The Implications of Well-formedness on Web-based Educational Resources

James L. Mohler
Purdue University
1419 Knoy Hall, Rm 363
West Lafayette, Indiana 47907-1419 USA
Email: jlmohler@tech.purdue.edu

Abstract: Within all institutions, web developers are beginning to utilize technologies that make sites more than static information resources. Databases, XML and XSL are key technologies that promise to extend the web beyond the "information storehouse" paradigm and provide additional functionalities. Thus, the purpose of this contribution is twofold. This paper begins by providing background information and an overview to the issues surrounding the transition from HTML to XML occurring on the web today. It must be noted that the implementation of these technologies is not limited to research and development arms of business and industry. Educators can use them to create knowledge warehouses where students can search, examine and consume educational materials more quickly and easily. The second half of this contribution examines a database-driven, education resource used at Purdue University and the efforts to migrate it toward an XML and XSL-based implementation.

Introduction

A web convergence is beginning to dawn upon the horizon of this new millennium. It is a shift from the "immediacy reaction" that spawned the millions of pages in the later half of the 90's to "intelligent proaction" where information, communication, and applications are singular and are guided by a process-based backbone.

In the middle to late 1990's the major concern in education, business and industry was establishing a web presence, using it as a real-time but static information resource. Many institutions frantically invested, web-enabling massive amounts of their data. From the syllabi and supplementary content of educators to the marketing content and the snapshots of corporate entities, most of the attention was spent on just getting information onto the web, with little concern for how it looked, how it was accessed or how it was found. Today, however, the concern has shifted. The motivation is not just about producing pages and pages of static information floating in ethereal global and local domains. It is about managing the converging *processes*, *information*, *communication* and *applications*, making them readily accessible and intelligently presented when and where they are needed. With the multitude of data that exist on the web, managing, finding and quickly interpreting data while at the same time merging processes, communication and applications is critical.

Regardless of content domain, several features and technologies are integral in any effort to make web-enabled data more efficient, intelligent, and effective. Key to this is the understanding that no single technology fulfills all aspects or needed features. Convergence and integration are crucial to generating intelligent web sites. Features of the web-enabled, intelligent enterprise include:

- Dynamic content that changes to meet the needs of the user in a just-in-time fashion.
- Technology implementations that utilize database systems but also allow real-time information updates and customized queries with little management overhead.
- Global data classification and local labeling systems that provide a means for the user to quickly find and access information in a culturally and environmentally independent way.
- Effectively designed media assets that clearly and logically communicate content while considering bandwidth limitations of end users.
- Use of human-computer interface design principles and theory to provide adequate communication channels that are non-interfering, unimposing and transparent to the user.

- Integration of systems and software that link information, communication, applications with a process-based backbone.
- Software that assists, automates and initiates standard processes and provides ready access to the information on which decisions are made.

Databases, XML and XSL

Current literature reveals the significant research and development effort that has been invested in creating web technologies that make managing, searching and finding data and data nodes easier. Much of the literature focuses on the Extensible Markup Language (XML) (Cadinaels, Duval, Olivie, 1998; Fowler, Fowler, Williams, 1996; Harold, 1999). Although most browsers are only beginning to support XML, the implications of future XML and Extensible Stylesheet Language (XSL) solutions are quite significant. XML is not intended to be coded by-hand, as is the case with HTML. Similarly it is not about the aesthetic appearance or layout and design of pages. Rather, XML is much more complex and diverse. XML is a markup language that can be used to not only describe and classify the contents of a page, but also the tagging scheme itself. In this aspect it is considered a meta-markup language because XML classifies content but also includes a self-contained description of the tags used.

The implied result of XML and XSL is that the complete language and technologies used for web site development could turnover completely within the next five years. Also fundamental to this initiative are the integration of open databases or database-type technology structures that are diverse in their content support, easily extensible and web-enabled in all aspects (Hendrikx, Duval, & Olivie, 1999). The combination of these technologies promises greater flexibility and extensibility for web pages and easier development, maintenance and management for the developer. For the end user, searches are more accurate and labeling systems tend to be more logically named and structured. They also enable advanced technologies such as alternative browsing capabilities, voice-based services, and intelligent search and retrieval agents.

Aside from the aforementioned advantages of XML, it allows several unique benefits for the developer. The most insightful is the ability to create domain-specific markup languages. In fields such as chemistry, mathematics and music studies, representing domain specific content, such as chemical bonds, mathematical equations or musical scores on the web, is very difficult. HTML is far too limiting to represent these domain specific symbols and materials, not to mention the limits of computer-based fonts and their inconsistencies across platforms. XML enables the developer to create his own markup language. Because XML both classifies and describes itself, specialized symbol systems can be used on the web with less concern over end-user technology. Examples of XML implementations already exist for chemistry via the Chemical Markup Language – CML (XML-CML.ORG, 2000), mathematics via the Mathematical Markup Language – MathML (Kamthan, 2000), and music via the Music Markup Language – MusicML. Many other general markup languages are also under development, including those for the delivery of multimedia and vector graphics (World Wide Web Consortium, 2000).

As the convergence of technologies continues, in the public sector the transmission, interpretation and creation of data does not go unaffected by developments on the web. Here, too, XML has application and provides significant potential. One of the difficulties in the financial world is the data tracked on consumers. From personal data to financial records, exchanging data about a consumer from one institution to another is difficult due to the variety of software applications and data formats that exist. Another initiative for XML is to use it as an intermediate data format, that is a neutral file format, for storing consumer data so that information exchange can occur more readily.

Migration to XML

The migration path to the implementation of full-scale XML (and eventually XSL) solutions requires careful observation of established rules that make the migration more easily implemented. Additionally, it requires a different approach to development. Except in the case of database-driven sites, most pages today begin as a formatting exercise, establishing the design and visual aspects of the page and then adding content with little or no attention to what the content is (classification). Future implementations based upon XML and XSL reverse this process. With XML, document content is classified and then rules for formatting, based upon the content classifications, are applied. Most of the developmental sites that implement XML use HTML or CSS for formatting. However, future pages will use XML for classification and XSL for formatting, style and visual attributes.

Additionally, future browsers will be based upon XML standards and will be very restrictive of the content they will render.

The rules for enabling the XML and XSL transition, called rules of *well-formedness*, are style and coding modifications that have little adverse effect on the display of pages in today's browsers. However, the adherence and application of these rules to today's HTML-based pages is vital to the eventual evolution of pages implemented entirely in XML and XSL, as well as being able to correctly view pages in XML browsers. This is also true when content is pulled from databases, which tend house exponential amounts of data.

In general, the rules of well-formed HTML documents, usually identified by the extension .xhtml, are a means of providing "clean code" and "code mechanisms" on the web. It is given that the majority of pages available on the web are written poorly at best and inconsistencies reign supreme, particularly of those pages created by generators or site management applications. The rules for well-formed HTML documents are in most cases a standardization of features that are inconsistent within the HTML Document Type Definition (DTD). Following the rules of well-formedness when creating HTML documents ensures an easier transition to XML as well as compatibility with future browsers. However, since most of today's page editors and site management tools do not understand or utilize XML, implementing rules of well-formedness must, in most cases, be performed by hand. The following partial list presents the main rules of well-formedness as they apply to the creation of .xhtml documents:

> *Close all tags.* Within HTML documents, there are many tags that do not require a closing tag, such as <P>, <LI>, and <DT>, when interpreted by the browser. However, within the HTML DTD, it is specified that these tags have closure. Yet, these tags when left unclosed in a page cause problems for XML interpreters and readers. Consequently, adding the closing tags causes no ill effects when rendered in an HTML browser because they are specified within the HTML DTD.
>
> *Avoid orphaned tags or overlapped tags.* When generating HTML by hand with a text editor, commonly developers will overlap sets of tags. Many page editors and site management tools also allow overlapping tags. For example, a text segment that is to be italic and bold would include both the <B> and <I> tags, one inside the other. When creating composites such as this, it is imperative that the order of the tags be correctly written. If the <B> tag is identified and then <I> tag, the closing order should be </I> and then </B>. If the opening sequence does not match the closing sequence, an orphaned tag exists. Rules of well-formedness require that composite tags be used in logical groupings without orphaning tags.
>
> *All attribute values should be encased in quotes.* As is common practice, some developers enclose all values in quotes and others do not. Consistency is what is usually stressed. Additionally, there are times when double quotations or apostrophes around attribute values are required by certain browsers, particularly in pages that utilize various scripting languages. Well-formed HTML documents surround every HTML attribute value with quotes for proper interpretation by XML browser.
>
> *Use escape sequences for specific symbols.* HTML provides a means for providing special symbols within the text rendered in the browser. These escape sequences permit one to include symbols generally used by the HTML code, such as the less than (<), greater than (>), and ampersand (&) symbols. Often developers mistakenly enter these symbols into their content, without realizing that they may or may not be rendered properly. For example, if the developer wants to have an ampersand (&) in his document, &amp should be entered rather than the symbol itself. For well-formed HTML documents, escape sequence use, as opposed to directly typing in the symbol, is mandatory. Also, the HTML version 4 specification included several additional entity references that can be used in pages. However, these character references are not supported and can cause problems in XML browsers. Developers should only use entity references for ampersand (&), greater than and less than symbols (< >), apostrophes ('), and double quotations (").
>
> *Include "required" structural tags.* Many of the browsers being used with frequency by the general public will ignore terse errors in HTML code, such as forgetting the structural <HTML> and </HTML> tags. For proper interpretation by XML browsers, developers must strictly adhere to the HTML DTD specification and include all required structural tags.

Observe case sensitivity. Unlike HTML, XML is case sensitive. It is best to enter all tags and attributes in lowercase or uppercase, preferably lowercase.

Close empty tags. As opposed to tags that require no closing tags, HTML includes several tags that are called empty tags, such as <BR>, <HR> and <IMG>. A specified in the HTML DTD, these tags have no official closure. Within .xhtml files, these tags must be closed, using <BR />, <HR />, and <IMG /> respectively. Note that for proper interpretation by legacy HTML brows ers and XML browsers there should be a single space between the tag name and the backslash (/).

Implementing the rules of web-formedness within existing large sites is a daunting task. However, where large data resources are concerned, the transition to XML must occur in steps, as it will undoubtedly happen on the web at large as well. The intermediate step is to implement the rules of well-formedness within today' s HTML documents. In the future, the complete overhaul of a site from poorly and loosely written HTML straight to the stringencies involved with XML will be significantly more difficult than the evolution from well-formed HTML to XML. Therefore, it is vital that educational institutions design pages according to the rules of well-formedness if they intend to eventually migrate to XML in the future.

At Purdue University

As the university strives to reach the next level of web integration (fully integrated or true distance education), many are concerned with the evolution of technologies taking place on the web. The most significant concern is the evolution from HTML and existing technologies to future XML and XSL implementations. To test the efficacy of this transformation, one of the major resources being used, the *Computer Graphics Question Repository (CGQR)*, has been targeted for evolution to well-formed HTML, and eventually to full-scale XML and XSL. The CGQR contains over 700 questions that have been answered to date.

The CGQR is a database driven site that provides a means for students to ask questions and find answers to pertinent issues related to computer graphics and the courses they are taking. Several of CGT courses are quite large, some including as many and 150 to 200 people. In the past, one of the difficulties for professors was being able to have one-to-one contact with all of his or her students. The CGQR was created to allow professors to be able to impact all the students in the course while also allowing all students to have direct access to the professor in charge of the course, even if they have another individual within smaller laboratory settings. Even within smaller courses, professors have found that the CGQR strengthens the course and thus the CGQR is used heavily across multiple courses and therefore contains a variety of information.

One of the problems in almost all courses is that students are often disinclined to ask questions. This is sometimes due to fear; often the fear that the question they will ask will be a source of embarrassment when answered. However, if qu estions are not asked, the amount of cognitive processing is questionable. Accordingly, the primary purpose for the CGQR is to increase student inquisitiveness as well as increase the learning that occurs both inside and outside the classroom.

CGT professors can use the CGQR in several ways. Predominantly, content for the CGQR is a direct result of lectures and demonstrations in various courses. At the end of every lecture and demonstration, students are required to submit a note card or small piece of paper for attendance purposes. However, in addition to their name, the submission can also include any questions the student may have in regards to the lecture material, particularly those things that might have been confusing or things for which more information is needed. Questions may also be more general in scope or on topics related to computer graphics but not necessarily related to the course topic at hand. The professor then answers these questions and the questions and answers are placed online anonymously. Students can then browse the questions that have been asked and answered, which reduces the number of redundant questions over the life of a course as well as over subsequent course offerings. Consequently, students may also submit questions directly to the CGQR. In this way, the CGQR' s effectiveness can reach beyond a single course.

The CGQR provides several means for students to access the information. Every question can be tracked by current course, topic, technology, date as well as specific ke ywords. Which elements are tracked and recorded depend on what the professor enters when answering the question. When browsing, the student uses drop down menus that are constructed at run-time based upon the entered fields in the database. Students can examine questions related to their particular course; other courses in the department; specific topics such as multimedia, imaging, et cetera; by technology such as HTML, Photoshop, Flash; by date; or by searching for a specific keyword. Once a student has made a selection, all questions pertaining to a particular topic, technology, et cetera are displayed in chronological order on the right hand side of the screen. Students can see the entire history of questions related to their query.

The CGQR has had many positive effects within the courses in the Department of Computer Graphics. The most significant effect is that use of the CGQR has alleviated student fear of asking questions. Students are much more open to questions concerning specific course materials as well as general topics. Although not quantitatively supported, it is assumed that free inquiry by students has been enhanced the CGQR. Additionally, student evaluations frequently comment that the CGQR is one of the best course resources that they have ever had the opportunity to use in any course at the university. Consequently, professors that choose to use the CGQR find that their student evaluation scores are higher as a result. It is assumed that this is due the ability to have oneto-one contact with students. Although face-to-face communication may not be possible in large courses, students perceive the CGQR as an adequate and effective alternative.

One of the primary concerns of professors new to the CGQR is the number of questions they might receive in a given course. With their time already pressed, many shutter at the thought of having to type in questions and their answers for numerous questions. However, within the courses in which the CGQR is being used, on average there are usually less than five questions per lecture or demonstration in a course of 100 students. Sometimes, there are no questions at all. Although the number of questions may vary, generally the time required to answer the questions is negligible, taking less than 15 or 20 minutes following each class meeting. Yet, as revealed by student evaluations, this time is one that is perceived as most valuable by students. Often even if a student asks no questions via the note cards, the option of being able to ask a question is perceived as significant.

Evolving the Computer Graphics Question Repository

The predominant technologies used within the CGQR include an SQL database, server side VBScripting (via Active Server Pages), client-side JavaScript, and HTML. Because most of the data for the CGQR is stored within the database and pulled via ASP pages, very few files required editing. However, the editing that was required was made more difficult due to the meshing of scripting and HTML. Although there are over 700 questions currentlyin the CGQR, the totality of the major search pages and form pages for the site number 15. As noted earlier, editors and site management tools do not generally support rules of well-formedness. Therefore editing the CGQR pages was performed by hand using a text editor.

Although the number of pages that required editing was few, the biggest concern for evolving the CGQR was the data contained within the database. Over the years, a variety of embedded HTML, that is HTML directly entered into a question or answer, had been placed into the database. As noted in the rules of well-formedness, specific character entities can be problematic. Similarly, open tags were also an issue because XML readers require closure on every tag for proper interpretation. To help resolve this problem, specific queries where developed within the database to help find potential problematic fields. Once found, the field data was modified to follow the rules for well-formedness.

Although minor issues concerning the evolution of the CGQR are still being dealt with, the predominance of the site conforms to the rules of well-formedness. Because of the limited number of pages, the CGQR offered an excellent and somewhat limited opportunity to determine the difficulty of evolving from HTML to xHTML. The time required was rather minimal, less than 40 working hours. However, it is obvious that larger sites with various complexities not dealt with in this example may pose more of a problem. The important point is that the evolution of resources such as the CGQR must occur in steps, if the evolution is to occur at all. Leaping from a very simple and base technology to one with the complexities defined in XML is a daunting task. Successful evolution requires small steps rather than major leaps. Additionally, the distribution of changes over time makes evolving from one technology to another much more cost effective and much more easily implemented.

Due to the success of the CGQR as a student and faculty resource, future endeavors will focus on continued technological evolution to enhance its features and functionality. Currently further tests and modification are being performed with XML browsers and XSL formatting to foster increased functionality.

Conclusion

Regardless of the capacity in which an institution is using web-based materials, it is imperative that they be created in a way that permits them to evolve as easily as possible to future technologies. Developers of content and the institutions that financially support the creation of these materials cannot do so in a vacuum. Wherever and whenever possible, materials should be designed with open-technologies and with foresight of coming technologies. Change is an inevitable part of designing for the web. Thus, educators and developers currently working on web materials should design those materials with the rules of well-formedness, and the eventual and inevitable evolution to XML and XSL, in mind.

References

Cadinaels, K., Duval, E. & Olivie, H. (1998). High-level database document specifications using XML. *The Proceedings of WebNet' 98*. Association for the Advancement of Computing in Education, Charlottesville, VA [Producer and Distributor].

Department of Computer Graphics. (2000). *Computer Graphics Question Repository* [On-line]. Available: http://www.tech.purdue.edu/cg/facstaff/jlmohler/q&a2/.

Fowler, R. H., Fowler, W. A. & Williams, J. L. (1996). 3D visualization of WWW semantic content for browsing and query formulation. *The Proceedings of WebNet' 96*. Association for the Advancement of Computing in Education, Charlottesville, VA [Producer and Distributor].

Harold, E. R. (1999). *XML bible*. Chicago: IDG Books Worldwide.

Hendrikx, K., Duval, E. & Olivie, H. (1999). Linking XML and databases. *The Proceedings of WebNet' 99*. Association for the Advancement of Computing in Education, Charlottesville, VA [Producer and Distributor].

Kamthan, P. (2000). *Mathematical Markup Language* [On-line]. Available: http://indy.cs.concordia.ca/mathml/.

World Wide Web Consortium. (2000). *Extensible Markup Language (XML)* [On-line]. Available: http://www.w3.org/XML/.

World Wide Web Consortium. (2000). *Synchronized Multimedia* [On-line]. Available: http://www.w3.org/AudioVideo/.

World Wide Web Consortium. (2000). *W3C Scalable Vector Graphics (SVG)* [On-line]. Available: http://www.w3.org/Graphics/SVG/.

XML-CML.ORG. (2000). *XML-CML.ORG - The Site for Chemical Markup Language* [On-line]. Available: http://www.xml-cml.org/.