Beyond HTML: Web-Based Information Systems

K. V. Chandrinos, P.E. Trahanias

Institute of Computer Science

Foundation For Research and Technology - Hellas (FORTH)

{kostel|trahania}@ics.forth.gr

In this paper we briefly review current status of Web-Based Information Systems (WBIS) and present a number of different WBIS technologies and how they are being integrated during the ARHON project (Archiving, Annotation and Retrieval of Historical Documents). The project amounts to transforming a vast collection of OCR-resisting historical manuscripts belonging to the Vikelaia Municipal Library at Heraklion, Crete, to a fully functional Digital Library supporting the needs of casual users as well as scholar researchers [1]. The solution we opted for, was the creation of an image database and the adoption of standard Web browsers enhanced with Java applets for browsing, querying and updating the database. This way we managed to provide an Intranet based querying and retrieval mechanism through a Web server. The access mechanism was designed to work with standard browsers so that at a later stage Internet access to the digital library should be trivial to implement. Initial decisions and results were presented in [2].

Introduction

The widespread use of the Web for information management, as opposed to mere presentation, has revealed the weak points of HTML as a general purpose scripting language for information retrieval and query result presentation over the HTTP protocol. Early implementations of Web front-ends to libraries, that proposed an HTML form page in place of typing into a telnet connection, have rather disappointed end-users who had to face extended network latency for no apparent profit. We share the view that digital libraries should offer the end-user enhanced functionality compared to the physical library [12, 14, 19]. Despite recent attempts to extend the language specification (HTML 4.0) and reverse its static nature utilizing server-side processing (e.g. Meta-html [8], dynamic HTML, etc.) it is still hard to achieve the dynamic interactive environments we would like to have at our disposal when accessing large online repositories. Based on the experience gained during our project design, we intend to present the main problems of standard HTML for Web-Based Information Systems (WBIS) and possible leeways for enhancement offered by recent developments. After reviewing current capabilities, we explain and justify our rationale for choosing a combination of standard HTML with Java applets and servlets to implement a 3-tier architecture for the manipulation of the digital library.

Web-Based Information Systems

Although most HTML programmers feel they could express their data of interest if they had just one more tag, one that would suit their particular project, a number of writers have identified the key problems of HTML that prohibit an effective transition of Information Systems to the Web [5, 7]. The chief shortcomings of HTML, which cannot be addressed by proprietary tags introduced from rival browser companies, boil down to lack of:

- *Extensibility*, since HTML does not allow users to define their own tags.
- *Structure*, since HTML does not allow deep structures needed to represent object-oriented hierarchies or database schemas.
- *Validation*, since HTML does not allow browsers to check data for internal validity on downloading.

These problems, combined with the nature of HTTP as a transfer protocol, have led to a number of difficulties affecting attempts to bring content to the Web. For example, they specify that only discrete transactions between client and server are allowed. Even worse, these discrete transactions can only be anonymous and stateless. As a result, current standard Web technologies can support distribution of only static multimedia documents. The introduction of plug-ins to remedy this, apart from being awkward, violates the openness and vendor-independent nature of the Web. Also, a discrete, stateless transaction model does not support either familiar interface elements (e.g. context sensitive help) or event-based interaction (e.g. error-checking on input, direct manipulation).

To avoid the shortcomings of HTML and HTTP most of the users first turned to server-side processing utilizing the Common Gateway Interface (CGI) scripting facility. Form processing answered, although clumsily, the problem of the client initiating a server response. Of course, the form validity could only be checked when it reached the server, a thing that added further delays to the already slow model of process-spawn-per-request, which CGI scripting imposed. An answer to data checking on input came from embedded JavaScript code or Java applets. As opposed to JavaScript, a scripting facility for web client applications developed by Netscape, the Java language was developed by Sun and brought in the promise of platform independence. This has yet to be fully realized at the time of writing, with particular vendors resisting full compliance to the Java specification issued by Sun, but momentum seems to gather in that direction. Java endows the Web programmer with capabilities to express the most complex interface elements in the form of applets. However, the true power of network programming and network-oriented languages like Java, stems from server-side processing. As one should expect, not all the material that deserves exposition to the Web is already formatted in an appropriate way. Server-side programming allows content publishers to afford a considerable choice of middleware between their legacy-formatted data and the Web. The most promising version of middleware seems to be the Common Object Request Broker Architecture (CORBA) which stems from a large industry powered consortium, the Object Management Group (OMG). This architecture allows existing or future components to be treated as objects, where the interfacing between these components is provided by static or dynamic methods. This way, communication with the components becomes independent of implementation (it could equally be C++ or COBOL code), provided interfacing to each component follows certain rules. Of particular interest are the CORBA services known as Internet Inter-ORB Protocol (IIOP) that allow components to communicate with each other over the Internet [17]. Using these services, one can hide implementation details or even execution platform and location of the components, making multi-tier architectures possible. CORBA is still in its infancy concerning deployment but despite the initial market reluctance to adopt it, spurred by the decision of Microsoft Corporation to pursue their own Distributed Common Object Model (DCOM), standardization moves at a fast pace. A lightweight version of middleware that can provide vendor independence can be implemented using Java servlets. Servlets are like applets, save for the graphical user interface. They only run in servers and, being genuinely multi-threading, avoid the caveats and latency of CGI scripting. Utilizing the Java Native Methods Invocation scheme, which allows direct calls of C-code from within Java programs, we have managed to implement a fast and reliable 3-tier architecture as we explain in a later paragraph.

Despite their shortcomings the combination of HTML & HTTP have managed to provide a more or less uniform way to transfer content over the Web. With the advent of Style Sheets they can also provide a minimum set of uniform presentation facilities. However, what they cannot cater for, is semantics. Descriptive markup provided by HTML tags such as <H1 is tightly bound to "structure". On the other hand, introducing markup with tags such as <AUTHOR falls into the category of "semantics". To achieve full semantics representation one could easily suggest the adoption of the Standard Generalized Markup Language (SGML). Realistically speaking though,

this is out of the question. Even if one managed to convince the browser producing companies to comply with the 500-page specification of SGML, spending time and money on a fully SGML compliant browser that they would probably have to give away for free, it would be outrageous to expect that the users will have to climb a slow learning curve to see their content semantically marked up on the Web. The World Wide Web Consortium (W3C) has acknowledged this deadlock and has initiated a Working Group that has come up with a rigorous subset of SGML which they call eXtensible Markup Language (XML) in their draft specification. XML manages to maintain an important number of SGML features in its 26-page draft specification and provides remedy for a lot of the HTML handicaps, e.g. XML advocates custom tagsets and n-ary links. All this without loss of the strictness attributed to SGML. XML intends to provide semantic markup facilities, thus allowing communities to formulate their common level of understanding. Such a development could lead to each community having its own *ontology* codified in an XML Document Type Definition (DTD), allowing future knowledge to be exchanged fast and reliably [5, 7, 11].

The ARHON design: a multi-tier architecture

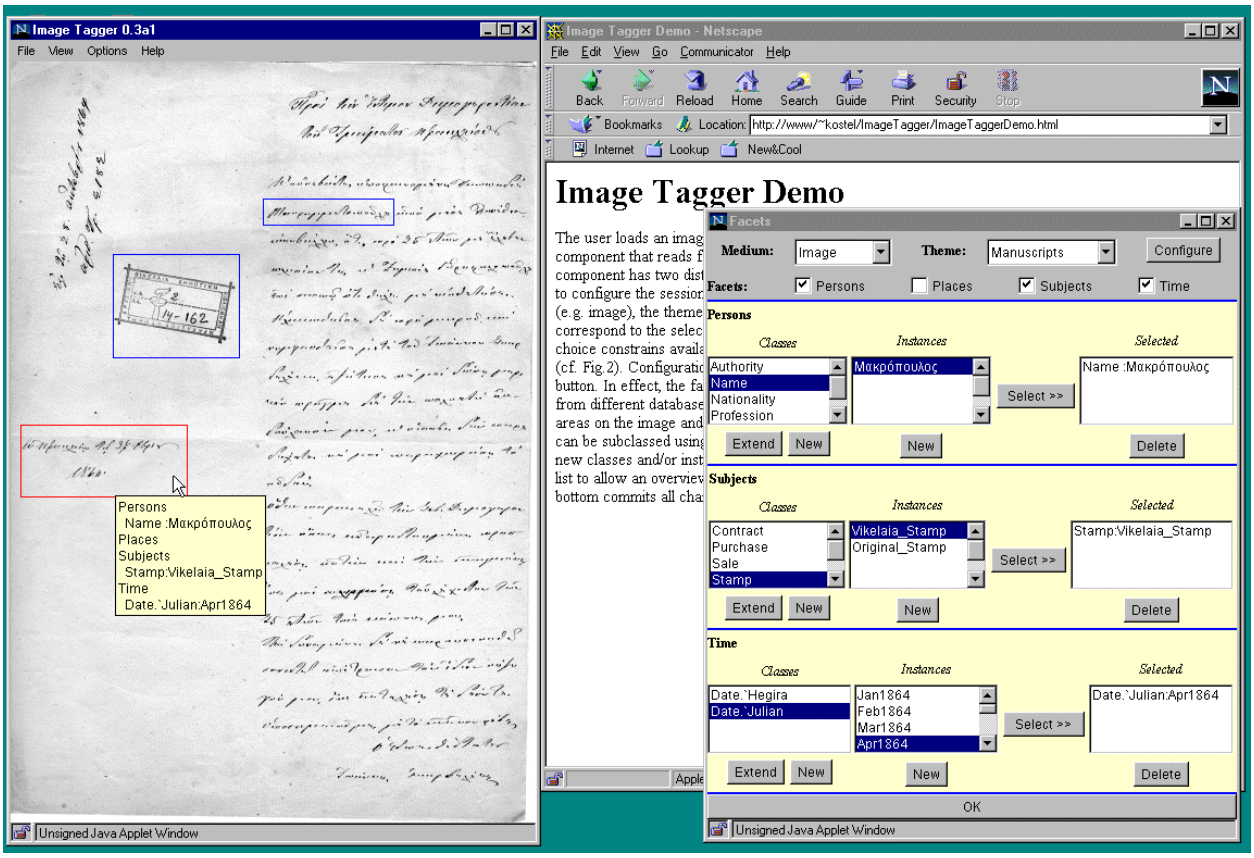

Figure 1*: ImageTagger* is a tool for rapid registering of data in image databases

Most client/server models (2-tier architecture) find hard to cater for the complexity of needs arising everyday. They give way to new models that include many servers, running on different platforms and physical locations distributing data and sometimes management of data. In our application, the first tier (client) is any number of Java-enabled browsers. These clients connect to a Web-server to download applets that handle complex interface tasks, consider error checking on input and present appropriately formulated query results. The second tier consists of servlets, Java programs running on a Web server, which encapsulate the rules, state and logic of our application, that is the management of a digital library. They also control access to the data repository (third tier) using a proprietary legacy API.

The problem

The quality and load of historical manuscripts we had to face in designing this particular digital library has deferred us both from OCR attempts and typed-in transcriptions [1, 13]. On the contrary, we decided to scan the documents and apply image-enhancing techniques to improve their quality, sometimes further than the originals. Since the semantic information appeared in several parts of each document it had to be explicitly identified by scholar researchers at different levels. To facilitate their work we developed *ImageTagger*, a tool for interactive mark-up and annotation of image documents [2]. Among the merits of this tool, as opposed to standard SGML mark-up editors, is that it tags and annotates the image of the document visually. There are a number of technically sound reasons why embedded mark up should be avoided when possible [6]. In our case, practical reasons have led us to the design of *ImageTagger* as a non-embedding markup alternative: it doesn't presuppose that you have a document in ASCII format and it doesn't affect the original, allowing for different treatment of the same data in the future (see figure 1). Also, since it is implemented as a Java applet it allows for remote tagging and annotation and does not depend on a database coming from a particular vendor. In our case, we used an object-oriented semantic net that supports semantic indexing (Semantic Indexing System [3]) developed at FORTH. However, the overall design does not restrict us from using any other DBMS, be it relational or object-oriented. On the contrary, it allows concurrent use of more than one DBMS.

The architecture

Accessing the data for querying or retrieval is achieved by a Java/HTML user-centered interface. A different incarnation of an AccessApplet is served to each client according to his/her status (administrator / scholar / casual user) decided at login time. The dynamic nature of the interface allows the client to change status or even language during a session if appropriate (figure 3 depicts the working version of our interface).

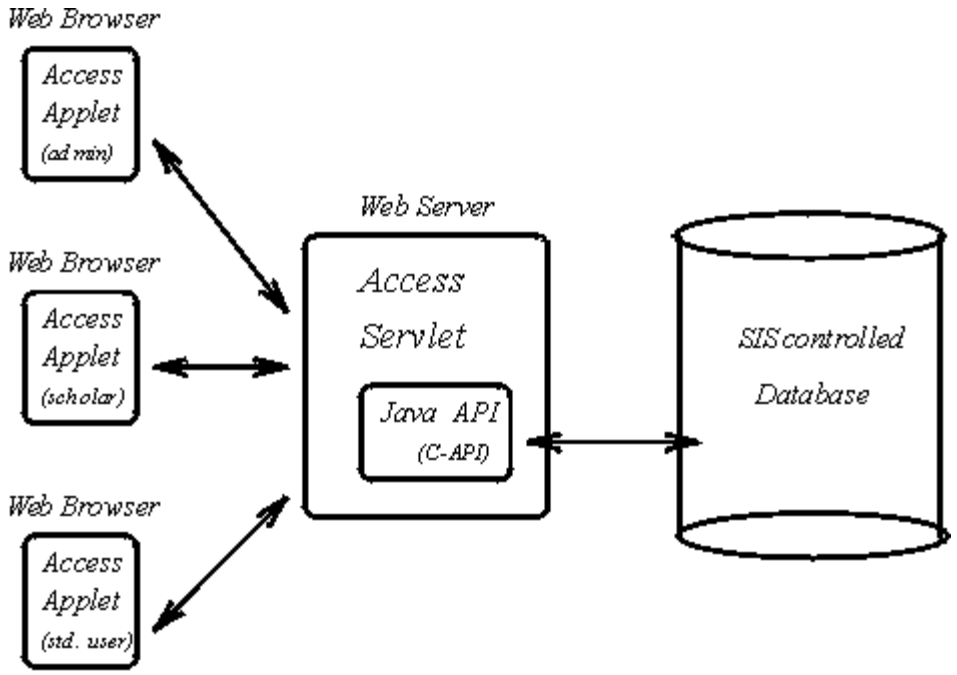

Figure 2: The ARHON architecture

To tackle the lack of state inherent to the HTTP protocol, we propose a server-side solution that is an improvement on "cookies", the current standard solution for state preservation. We rely on

a multithreaded AccessServlet running on the Web-server all the time, delegating after approval all client connections to the database. This servlet is also responsible for monitoring the consistency of transactions and integrity of the database, as it can roll back all unfinished transactions and track simultaneous attempts by users to update the database. Since servlets are programmed in Java we would have either to design and implement a Java API for the semantically indexed database, or convert an existing and proven C-API of the client-server model. This C-API has worked for a number of multi-linked thesauri installed in the past few years with the support and maintenance of the Information Systems and Software Technology group at FORTH. Our choice was to utilize the existing C-API , by creating a Java API as a wrapper, in the Java Native methods Invocation scheme (JNI) which proved a fast and efficient way to utilize legacy code. Initial experiments show that the API interface latency is negligible.

The above sketched architecture (figure 2) has allowed us to surpass a number of problems presented by static HTML. At this point of the project we are turning our attention to digitizing a large amount of data and inserting it in the database so that we can perform efficiency analysis and improvements. Once a critical mass of digitized documents aggregates we intend to research on techniques concerning user profiles and query refinement.

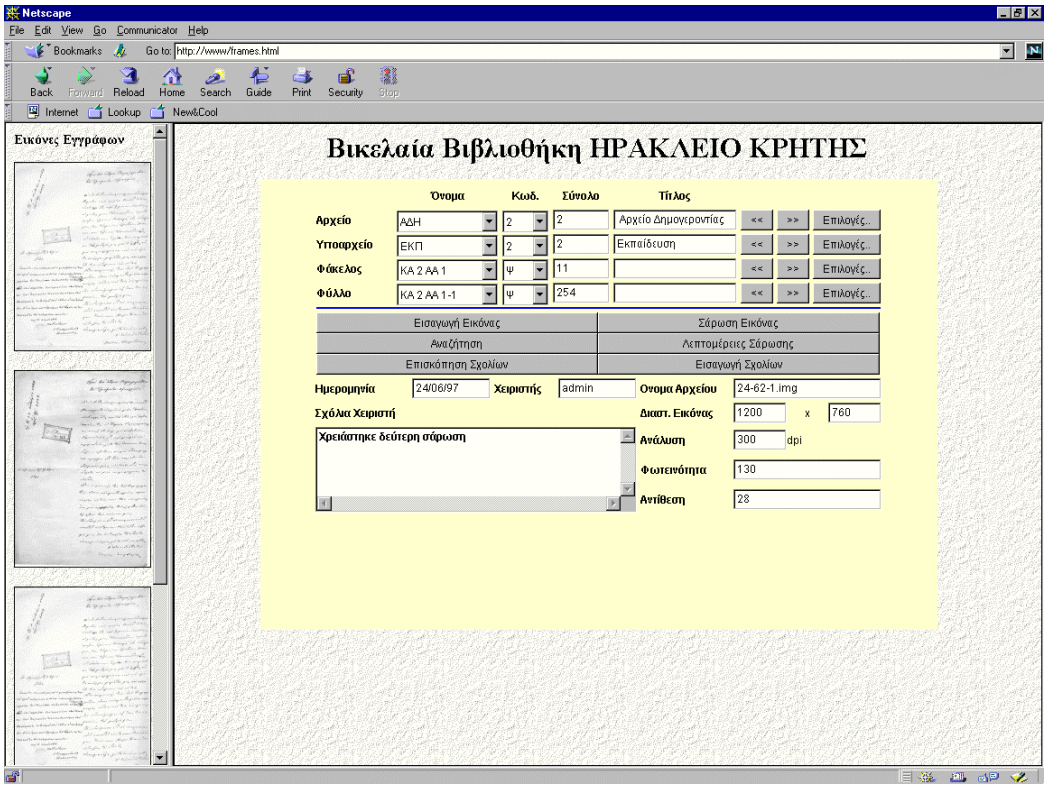

Figure 3: The AccessApplet acting as an interface to our image database

A Note on Security

Although at this stage of the ARHON project Internet access is not be provided at least for an initial period, we have strived to provide the infrastructure that will ease the transition from an Intranet to an Internet use of the digital library. One crucial point to notice, particularly when working with multimedia databases such as ours, is the notion of security. Granting Internet access to a digital library is compared to opening a window at your house to allow other people to look in. Whether they will jump in to steal your belongings is a very complex social discussion. It is obvious that the level of security should reflect the value of the assets and the

expenses the owner (or publisher, in our case) is ready to put themselves to. It is also obvious that with respect to image databases that can be accessed over the Internet there is no perfect security scheme, yet. In this light, what can be done is making the life and works of the potential misuser a bit more difficult. For our particular application, we have devised and implemented a digital watermarking technique that is meant to dissuade people from falsely claiming ownership of the images in our database.

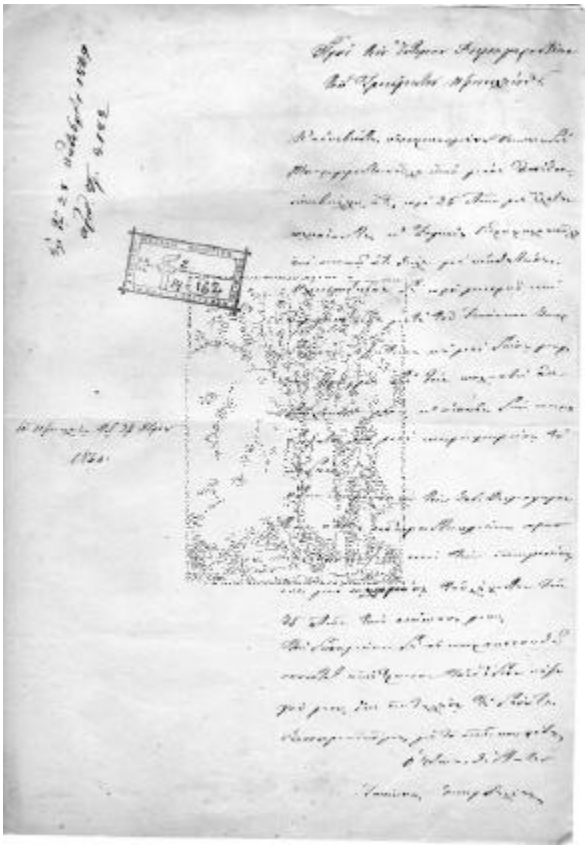

Figure 4. The digital watermark superimposed on a document image

Watermarking techniques are used for the identification of the originator (owner) or the recipient of material. They are usually either visible or invisible [15, 16]. In our case, we devised a method that combines an invisible 128-bit digital signature, hidden in the image of the actual document with a visible watermark that informs potential misusers that this image is under copyright. We have made the visible watermark very robust by interweaving it with our image, making sure it resists statistical or other attacks, which will remove part of the text appearing on the document image along with the watermark (see figure 3). The only way to remove it is by using the actual digital signature hidden in it as the key to an algorithm that we trust with the archivists of the library. This way, either them or the persons they pass their trust to, are the only people who can remove the watermark in order to examine the original document. Of course, this security scheme, like almost every other, will collapse if someone is determined to sacrifice an enormous amount of human and technological resources. However, we feel that it suffices for our cause, which is to make our digital library less vulnerable to malicious attacks.

Conclusions

A Web-based architecture should serve as the connecting glue between subsystems developed or even executing on different platforms. It presents excellent modularity and openness per component. It also allows for extremely fast prototyping. Since nowadays the time factor seems to have shifted from the speed of software execution to the time-to-completion of a reliably

working application (or project) we consider this latter aspect of Web-based architecture very critical. The same architecture imposes uniform interfacing to the data over different platforms, except for minor differences in the layout of each browser. Additionally, with the expected gradual adoption of the Unicode encoding scheme we anticipate enhanced automatic support for multilinguality. Given that the Web is a widely distributed environment the extension of a digital library with new material or even the concatenation of different digital libraries can be easily and transparently achieved [4, 9, 10]. Also, we should point out the great advantage of open standards governing the Web: one is not trapped through this solution to a proprietary format or even a particular vendor and his/her support habits. The number of people currently working on applications of Web technologies is rising, and this, apart from investment protection for the future, guarantees low cost and reliable support during development. Most important, this support can be provided remotely. Until all content is prepared with the Web in mind, there is a clear need to utilize existing technologies and augment them so as to make current content accessible and presentable. In this paper we have provided an example of a multi-tier architecture implemented with state-of-the-art technologies. We feel that this example can provide inspiration for further refinements that will set up the foundations of WBIS relying on existing content.

Acknowledgements

The authors would like to thank Dr. John Immerkaer for constructive reviewing of the image database architecture and technical discussions. The prototype implementation of the digital watermark techniques mentioned in this document were carried out by Ghislain Bidaut.

Production Note:

This document was originally written in HTML. It can be found at http://www.ics.forth/~kostel/papers/delos6.html with its links and references active, where appropriate. It will be maintained and updated by kostel@ics.forth.gr

References:

[1] ARHON: A Multimedia Database Design for Image Documents.
K.V. Chandrinos, J. Immerkaer, P.E. Trahanias, EUSIPCO 98, Special Session on Multimedia Signal Processing

[2] A Visual Tagging Technique for Annotating Large-Volume Multimedia Databases..
K.V. Chandrinos, J. Immerkaer, Martin Doerr, P.E. Trahanias, ERCIM 5th DELOS Workshop on Filtering and Collaborative Filtering, Budapest, Oct. 97

[3] http://www.ics.forth.gr/proj/isst/Systems/SIS

[4] The German National Bibliography 1601 - 1700: Digital Images in a Cooperative Cataloguing Project.
M Doerr, H. Haddouti and S. Wiesener, in Proccedings ACM Digital Libraries '97

[5] XML, Java and the future of the Web.
Jon Bosak in D. Connolly eds., XML: Principles, Tools and Techniques, World Wide Web Journal, Vol 2 (4) 19'97

[6] Embedded Markup Considered Harmful,
Theodor Holm Nelson in D. Connolly eds., XML: Principles, Tools and Techniques, World Wide Web Journal, Vol 2 (4) 1997

[7] XML: A door to Automated Web Applications
Rohit Khare, Adam Rifkin, IEEE Internet Computing, July-August 1997

[8] Meta-HTML: A Dynamic Programming Language for WWW Applications
Brian J. Fox (http://www.metahtml.com)

[9] Building a Digital Library: The Perseus Project as a Case study in the Humanities
Grecogory Crane, in Proccedings ACM Digital Libraries '97

[10] Development of the NIST Virtual Library
Sandy Ressler and Bill Trefzger, IEEE Internet Computing, September-October 1997

[11] The XML FAQ, ver. 1.3, June 1, 1998
Maintained by the W3C XML Special Interest Group

[12] Moving Beyond HTML to Create a Multimedia Database with User-Centered dsign: A case study of a Biological Database.
E. Doerry, S. Douglas, T. Kirkpatrick and Monte Westerfield, Tech. Rep. CIS-TR-97, Computer & Information Science Dept. U. of Oregon

[13] Automating the Structural Markup process in the Conversion of Print Documents to Electronic Text
C. Palowitch, Darin Stewart, in Proceedings of DL 95

[14] Browsing is a Collaborative Process
M. Twidale, D. Nichols, C. Pace, Information Processing and Management, Vol. 33 (6) pp. 761-783, 1997

[15] Safeguarding Digital Library Contents and Users: Digital Watermarking
F. Mintzer, J. Lotspiech, N. Morimoto, D-Lib Magazine (http://www.dlib.org), December 1997

[16] Digital Signature of Color Images Using Amplitude Modulation.
Martin Kutter, Frederic Jordan, Frank Bossen in Proceedings SPIE-EI97, 1997

[17] Client/Server Programming with Java and CORBA,
R. Orfali and D. Harkey, 2nd ed., Wiley, 1998

[18] Protecting Ownership Rights through Digital Watermarking,
Berghel, H. and L. O'Gorman, IEEE Computer, 29:7, pp. 101-103 (1996)

[19] Database Backed Websites, *P. Greenspun, ZD-Press, 1997*