SGML on the Web: A Tale of Two Sites

Joshua Lubell

National Institute of Standards and Technology
Manufacturing Systems Integration Division
Building 220, Room A127
Gaithersburg, MD 20899
USA
joshua.lubell@nist.gov
http://www.nist.gov/msid/

Reprinted from the proceedings of *SGML/XML '97*, Washington DC, December 8-11, 1997

Abstract

Businesses and organizations are increasingly finding that HTML (Hyper-Text Markup Language) offers no help whatsoever in managing the information on their web sites. SGML (Standard Generalized Markup Language) provides the flexibility and reuse lacking in HTML. However, SGML alone does not address the problems involved in maintaining on-line document repositories. Although traditional database management systems are clumsy at managing hyperlinked documents, a system combining SGML, database technology, and the protocols of the Web can provide a reasonably robust environment for developing and maintaining a web site. Two possible site designs employing SGML are discussed and evaluated with respect to a set of design objectives and choices.

List of Acronyms Used

CGI
    Common Gateway Interface
DTD
    Document Type Definition
HTML
    Hyper-Text Markup Language
HTTP
    Hypertext Transfer Protocol
HyTime
    Hyper-media Time-based Structuring Language
MCF
    Meta Content Framework
NIST
    National Institute of Standards and Technology
SGML
    Standard Generalized Markup Language
SQL
    Structured Query Language
STEP

Standard for the Exchange of Product Model Data

URL
: Universal Resource Locator

XML
: Extensible Markup Language

---

1. Introduction

Businesses and organizations are increasingly turning to the World Wide Web as a means of sharing information. They are finding that, while the Web provides a convenient platform-independent and geography-independent way for individuals to communicate with one another and to interact with software applications, the language of the Web - HTML (Hyper-Text Markup Language) [Rag97] - offers no help whatsoever in managing information. Although HTML is a good electronic delivery medium for many texts, it is often a poor choice for representing source documents. HTML provides no standard means for reusing the same text in multiple documents. Also, HTML limits content providers to a fixed set of tags.

SGML (Standard Generalized Markup Language) [Cov97][Gol92] provides the flexibility and reuse lacking in HTML. By using an application-specific DTD (Document Type Definition) to represent a web site's contents, site developers can take advantage of the structure defined in the DTD when building data access interfaces and document management tools. Even if the site's contents are represented using one of the standard HTML DTDs, developers can still take advantage of SGML features such as entities and marked sections to promote data reuse. However, SGML alone does not address the problems involved in maintaining on-line document repositories. Although traditional database management systems are clumsy at managing hyperlinked documents, a system combining SGML, database technology, and the protocols of the Web can provide a reasonably robust environment for developing web sites. In this paper, I shall discuss and evaluate two possible site designs employing SGML, with respect to a set of design objectives and choices.

2. Web Site Design Objectives

This discussion addresses several web site design objectives. Each objective's relative importance depends on a site's application-specific characteristics: the kinds of documents being accessed, the volume of information being managed, and who is providing the content. These objectives are not meant to be an exhaustive list. In particular, they do not include goals already being addressed by the XML (Extensible Markup Language) [XML1] standard, such as improving upon the formatting and hyperlinking capabilities of HTML.

2.1. *Creation*: Ease of Data Creation

It should be easy for content providers to create and modify documents. HTML does a fairly good job meeting this goal. HTML has a simple tag set and, with the help of a high quality HTML editing software package, it is not hard for good writers to learn enough HTML to create nice looking, easy to read documents. Editing SGML documents, on the other hand, can be cumbersome when the DTD is complex.

2.2. *Accessibility*: Accessibility to Web Search Engines

Numerous search engines with publicly accessible HTML user interfaces are available for locating resources on the Web. These search engines rely on ‘web spider’ programs [Che95] that traverse the Web to build their search indexes. Therefore, in order for a web site’s documents to be searchable using these web search engines, the documents need to be in the form of static HTML (or plain text) files. Document authors have the option of providing cues to a search engine through HTML’s ‘META’ tag, which can be used to specify name/value pairs describing arbitrary characteristics of the document. However, different search engines may use different ‘META’ syntax, since HTML does not dictate any particular set of properties.

2.3. *Sensitivity*: Context-sensitive Searching

If context-sensitive searching is required, then the data should be structured in an application-specific manner, i.e. according to an SGML DTD. HTML alone provides only very limited structuring (titles, headings, paragraphs, etc.) and fails to represent application-specific relationships. Therefore, this objective conflicts with the *accessibility* objective. The proposed MCF (Meta Content Framework) [Guh97] uses XML syntax to specify information about the content of a web document. If MCF is implemented in web browsers, then web site designers will no longer be forced to choose between the *accessibility* objective and the *sensitivity* objective.

2.4. *Maintainability*: Ease of Maintenance

Once the total size of all static HTML documents in a web site grows beyond a few hundred kilobytes, it becomes necessary to spend a lot of effort keeping the site’s contents from becoming stale. This requires checking for dead hyperlinks, making sure information is up to date, and maintaining consistency between the documents. HTML provides no support for any of these tasks. In particular, HTML lacks a mechanism for reuse, requiring document authors to specify the same content over and over again, inviting inconsistencies.

2.5. *Luminosity*: Illumination of Off-Site Resources

One of the reasons for the Web’s growing popularity is the ability of a web user to quickly surf from one web site to another with a simple click of the mouse on a hot spot. The degree to which a web site provides hyperlinks to documents on other web sites is referred to as the site’s *luminosity* [Bra96]. Although luminosity is generally a good thing, off-site hyperlinks should be used with discretion since too many of them can be distracting to the web surfer.

3. Web Site Design Choices

A web site designer using SGML has several decisions to make. Which choices are best depends on the relative importance of five objectives given the application. In particular, the designer must address the following issues:

3.1. SGML-Conforming HTML Versus Application-Specific DTD

The designer must decide whether to represent the source documents as HTML conforming to an HTML DTD or to use a DTD specifically tailored to the Internet's requirements. Each approach has its advantages and disadvantages. It is usually easier for content providers to author documents in HTML than to use an application-specific DTD. On the other hand, documents conforming to an application-specific DTD can be searched using an SGML search engine according to the structures in that DTD. Thus, using SGML-conforming HTML to represent source documents favors the *creation* objective, and using an application-specific DTD favors the *sensitivity* objective.

3.2. Dynamic HTML Versus Static HTML

Until most web browsers support the display of SGML documents, the majority of SGML-based web sites will need to display their SGML source documents as 'browser-ready' HTML. Even if the source document is SGML-conforming HTML, all entity references and marked sections must be normalized. Therefore, an SGML-based site design needs to include an SGML-to-HTML translator to render the SGML source as browser-ready HTML. While the implementation of a translator is straightforward, the designer needs to decide whether the translation should be done on demand whenever a web browser requests access to a document or whether the translation should be done a single time after a document is created or modified. Figures 1 and 2 illustrate high-level SGML-based site designs using dynamic HTML and static HTML respectively.

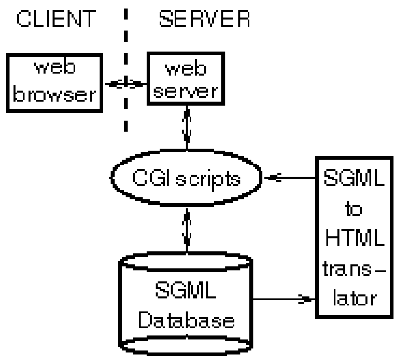

Figure 1: Web Site Design Employing SGML With Dynamically Generated HTML

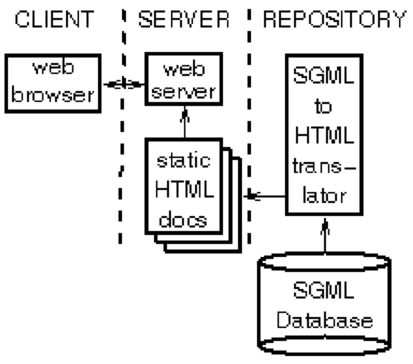

Figure 2: Web Site Employing SGML With Static HTML Documents

Each alternative has its advantages and disadvantages. Generating the browser-ready HTML dynamically simplifies the web site’s design because it eliminates the need to keep statically-generated HTML documents in sync with the SGML source. On the other hand, dynamically-generated documents are not accessible to external search engines. Also, dynamic generation of HTML requires the use of CGI (Common Gateway Interface)[Rag97][NCSA], a standard for interfacing applications with web servers. CGI adds overhead to the server, in addition to the processing required to generate an HTML document. Thus, frequent requests to generate HTML from SGML on demand can degrade web server performance. To summarize, dynamic HTML works best when the *maintainability* objective is more important than the *accessibility* objective and when the generated HTML is in response to a request not likely to be repeated frequently (such as a database query).

Serving static HTML documents enables the use of external search engines and eliminates the need for the web server to translate SGML to HTML and to incur CGI overhead every time a document is requested. Also, because translation from SGML to HTML is done independently of HTTP (Hypertext Transfer Protocol) [Fie97] requests from web browsers, the SGML document repository can exist on a server other than the web server. However, keeping the SGML source and the HTML consistent requires an infrastructure for configuration management. The larger the number of documents in the web site, the more sophisticated the data management needs to be. A very large site requires a high powered relational or object database management system to keep track of the associations between the SGML source and the browser-ready HTML. Thus, serving static HTML works best in situations where either of the following conditions hold:

- The *accessibility* objective is important, and the frequency of requests relative to the number of possible requests is large.
- Support for CGI scripting is not available from the Internet service provider.

3.3. References to Documents Outside the Web Site

The more hyperlinks to off-site documents a web site has, the more luminous it is. On the other hand, pointers to external resources need to be checked regularly for staleness since off-site documents can disappear or undergo modification at any given time. Thus, a site design promoting references to external documents favors the *luminosity* objective over the *maintainability* objective while restricting external references favors the *maintainability* objective over the *luminosity* objective.

4. Two Web Site Case Studies

I am currently implementing two web site designs. The sites being developed use very different approaches with respect to the issues previously discussed and emphasize different design objectives. The first site [Lub97][Lub96], being used as part of an environment for developing and deploying standards, uses application-specific DTDs [APDE], an SGML search engine installed on the web server, and makes extensive use of CGI. The second site, containing a mix of project information, home pages, and documentation, relies on SGML-conforming HTML to represent the data and serves static browser-ready HTML in response to requests for URL (Universal Resource Locator)s. Although these two web sites are interesting to compare with one another, they by no means represent the only design choices available. For example, a web site designer can combine an application-specific DTD with generation of static browser-ready HTML documents as Norman Walsh has done [Wal97]. Walsh represents the source data for his entire web site as a single SGML document instance conforming to the DocBook DTD [All97] and has built an SGML-to-HTML translator that creates multiple hyperlinked

HTML documents from the source. Walsh’s approach enables him to rely on an SGML-validating parser to ensure that all cross references internal to the web site are consistent although another means must be used to check for stale links to external documents.

4.1. Site 1: Dynamic HTML Generated From SGML Database Queries

This web site design is centered around a repository of SGML documents that are indexed for fast structured-based retrieval using an SGML search engine. The DTDs used have been created specifically to represent documents comprising STEP (Standard for the Exchange of Product Model Data) (ISO 10303)[STEP], a family of standards that attempt to define an ontology for the exchange of product data throughout a product’s life cycle. Therefore, these DTDs support queries that are highly application-specific. A set of CGI scripts generate HTML in response to queries composed from information entered onto HTML forms. The CGI scripts use a library of access functions which provides an interface to the repository’s search engine. If the result of a query is a block of tagged text, then a translation module converts the raw SGML data to HTML. Because all HTML is dynamic, this web site design closely resembles Figure 1.

Site 1 succeeds well at the *sensitivity* objective because it permits queries with search criteria tailored to the semantics of the documents in the repository. It does so, however, at the expense of the *creation* objective in that the complexity of the DTDs makes it difficult to author new standards in SGML as well as convert existing standards created in word processor formats. Although Site 1 does not meet the *accessibility* objective, the *accessibility* objective’s importance is diminished because the ability to perform context-sensitive searches lessens the need to use the Web’s general purpose search engines. Site 1 meets the *maintainability* objective to the extent that a validating SGML parser can detect inconsistent ID references and entity references. Because Site 1’s DTDs do not support HyTime (Hyper-media Time-based Structuring Language) [HyT97], XML linking [XML2], or some other means for representing hyperlinks from one document to another, the *luminosity* objective is not met. However, this is more a limitation of the STEP DTDs than of the framework of the design itself. If support for hyperlinks between documents and off-site were added to the DTDs, then Site 1’s library of access functions and CGI scripts could be augmented to achieve the *luminosity* objective.

Site 1 is best suited for applications where data is highly structured, modification of documents is tightly controlled, and (unless the DTDs used support hyperlinks to other documents) data is fairly self-contained. Site 1 is also a good choice when the volume of data is large, and it would be impractical to maintain it as static HTML. The area of application in which Site 1 is being deployed meets the above criteria. STEP standards are highly structured, and a lengthy approval process is required to change their contents. They also tend to be large - some of them are over a thousand pages long when printed on A4 paper.

4.2. Site 2: Static HTML Generated From SGML-Conforming HTML

This web site design, which resembles Figure 2, specifies a collection of SGML documents conforming to the HTML 3.2 DTD. Each SGML-conforming HTML document is stored in a file. These documents may contain SGML constructs not understood by web browsers such as references to general entities defined specifically for the web site and marked sections. In particular, all HTML links (i.e. anchors containing ‘HREF’ attributes) pointing to external documents are defined as general entities. Whenever a document is created or its source modified, browser-ready HTML is created by running a script that uses the *spam*[1] SGML normalizer, resolving entity references and instantiating any marked sections[Cla97][Kim96].

[1] Third party software tools are identified in this paper to foster understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products identified are necessarily the best available for the purpose.

In order to prevent the hyperlinks from getting stale and to manage the generation of web-browser-ready HTML, a database is needed to keep track of entity use. The database's data model, shown in Figure 3, contains objects representing authors, web documents, and entities. Each author object contains identification information for the author as well as pointers to all the documents for which the author is responsible. Document objects contain a pointer to the author responsible for maintaining the document, the location of the SGML source file, the URL for the corresponding browser-ready HTML file, and a list of all entity references specified in the document. Each entity object contains the entity's name, the entity's value, and pointers to all the documents referencing it.

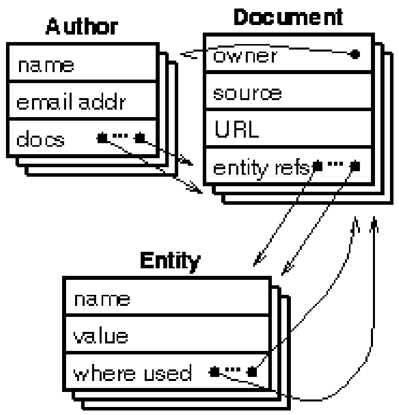

Figure 3: Document Management Data Model for Site 2

Whenever a content provider creates a new document or modifies an existing one, a 'check-in' procedure performs the following steps:

1. If the document is a new document, create a document object whose owner is the author.
2. Update the document's list of entity references.
3. For each entity object affected, update its list of document pointers.

Whenever an entity is added or modified, the following steps are performed:

1. If a new entity has been added, create an entity object with a null list of document pointers.
2. If the entity's value has been changed, run the SGML-to-HTML translator on the SGML source file of each document using the entity, writing the result to the location of the browser-ready HTML file. Notify the document's owner by email that the entity has been modified.
3. If the entity's name has been changed, notify the owners of all documents affected.
4. Generate an entity catalog file conforming to the SGML Open standard [Gro97] for entity management. This is needed for *spam* as well as for any SGML editing applications used by content providers to create the source documents.

A link-checking program is needed to ensure that there are no dead or stale hyperlinks in the entities in the database. Any problems uncovered by the link-checking program should be brought to the attention of the web master by email. The server should be configured to run this program automatically on a regular schedule, perhaps daily, so that the web master does not have to remember to do it.

Site 2 meets all design objectives except the *sensitivity* objective. Document creation and modification is simple provided that the authors have access to software that creates SGML-conforming HTML and supports referencing of non-HTML-defined entities. Because the web-browser-ready HTML is generated only once after document modification rather than dynamically in response to queries, the web-browser-ready HTML documents can be accessed using the Web's general purpose search engines. Context-sensitive searching is not supported because HTML's tag set is not application-specific. Ease of maintenance is achieved, although it requires a complex and possibly expensive infrastructure. The only limitations on luminosity are those self-imposed by the content providers.

Site 2 is best suited for loosely structured web sites with multiple content providers where the content providers are free to provide hyperlinks to wherever they please. Many of today's web sites fit this description. Site 2 also makes sense for web masters whose Internet service provider does not provide them with support for CGI scripting as part of their site hosting package. Depending on the size of the web site, implementing Site 2 may be possible using a low-powered database implementation, or it may require a full featured SQL (Structured Query Language) or object-oriented database engine [Ull88].

5. Status and Future Directions

Sites 1 and 2 are both under development, and the portions of them that have already been implemented are experiencing use. Site 1's underlying framework - the data access interface to its SGML search engine - is fully implemented. CGI applications have been developed for accessing a subset of the STEP standards, and the queries supported represent a subset of those possible given the application-specificity of the STEP DTDs. Since these applications [UoF97] permit developers and implementors of STEP to perform structured searches and to view data associations not explicitly stated in the standards documents themselves, they are potentially far more useful than paper versions or on-line documents in word processor formats. Current efforts are focusing on converting more existing STEP standards into SGML so that they can be included in the repository and implementing additional kinds of queries. Future efforts will involve developing a document architecture for STEP and related standards using architectural forms and creating additional DTDs using this architecture.

Development of Site 2 is in its early stages. The translator for converting SGML-conforming HTML documents to browser-ready HTML and a catalog of entities currently exist. I maintain a variety of web pages with source written in SGML and using the entity catalog. Since the document management database discussed in the previous section has not yet been implemented, I check my documents manually using a third party HTML link validator. Once implementation of the document management database is complete, this web site will be expanded to include more documents and to support content providers other than myself.

Sites 1 and 2 illustrate a dilemma that web site developers wishing to take advantage of the benefits of SGML currently face. On the one hand, they can rely heavily on SGML's ability to represent data in an application-specific, structured manner and on CGI to dynamically generate browser-ready web output in response to SGML database queries. While such a site design enables users to quickly find information through application-specific queries and is easier to maintain than a collection of HTML documents, it requires extra effort on the part of content providers, additional server overhead, and the implementation of HyTime linking or some comparable mechanism if hyperlinks to off-site web pages are desired. On the other hand, web site developers may choose to minimize the burden on content providers and to maximize server performance, interoperability with web search engines, and linkage with other web sites. In this case, they must sacrifice application-specific structured query capability and implement tools for managing entities and maintaining hyperlinks.

The emerging XML standard promises to provide web site developers with the best of both worlds, allowing them to enjoy most of the benefits of SGML while not sacrificing the convenience of HTML and interoperability with the rest of the Web. If XML is ultimately successful, not only will it be easier for web site developers to use SGML, but also they will be able to build applets embedding capabilities supporting the manipulation of SGML data in web clients [Bos97]. This will reduce the burden on the server and, more importantly, will open a new world of possibilities for interaction between SGML repositories and other databases and applications.

More information about the work discussed in this paper is available at http://www.nist.gov/apde/ on the Web.

Acknowledgements

This work was funded by NIST (National Institute of Standards and Technology)’s Systems Integration for Manufacturing Applications program, a part of the U.S. Government’s High Performance Computing and Communications initiative.

I wish to thank this document’s NIST reviewers - Don Libes, Lisa Phillips, Jim Fowler, and Tom Kurihara - for their corrections and helpful comments.

Bibliography

All97
: Terry Allen, Eve Maler, and Norman Walsh, *Reference for the DocBook DTD*, HaL Computer Systems, Inc., O’Reilly & Associates, Inc., Fujitsu Software Corporation, and ArborText, Inc., Release 3.0, January 1997, URL: http://www.ora.com/davenport/.

APDE
: Application Protocol Development Environment home page, National Institute of Standards and Technology, URL: http://www.nist.gov/apde/.

Bos97
: Jon Bosak, *XML, Java, and the future of the Web*, World Wide Web Consortium, March 10, 1997, URL: http://www.w3.org/XML/.

Bra96
: Tim Bray, *Measuring the Web*, Proceedings of Fifth International World Wide Web Conference, Paris, France, May 6-10, 1996, URL: http://www5conf.inria.fr/.

Che95
: Fah-Chun Cheong, *Internet Agents: Spiders, Wanderers, Brokers and Bots*, New Riders, ISBN: 1562054635, October 1995.

Cla97

James Clark, SP SGML Parser, version 1.2, URL: http://www.jclark.com/sp/.

Cov97
: Robin Cover, *The SGML/XML Web Page*, URL: http://gopher.sil.org/sgml/.

Fie97
: R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee, *Hypertext Transfer Protocol - HTTP/1.1*, World Wide Web Consortium, RFC 2068, January 1997, URL: http://www.w3.org/Protocols/.

Gol92
: Charles Goldfarb, *The SGML Handbook*, Oxford University Press, 1992.

Gro97
: Paul Grosso, *Entity Management*, SGML Open Technical Resolution 9401:1997, URL: http://www.sgmlopen.org/.

Guh97
: R.V. Guha and Tim Bray, *Meta Content Framework Using XML*, submitted to the World Wide Web Consortium, June 6, 1997, URL: http://www.textuality.com/mcf/.

HyT97
: International Organization for Standardization and International Electrotechnical Commission, Information processing-Hypermedia/Time-based Structuring Language (HyTime) - 2d edition, ISO/IEC 10744:1997.

Kim96
: W. Eliot Kimber et al., *SPAM.CGI: Treating HTML as SGML in a Simple Yet Cool Way*, articles posted to comp.text.sgml newsgroup, April 25, 1996 through May 2, 1996.

Lub96
: Joshua Lubell and Lisa Phillips, *SGML Application Development: Tradeoffs and Choices*, Proceedings of SGML'96 Conference, Boston, Massachusetts, November 1996.

Lub97
: Joshua Lubell, *The Application Protocol Information Base World Wide Web Gateway*, Proceedings of ASME Design Engineering Technical Conferences: Computers in Engineering Conference, Sacramento, California, September 1997.

NCSA
: National Center for Supercomputing Applications, *The Common Gateway Interface*, URL: http://hoohoo.ncsa.uiuc.edu/cgi/.

Rag97
: Dave Raggett, *HTML 3.2 Reference Specification*, World Wide Web Consortium, January 17, 1997, URL: http://www.w3.org/TR/REC-html32.html.

STEP
: International Organization for Standardization, Industrial automation systems and integration-Product data representation and exchange-Part 1: Overview and fundamental principles, ISO 10303-1, 1994.

Ull88
: Jeffrey Ullman, *Principles of Database and Knowledge-Base Systems*, Computer Science Press, Rockville, Maryland, 1988.

UoF97
: Joshua Lubell, Units of Functionality Repository, URL: http://www.nist.gov/apde/uofs/.

Wal97
: Norman Walsh, home page, URL: http://www.berkshire.net/~norm/.

XML1
: Tim Bray, Jean Paoli, and C.M. Sperberg-McQueen, *Extensible Markup Language (XML)*, World

Wide Web Consortium, W3C Working Draft 07-Aug-97, URL: http://www.w3.org/xml.

XML2
Tim Bray and Steve DeRose, *Extensible Markup Language (XML): Part 2. Linking*, World Wide Web Consortium, W3C Working Draft July-31-97, URL: http://www.w3.org/xml.