department of Defense, the U.S. Government Printing Office, publishers, and many other businesses adopted SGML successfully. Yet, SGML remains unpopular due to its confusing complexity. It requires the understanding of various "minimization" rules, exceptions, and a mandatory Document Type Definition (DTD) specification that limits choices. SGML consultants Brian Travis and Michael Hahn explain that SGML's small vendor markets and the need for professional consultants make SGML systems both difficult and expensive to maintain .18

Publishers and Web designers created XML by stripping down SGML for the less technically literate. While the SGML specification features nearly 300 pages, XML's specification takes about a tenth of that.19 Among the many omitted laws and redundant options, XML does not need a complicated DTD to enforce rules. David Zwang of *American Printer* explains it as "90 percent of the power of SGML combined with ten percent of its complexity.20

Given XML's momentum, I conclude that it has the potential to dominate over HTML and SGML as long as industries accept it as the ideal strategy for complex data management, publishing, and online commerce. I predict this "revolution" will begin at the end of 1998 when XML document and stylesheet authoring becomes more user friendly.

CHAPTER III: RESEARCH METHODS AND PROCEDURES

I programmed an XML database for the Robert E. Kennedy Library. I will graduate in March 1999 and would like to simplify operations for my successor in Kennedy Library's multimedia department. Currently, the library maintains a listing of new books at gopher://gothic.lib.calpoly.edu:70/11/CalLib_Info/Book_List/. The web page classifies each book in the following subdirectories: Cal Poly major and then month published. Each book listing requires the following recurring information: title, author, subject, publisher, publishing date, major, and call number. As the system stands, I must manually input these fields into an HTML page. I plan to "tag" the

fields with XML to simplify operations for myself and the "successor" who will take over my job when I graduate.

I used descriptive research to describe "what exists." Under the hypothesis that XML is easier to use than HTML, I tagged each book by XML and HTML. XML uses conceptual tags like "title" and "author" while HTML uses formatting tags like "bold" and "paragraph space." Using a stopwatch, I measured the time it takes to construct XML and HTML documents that follow library specifications. My table compares time with number of book listings, i.e., the time for one, ten, twenty, and thirty books per listing. I also needed a row for initial start-up time since the XML document requires an XSL stylesheet. Along the way, I noted any complications I encountered from the XML/XSL system.

I expected that start-up time for an XML document to take longer than start-up for an HTML document, since an HTML document requires little preparation. However, in the long-run I expected XML's automatic formatting to overtake manual HTML design in speed.

My supervisor at the Kennedy Library did not allow me to download Microsoft Internet Explorer 5.0 Beta due to her fears of its instability. Since only Internet Explorer 5.0 and Netscape Communicator 5.0 will accept pure XML, I had to convert all my XML files into HTML files via XSL stylesheets to accommodate the current browsers. Internet Explorer 5.0 and Netscape Communicator 5.0 will come out in the near future.

To convert the XML to HTML via XSL, I downloaded IBM's LotusXSL converter from http://www.alphaWorks.ibm.com/formula/lotusxsl on February 8, 1999 and an XML parser from http://www.alphaWorks.ibm.com/formula/XML on February 9, 1999.The converter and parser only work in Java, so I also downloaded Sun Microsystems' "Java Development Kit" from http://www.javasoft.comm/products/jdk to run it. These all require MS-DOS to work. I installed them into the hard drive of my 233 MHz Pentium PC in the multimedia department. I typed the XML-tagged books through "notepad" in Windows 98.

I used Microsoft FrontPage Editor 98, Kennedy Library's World Wide Web editor of choice, for the HTML side of the experiment. I suspect that my average typing speed (60 words per minute) and mastery of FrontPage Editor slightly skewed the results.

CHAPTER IV: RESULTS

In preparing my experiment to prove the ease of XML, I encountered various complications. My first problem originated on December 16, 1998 when the World Wide Web Consortium (WC3) completely revised its XSL 1.0 working draft from the August 18, 1998 version. XML utilities changed to accommodate the new draft, ultimately resetting my learning curve and making my XML reference books obsolete. This caused me to develop serious reservations about the stability of XML's companion language, XSL. It also cost me a week to learn the new specifications.

Second, upon downloading the new XML utilities, I had difficulties linking the XML converter and XML parser together. Eventually, I sorted out the directories and connected the two utilities by using the MS-DOS command prompt to type "set classpath=c:\lotusxsl\xml4j\xml4j_1_1_14.jar;c:\xml\lotusxsl\lotusxsl\lotusxsl.zip" and then running the program by typing "java com.lotus.xsl.xml4j.ProcessXSL -in test.xml -xsl test.xsl -out test.html"--not the user-friendliness I had hoped for. I lost two days to solve this problem.

Finally, simple typos in the XML and XSL versions disrupted the entire process. I had to consume more time fixing such bugs. Eventually, I refined an XSL stylesheet and XML test file for new books at the library (see Figures 4 and 5).

The creative process took about 27 minutes with an extra 10 minutes for debugging. In addition, the ten-book catalog listing took about an extra half-hour to debug as I had difficulty interpreting the XML parser's cryptic error messages. It turned out that a single ampersand (&) had disrupted the parser and caused me that extra half-hour of grief. Regrettably, while browsers can display HTML pages despite their errors, one error in an XML page will cause little or no output and a cryptic error message.

Table 1 illustrates the results of the timed HTML and XML/XSL comparisons. I dedicated February 19, 1999 and February 25, 1999 to gather data. On February 19, I tagged the December 1998 book listings for finance, journalism, and electrical engineering in both HTML and XML (see Table 1, Appendices A and B). On February 25, I tagged the December 1998 biological science listings in both HTML and XML (see Table 1, Appendices A and B). I chose these particular book listings because of their coincidental book quantities.

These results do not reflect the time I would spend in real-life for proofreading. As stated before, my mastery of FrontPage Editor made the HTML results a little faster than normal. Also, my average typing speed (60 words per minute) made both columns a little faster than the normal typist. Times may have slowed down for the biological science listings for two reasons: fatigue from tediously typing in 30 books without a break, and long biological science jargon that I had trouble typing.

CHAPTER V: CONCLUSIONS

I do not recommend using XML for Kennedy Library's "new book" catalog until Microsoft Internet Explorer 5.0 or Netscape Communicator 5.0 become widely accepted. Converting XML to HTML via XSL only seems to complicate basic HTML inputting, and currently defeats my hypothesis that XML is easier to use than HTML.

First, the XML results demonstrated a negligible time advantage over HTML. Simply put, FrontPage Editor's "cut and paste" feature speeds up efficiency that rivals XML's easy tagging.

Second, I found XML's sensitivity to errors frustrating. HTML is much more tolerant to typographic or logic errors. If an XML page has a

single error, the parser will display cryptic error messages instead of the flawed page. Debugging the ten-book catalog consumed an extra half-hour of my time on a page that should've taken less than ten minutes to construct. Since I and my successor will inevitably err, XML will no doubt compound our stress.

Third, I suspect instability in XML's companion, XSL. Within roughly four months, the specifications for the language had completely changed. It means the library would have to generate a new XSL stylesheet to keep up with the new changes.

In conclusion, I predict HTML will continue to dominate for some time--at least until Microsoft Internet Explorer 5.0 or Netscape Communicator 5.0 become widely accepted. My project will help prepare the Kennedy Library for the upcoming benefits of XML, including searches by title, author, publisher, year, call number instead of just keyword; and increased compatibility with different applications. But for now, XML's meager support and difficult HTML conversion will only complicate workflow at the Kennedy Library.

REFERENCES

Gottesman, Ben Z., PC Magazine, *Why XML Matters*, October 6, 1998, pp. 215-233.
Karpinski, Richard, Internetweek, *XML Tools Take On Multimedia*, October 12, 1998, p. 21.
Bray, Tim, http://developer.netscape.com/viewsource/bray_xml.html, *Beyond HTML: XML and Automated Web Processing*, September 1997, p. 1.
Kutler, Jeffrey, American Banker, *OTP Specification Draft Gaining Momentum Series: 14*, January 15, 1998, p. 18.
Dowdie, Dale, Network World, *XML: The Technology Most Likely to Succeed*, September 14, 1998, p. 46.
Zwang, David L., American Printer, *What in the World Is XML?*, May 1998, pp. 65-66.
Travis, Brian E. and Hahn, Michael, TAG The SGML Newsletter, *HTML, SGML, PDF, XML: What is the difference?*, May 1998, pp. 1-2.
Gillespie, Thom, Library Journal, *XML*, October 1, 1998, p. 129.
Beale, Stephen, Macworld, *XML Ascends on the Web*, February 1998, pp. 28-30.
DuCharme, Bob, The SGML Newsletter, *XML, XLL, and XSL: Current Status, Next*, January 1998, p. 2.
Walsh, Jeff, InfoWorld, *W3C Offers XML Forecast*, July 6, 1998, p. 16.
Bradley, Neil, Document World, *SGML, HTML and XML confused?*, June/July 1998, p. 30.
Boeri, Robert J. and Hensel, Martin, E Media Professional, *XML: The New Document Standard*, June 1998, p. 33.

[BACK TO PROJECTS]