An Overview of the Extensible Markup Language and Related Content-Management Technologies

Greg Meyer
Nichols Research Corporation

The HyperText Markup Language is the most common document format encountered on the World Wide Web but is limited to presentation control. Several emerging technologies such as the Extensible Markup Language are currently being developed that promise dramatically enhanced content management on the Web. This article introduces these technologies and presents issues to consider when implementing them.

The vast majority of Web documents are created and presented in the HyperText Markup Language (HTML). HTML is well suited for hypertext linking and the display of small, relatively simple documents. HTML is an application of the Standard Generalized Markup Language (SGML), defined in ISO 8879:1986. SGML is a metalanguage designed to define document formats. SGML allows documents to describe their own grammar, which is implemented as a set of tags and the structural relationship that these tags represent. Although HTML also provides for a tag set that makes it easy to build Web documents, the HTML tag set is comparatively small and cannot be extended. This lack of flexibility limits HTML's ability to address extensibility, structure, and validation [1]:

- Extensibility: HTML does not allow users to specify their own tags or attributes, which limits their ability to parameterize or semantically qualify their data.
- Structure: HTML does not support the specification of deep structures needed to represent database schemas or object-oriented hierarchies.
- Validation: HTML does not support the kind of language specification that allows consuming applications to check data for structural validity.

At the other end of the functionality spectrum is SGML. SGML allows for the extensibility, structure, and validation that are missing in HTML. With SGML, document formats can be defined, and extremely large document repositories can be managed. However, SGML implementations are expensive, and SGML provides many features that are either unnecessary for Web publishing or require a large effort to implement in a Web environment.

Enter the Extensible Markup Language, (XML). As reported by *Time Magazine* in November 1997, "Doing business on the net is hard because the underlying software is so dumb. XML will fix that." [2] To put it just as eloquently, Bill Gates, chief executive officer of Microsoft, stated that "XML is a breakthrough technology."

Procedural and Generalized Markup

To understand XML and its impact on the Web, a brief introduction to generalized markup is necessary. Markup in electronic documents is the codes embedded in a document text that store the information required for electronic processing. Common examples of document markup include font family and font size.

Markup that represents a procedure for output devices is often referred to as procedural output. For example, when we use a word processor we choose fonts, boldness, and location of text on the page. By marking a word bold, we have defined a procedure that is carried out by an output device: When we view the document on a computer monitor, the word appears bold; when we print the document, the printer prints the word in bold. Although procedural markup can be valuable if all that concerns us is presentation, it has several limitations:

- Procedural markup does nothing to maintain information about the document structure—it is based on the assumption that document structure is directly related to document appearance; only document formatting is recorded and all structure is lost. For example, both quotations and emphasized words may be italicized, even though a quotation has a different function than an emphasis.
- Procedural markup is time-consuming and requires a significant amount of operator training. For example, the documentation for a large software development project may contain thousands of pages, and each of these pages might adhere to a standard formatting convention—the effort to ensure this adherence can be extremely costly.
- Procedural markup is inflexible. When a change to a formatting convention is applied, it requires the manual change of all elements in the document that are affected. In addition, the formatting codes are system dependent: One system may have a particular typeface that another system lacks.

Unlike procedural markup, generalized markup is not concerned with formatting. A Generalized Markup Language (GML) requires two characteristics from the markup:

- Markup should not describe the processing to be performed on the

document; rather, markup should describe the document's structure. This descriptive markup needs to be done only once and will apply to all future processing.
- Markup should be formally defined. With this formal definition of markup, external programs can be used to process the document.

In a GML, a tag is attached to text elements, and formatting rules are associated with these tags. A formatter processes the text and produces a document in a format that is suitable for the output device. The advantages that a GML has over a procedural markup process include the following:
- Generalized markup describes document structure. Meaningful names can be given to tags, such as <PARAGRAPH> to represent a paragraph, and <SURNAME> to represent a person's last name. This application of meaningful names to tags allows the automatic processing of the document, such as the compilation of an index of tagged words.
- Generalized markup allows for much flexibility. To change the appearance of the document, it is necessary only to modify an external procedure that processes the document. This single modification will suffice for all occurrences of the appearance change, and the labor involved with hundreds or thousands of manual changes can be avoided.

XML

XML is a coding system that allows any type of information to be delivered across the Web. Like HTML, the heritage of XML is in SGML. In fact, like HTML, XML is often considered an SGML application (technically, HTML is an SGML application, whereas XML is an SGML profile).

The XML specification was developed by a group of SGML industry leaders and the World Wide Web Consortium (W3C) in 1996, with Jon Bosak of Sun Microsystems as the acting chairman. The goal of the XML Working Group (originally known as the SGML Editorial Review Board) was to develop a markup language that had the functionality of SGML but could be effectively presented on the Web. The initial working draft was completed in late 1996 and became a W3C Proposed Recommendation Dec. 8, 1997 and a W3C Recommendation Feb. 10, 1998. The recommendation (REC-xml-19980210) can be found on the W3C site at http://www.w3.org/TR/1998/REC-xml-19980210.html.

The recommendation outlines the design goals for XML [3]:
- XML shall be straightforwardly usable over the Internet.
- XML shall support a wide variety of applications.
- XML shall be compatible with SGML.
- It shall be easy to write programs that process XML documents.
- The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
- XML documents should be human-legible and reasonably clear.
- The XML design should be prepared quickly.
- The design of XML shall be formal and concise.
- XML documents shall be easy to create.
- Terseness in XML markup is of minimal importance.

The results of the XML Working Group is a GML that allows the creation of new tag sets, instead of being forced to use the minimal tag set available in HTML. More important, XML allows documents to be self-describing and provides for the validation of documents.
- XML documents are self-describing in that they can contain header information known as a Document Type Definition (DTD). The DTD describes the structural rules that the markup in the document is to follow, declares internal and external resources that form part of the document or might be required within the document, and lists non-XML resources that are found in the document for which external helper applications are required. This DTD is instrumental in the successful application of XML processing software.
- XML and a DTD enables a document to be validated by describing a rule set to which that the document must adhere. (It is not necessary for an XML document to contain a DTD—XML documents without a DTD are considered *well formed* but not *valid*. A *well-formed* document adheres to a standard set of rules such as a requirement that each opening tag is accompanied by a closing tag.)

So what is the end result of an XML document? It can be summarized as
- A document that "understands itself" – header information that specifies which elements are allowed and the properties of these elements.
- A document with a browseable and searchable structure – the refusal to allow the exclusion of necessary markup tags allows XML documents to be accessed by XML-aware tools.

The best way to appreciate XML is to look at an example of XML code. In this example, imagine that a company sells automobile parts on line. Marketing descriptions of the products are written in HTML, but names and addresses of customers, prices, and discounts are formatted with XML. Following is the information that describes a customer.

```
<CUSTOMER-DETAILS>
  <NAME>American Wholesale Auto Parts</NAME>
  <ADDRESS>
     <STREET>1234 Maple Drive</STREET>
     <CITY>Grayson</CITY>
     <STATE>Colorado</STATE>
     <ZIP-CODE>80113</ZIP-CODE>
  </ADDRESS>
</CUSTOMER-DETAILS>
```

The XML tags such as <STREET> and </STREET> give meaning to the text "1234 Maple Drive." Its simple syntax is easy to process by machine and has the attraction of remaining understandable to humans.

Related Technologies

As with any other emerging technology, XML brings along with it a host of related technologies. Two of the most impor-

tant of these technologies are the XML linking mechanism and XML style sheets.

XML Linking Language (XLink)
(http://www.w3.org/TR/WD-xml-link.html)

XML linking is defined by the XLink specification as "a simple set of constructs that may be inserted into XML documents to describe links between objects and to support addressing into the internal structures of XML documents. It is a goal to use the power of XML to create a structure that can describe the simple unidirectional hyperlinks of today's HTML as well as more sophisticated multiended, typed, self-describing links." [4] XLink allows specification of which elements in a document are to be interpreted as links and the specific nature of these links. For example, a default link behavior can be defined that requires the user to take a specific action before anything is done with the link. XLink also introduces extended links into Web documents. Extended links can point to any number of targets and can also be bidirectional and multidirectional.

Extensible Style Language (XSL)
(http://www.w3.org/TR/NOTE-XSL.html)

XML style sheets are defined by the XSL specification as "the deliverable for Phase III of the SGML, XML, and Structured Document Interchange Activity of the W3C." [5] The charter for this activity specifies the use of ISO/IEC 10179 Document Style Semantics and Specification Language (DSSSL) for the style-sheet language component. XSL is based on DSSSL and is a style-sheet language designed for the Web community. It provides functionality beyond HTML's Cascading Style Sheets (CSS) such as element reordering. It is expected that CSS will be used to display simply structured XML documents, and XSL will be used where more powerful formatting capabilities are required or for formatting highly structured information such as XML-structured data or XML documents that contain structured data.

Capabilities provided by XSL allow the

- formatting of source elements based on ancestry and descendency, position, and uniqueness.
- creation of formatting constructs, including generated text and graphics.
- definition of reusable formatting macros.
- writing of direction-independent style sheets.
- creation of an extensible set of formatting objects.

A few other XML-related technologies include the following:

Resource Description Framework (RDF)
(http://www.w3.org/Metadata/RDF/)

RDF may prove to be one of XML's most important applications. RDF allows applications to describe new data fields and classes—defining relationships between XML data that might otherwise be left undefined. For example, RDF can be used for bookmarks, user preferences, and a host of other information not directly related to an XML document. This application is a prominent use of metadata (data about other data). RDF will enable enhanced search engines, descriptive relationships between content within a single Web site or between different Web sites, and content ratings for privacy and child protection.

Channel Definition Format (CDF)
(http://pushconcepts.com/microsoft.htm)

CDF provides the ability to author content once for publishing via many different vehicles using push, pull, and static mechanisms. CDF depends on XML for its declarative syntax.

Open Software Description Format (OSD)
(http://www.w3.org/TR/NOTE-OSD.html)

OSD uses unique XML tags to describe software components, including their versions, underlying structure, relationships to other components, and dependencies. It can describe and reference platform native code. Software packages that are described using OSD can be delivered automatically using push technology, allowing for simplified software upgrades and avoiding cross-platform installation complexities.

Open Financial Exchange (OFX)
(http://www.ofx.net)

OFX is a framework for exchanging financial data and instructions among financial institutions and their customers.

XML/Electronic Data Interchange (XML/EDI)
(http://www.geocities.com/WallStreet/Floor/5815)

XML/EDI provides a standard framework to describe different types of data such as shipping invoices and health-care claims. XML/EDI allows information in these various types of data to be searched, decoded, manipulated, and displayed consistently and correctly by implementing EDI dictionaries.

Implementing Content Management Technologies

As an emerging technology, XML has yet to garner widespread industry tool support. This, however, is sure to change in the near future. At a minimum, XML implementation requires an XML or ASCII text editor, an XML parser, and an XML viewer.

Any ASCII text editor can be used to author XML documents. However, there are a few XML-specific authoring tools that make the authoring process significantly easier. XML can be parsed using several tools.

A few of the growing bin of XML-specific software include

- Jumbo (by Peter Murray-Rust) – a set of Java classes designed for viewing XML applications (http://ala.vsms.nottingham.ac.uk/vsms/java/jumbo).
- DataChannel XML Development Kit (by DataChannel) – an enterprise development tool to integrate databases, legacy systems, and business-to-business transactions over the Web using XML (http://www.datachannel.com).
- Lark (by Tim Bray) – an XML processor written in Java (http://www.textuality.com/Lark).
- Copernican XML Developer's Toolkit (by Copernican Solutions) – a toolkit that provides for the checking, valida-

tion, loading, and access of XML documents (http://www.copsol.com/products/xdk/XDK).
- Internet Explorer 4.0 and MXSML (both by Microsoft) – MXSML is an XML parser written in Java, whereas Internet Explorer contains the first public implementation of an XML engine within a Web browser (http://www.microsoft.com/workshop/author/xml/parser).
- TclXML (by Steve Ball) – a Tcl add-on that allows the parsing of XML documents and DTDs (http://tcltk.anu.edu.au/XML).
- XML Styler (by Arbortext) – an XML style sheet editor (http://www.arbortext.com).
- FrameMaker (by Adobe) – a comprehensive document-authoring suite that is XML enabled (http://www.adobe.com).

Closing Thoughts

A final but important question to think about when considering the implementation of XML is the large base of HTML documents that currently exist. Does the advent of XML portend the demise of HTML? Probably not. In most cases, developing XML applications will not be cost effective. HTML is an application that works without modification, there is an incredibly large base of authoring software for creating HTML pages, and a strong industry exists that provides search and retrieval tools for HTML. Information technology is now, and will probably be for many years, concerned primarily with delivery of static information. HTML will probably continue to provide the ideal solution for the bulk of information delivery across the Web for several years.

However, if intelligent information management across the Web is required, XML is a viable, if not dominant, solution. With self-describing and validating mechanisms, browseable and searchable document structures, sophisticated linking, and incredibly flexible presentation support, XML is ideally situated to leverage the information in Web-based documents. ◆

About the Author

Greg Meyer is a principal consultant in the Enterprise Solutions Business Unit of Nichols Research Corporation. He has over 13 years experience in complex systems integration, Web site development, imaging, document management, and collaborative computing. He is a certified document imaging architect and a master of information technology in the Association for Information and Image Management. His professional interests include document management, Web-based collaborative computing, and markup languages.

Nichols Research Corporation
10260 Old Columbia Road
Columbia, MD 21046-1707
Voice: 410-290-9500, ext. 103
Fax: 410-290-7012
E-mail: meyerg@nichols.com
Internet: http://www.nichols.com

References

1. Bosak, Jon, "XML, Java, and the Future of the Web," http://sunsite.unc.edu/pub/sun-info/standards/xml/whyxmlapps.html, March 10, 1997.
2. Krantz, Michael, "Keeping Tabs Online," *Time Magazine*, Nov. 10, 1997.
3. Bray, Tim, Jean Paoli, and C.M. Sperberg-McQueen, "World Wide Web Consortium Recommendation: Extensible Markup Language (XML) 1.0," http://www.w3.org/TR/PR-xml-971208.html, Dec. 8, 1997.
4. Bray, Tim and Steve DeRose. "World Wide Web Consortium Working Draft: Extensible Markup Language (XML): Part 2. Linking," http://www.w3.org/TR/WD-xml-link-970731.html, July 31, 1997.
5. Adler, Sharon, et al., "World Wide Web Consortium Note: A Proposal for XSL," http://www.w3.org/TR/NOTE-XSL-970910.html, Aug. 27, 1997.

Additional Reading

1. Light, Richard and Tim Bray, *Presenting XML*, Sams Publishing, Indianapolis, Ind., 1997.

Web Addition

This article can be found in its entirety on the Software Technology Support Center Web site at http://www.stsc.hill.af.mil/CrossTalk/crostalk.html. Go to the "Web Addition" section of the table of contents.

Organizations, Operations, and Officers

Databases Supporting NATO's Operations in Bosnia

Joseph Arsenault
Canuck Consultants

Long before the first allied solider crossed the Sava River into Bosnia or the first air force crew person landed at Sarajevo Airport or the first maritime sailor debarked at the Split, Croatia seaport, databases were being used to plan, organize, track, and deploy allied forces. This article presents a review of databases that support NATO's Implementation Force and Stabilization Force operations in Bosnia.