Towards a Richer Web Object Model*

Frank Manola
Object Services and Consulting, Inc.
151 Tremont Street #22R
Boston, MA 02111
fmanola@objs.com
http://www.objs.com/

1. Introduction

The World Wide Web is becoming an increasingly important factor in planning for enterprise distributed computing environments, both to support external access to enterprise systems and information (e.g., by customers, suppliers, and partners), and to support internal enterprise operations. Organizations perceive a number of advantages in using the Web in enterprise computing, a particular advantage being that it provides an information representation which

- supports interlinking of all kinds of content
- is easy for end-users to access
- supports easy content creation using widely-available tools

However, as organizations have attempted to employ the Web in increasingly sophisticated applications, these applications have begun to overlap in complexity the sorts of distributed applications for which distributed object architectures such as OMG's CORBA, and its surrounding Object Management Architecture (OMA) [Sol95] were originally developed. Since the Web was not originally designed to support such applications, Web application development efforts increasingly run into limitations of the basic Web infrastructure.

If the Web is to be used as the basis of complex enterprise applications, it must provide generic capabilities similar to those provided by the OMA (although these may need to be adapted to the more open, flexible nature of the Web, and specific requirements of Web applications). This involves such things as providing database-like services (such as enhanced query and transaction support) and their composition in the Web. However, the basic data structuring capabilities provided by the Web (its "object model") must also be addressed, since the ability to define and apply powerful generic services in the Web, and the ability to generally use the Web to support complex applications, depends crucially on the ability of the Web's underlying data structuring facilities to support these complex applications and services.

2. Increasing the Structuring Power of the Web

The basic data structure of the Web consists of hyperlinked HTML documents. It is generally recognized that HTML is too simple a data structure to support complex enterprise applications [Bos97], e.g.:

- applications that require the Web client to function as the front-end to enterprise applications or mediate between multiple heterogeneous databases,
- applications that require more flexibility in distributing processing load between Web servers and clients, and
- applications that require the Web client to present different views of the same data to different users or in which intelligent Web agents need to tailor information discovery to the needs of individual users.

This is primarily because HTML tags deal primarily with the *presentation* aspects of document structure; they do not permit the straightforward tagging of semantically-meaningful units of data.

A fundamental direction of efforts to address HTML limitations has been attempts to integrate aspects of *object technology* with the basic infrastructure of the Web. There are a number of reasons for the interest in integrating Web and object technologies:

- The Web can already be viewed as a simple form of distributed object system in which HTML pages are considered as objects, having identity provided by URLs, and methods defined by, or that are invoked via, HTTP servers. The basic resemblance of the Web to a simple object system has created a natural interest in seeing

* This research is sponsored by the Defense Advanced Research Projects Agency and managed by the U.S. Army Research Laboratory under contract DAAL01-95-C-0112. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, U.S. Army Research Laboratory, or the United States Government.

how far the resemblance can be further developed.

- Object technology is seen as a particularly-convenient way of adding functionality (e.g., behavior) to the Web, both by adding the behavior provided by objects to the static content of HTML, and by allowing Web clients and servers, through distributed object technology, to access other computing resources. For example:
  - Java can be added to Web pages and, once downloaded to the client, can then execute. In some cases, the client objects then interact with server objects, possibly using a different protocol, e.g., OMG's IIOP or Java's RMI.
  - Web pages can be treated as objects or collections of objects for use by other code. Dynamic HTML developments by Microsoft and Netscape, and current World Wide Web Consortium (W3C) work on a Document Object Model (see below), allow the contents of an HTML document to be treated as a collection of programmable objects. Client-side code can then access these objects and manipulate them dynamically (e.g., causing immediate changes in the document displayed to the user).

In addition, work within the database community on "lightweight object models" for semi-structured data [PGW95, BDHS96] has attempted to apply selected object concepts in describing Web-structured data, and to use those models as a basis for building query and other database services.

Such efforts contribute valuable ideas toward making Web-related technologies capable of directly supporting a wider variety of activities, in more flexible and extensible ways. However, until recently these efforts have still been based on HTML, with its basic structuring limitations, and much effort is expended in this work attempting to deal with these limitations. Moreover, some of these efforts have not been well-integrated with key areas of emerging industry consensus on Web data structuring technologies.

If the Internet is to develop to support advanced application requirements, there is a need for both richer individual data structuring mechanisms, and a unifying overall framework which supports heterogeneous representations and extensibility and provides metalevel concepts for describing and integrating them.

3. Components of a Web Object Model

Under the auspices of the World Wide Web Consortium (W3C), industry is currently developing a number of component technologies addressing the limitations of current Web data structuring technology. Several of these separate "threads" of Web standards development can be combined to form the basis of a richer Web object model to address the requirements of more powerful Web applications. This combination is based on the observation that key components of any object model are:

- data structures that can represent *object state*
- ways to associate behavior (*object methods*) with the object state
- ways for the object methods to access and operate on that state

Applying this idea to the Web environment, Web pages can be considered as state, and objects can be constructed by enhancing those pages with additional *metadata* that allows the pages to be considered as objects in some object model. In particular, it is desirable to enhance Web pages with metadata consisting of programs that act as object methods with respect to the "state" represented by the Web page.

Thinking in this way, the Web object model could be improved by providing:

- a richer base representation than HTML, in order to better represent "object state" (in particular, provide better support for semantic identification of fields, rather than simply supporting the presentation aspects of text)
- an API to this state, so that programs can readily access it (without complex parsing)
- an enhanced ability to define relationships between this state and specified pieces of code that can serve as object methods, to provide an enhanced ability to compose objects from both existing and new Web resources

At the same time, the openness of the Web compared to conventional object models needs to be preserved, due to the distinct requirements of the Web environment for openness and scalability.

The following sections briefly describe several W3C technologies addressing these issues.

3.1 The Extensible Markup Language (XML)

W3C's Extensible Markup Language (XML) <http://www.w3.org/XML/> is a data format for structured document interchange on the Web. XML defines a simple subset of SGML (the Standard

Generalized Markup Language), and is intended to make it easy to use SGML on the Web. Unlike HTML, which defines a fixed set of tags, XML allows the definition of customized markup languages with application-specific tags, e.g., <AUTHOR> or <QTY-ON-HAND>, for exchanging information in particular application domains such as chemistry, electronics, or general business. As a result, XML provides direct support for using application-specific tagged data items (attribute-value pairs) in Web resources, as opposed to the current need to use ad hoc encodings of data items in terms of HTML tags. [KR97] provides a useful overview of the potential benefits of using XML in Web-related applications.

The XML specifications include several parts. The XML (language) specification <http://www.w3.org/TR/PR-xml-971208> defines specifications for Document Type Definitions (DTDs), and documents conforming to them. A DTD is a formal definition of a particular type of document. This acts like a database schema, and defines what names can be used for elements, where they may occur (e.g., <STREET> may only be meaningful inside <ADDRESS>), and how they all fit together. The linking of resources with their DTDs is similar to the association of a database record with its schema type, and to the association of an object with its type or class definition. The XML linking model is described in the XLL draft <http://www/w3/org/TR/WD-xml-link>. The full hypertext linking capabilities of XML are much more powerful than those of HTML, providing support for both bidirectional and multi-way links, as well as links to a span of text (i.e., a subset of the document) within the same or other documents. XSL <http://www.w3.org/TR/NOTE-XSL.html> is a draft specification defining stylesheet capabilities for XML documents. XML stylesheets enable formatting information to be associated with elements in a source document to produce formatted output. XSL capabilities are based on a subset of the ISO Document Style Semantics and Specification Language (DSSSL) [ISO96] used in formatting SGML documents. The XML language specification is fairly complete; the XLL and XSL specifications are still under development.

XML has considerable industry support. For example, Microsoft has built an XML parser into Internet Explorer 4.0 (which uses XML for certain applications), has made available an XML parser in Java <http://www.microsoft.com/xml/>, and has contributed a number of proposals to W3C on the use of XML as a base for various purposes. A number of industry groups have defined SGML DTDs for their documents (e.g., the U.S. Defense Department, which requires much of its documentation to be submitted according to SGML DTDs). In many cases these could be either used with XML directly or converted in a straightforward fashion. Work is already underway to define XML-based data exchange formats in both the chemical and healthcare communities. Work has also been done on other applications of XML, and numerous articles are beginning to appear in the trade press about XML.

3.2 The Document Object Model

W3C's Document Object Model (DOM) <http://www.w3.org/TR/WD-DOM/> defines an object-oriented API for HTML or XML documents which a Web client can present to programs that need to process the documents. Through this API, scripts or programs access and manipulate a document's content (including all markup and any DTDs) as a collection of objects. The client (at least conceptually) operates off this collection of objects in displaying the document. By operating on the collection of objects representing the Web page, scripts or programs can change styles and attributes of page elements, or even replace existing elements with new ones, causing direct changes in the document's presentation. As a result, DOM allows the implementation of dynamic content on the client, rather than forcing all such implementation to be done on the server (with the associated need for additional client-server message traffic). It also provides a basic way to integrate a document's data with processing code. In addition to using DOM itself, a Web client could provide a DOM interface to external applications, allowing them to access the document via the client. DOM is a generalization of Dynamic HTML facilities defined by Microsoft and Netscape. DOM Level 1 extends these capabilities to, for example, allow creation "from scratch" of entire Web documents in memory by creating the appropriate objects.

DOM is currently under development, and further work is required to integrate it with other Web technology developments. For example, SGML's DSSSL (mentioned in Section 3.1) defines a very general object model for runtime processing of SGML documents called *groves*, which resembles the DOM in some respects. However, it is not yet clear whether (or to what extent) DOM and grove capabilities will be integrated, since their intended uses are somewhat different.

3.3 Web Metadata Mechanisms

The W3C has done considerable work on metadata mechanisms for describing various characteristics of Web resources. Relevant W3C metadata technologies include the Platform for Internet Content Selection (PICS) <http://www.w3.org/PICS/> and its generalization, the Resource Description Framework (RDF) <http://www.w3.org/Metadata/RDF/> Web metadata models. PICS defines a set of technologies for defining machine-readable descriptions of rating systems for labeling Internet content, generating content labels according to those rating systems, associating labels with specific Internet resources, and distributing the labels to Web clients as the resources are accessed.

The RDF combines extensions of the PICS technology with other work on metadata models such as Netscape's Meta Content Framework [GB97]. The intention is to support more general metadata applications, including resource discovery by search engines, cataloging, knowledge sharing and exchange by intelligent software agents, and electronic commerce. RDF <http://www.w3.org/TR/WD-rdf-syntax/> defines both a data model for representing RDF metadata and an XML-based syntax for expressing and transporting the metadata. The basis of RDF is a model for representing named properties and their values. This is based on propositional logic, plus certain modalities. RDF properties serve to represent both attributes of resources and relationships between resources. A property is a 3-tuple consisting of the resource being described, a property name or type, and a value. The resources being described and the values describing them are conceptually nodes in a directed graph, with the edges being labeled by the property names. A collection of property triples describing the same resource is called an *assertions* [sic]. RDF allows the reification of properties, so that individual properties and assertions can themselves be described by properties. Assertions may be associated with the resource they describe in several ways, e.g., embedded in the resource, external to the resource but supplied with the resource in the same retrieval transaction, or retrieved from a separate source. The RDF URL cited above includes a number of examples showing how RDF can be used in supporting different metadata requirements.

4. Discussion

XML, DOM, and RDF constitute important enhancements to the basic infrastructure of the Web. Moreover, they are being developed by industry as part of Web standardization efforts, rather than as individual proprietary activities, and hence are likely to see widespread use. By beginning to build on these foundations, it should be possible both to support enhanced Web facilities, and to perform more productive research on adding capabilities to the Web, since less work will be needed simply to deal with fundamental Web structural deficiencies.

For example, because it directly supports application-defined attribute-value pairs, XML can be directly used to support database record structures (including nesting), and hence provides a much better basis for higher-level database facilities than HTML. Together with its advanced linking facilities, it subsumes the power of many semi-structured data representations. Work has already been done on query languages for SGML (e.g., [CACS94]), and this work could be adapted to XML (DSSSL also defines a form of query language called SDQL for accessing parts of SGML documents for use in formatting). The provision of query languages for XML, together with associated algebras and calculi, would provide an important base for the development of full-fledged database processing capabilities for Web documents represented in XML. View facilities would also be helpful in providing for efficiently-constructed customized presentation of Web data. Work on views could draw on both the query language work, and also on the DOM and DSSSL grove object models for representing structured documents (we are currently exploring both query languages and views in the context of XML and DOM). These same technologies can also play other valuable roles in distributed object architectures (for example, XML could be used as the basis for a serialization representation for moving objects within a network).

In addition, these Web technologies, together with others, provide the potential for constructing actual Web objects from Web data and code resources. Specifically:

- XML provides a good representation for object state, including both application-specific tagged data elements and nested structures, and its more powerful linking facilities. In supporting an object model, XML pages (like HTML pages) can also be used as containers for embedded objects and object methods (e.g., Java applets)
- DOM provides an API for XML documents used as object state, and hence provides a mechanism for integrating object state and associated code.
- the OBJECT element from HTML 4.0 (a generalization of the APPLET element) can be used for representing embedded object methods
- non-embedded object methods can be represented by arbitrary pieces of code defined as or within Web resources separate from the state (e.g., code libraries identified by URLs) that are designed to access the state via its DOM-based interface
- concepts from PICS and RDF can be used to define both conventional types of metadata (e.g., schema or type information), to define relationships between documents containing state and separate documents containing code representing object methods, and to support automatic access to these methods when the state is accessed.

A full technical report [Man97] explores this idea more fully. In addition, it:

- more thoroughly describes key examples of existing work from the Web, database, and OMG communities (including those mentioned above) that contribute both ideas and technology toward providing the components of a Web object model
- identifies some key underlying principles behind this work
- identifies a framework which allows this work to be unified and extended to support the requirements of advanced Web applications for object technology

5. Conclusions

A more complete integration of object technologies into the Web would provide the basis of both powerful capabilities for integrating all kinds of data and information, and a wide variety of enhanced services, within a distributed architecture that is both widely-available, and easy-to-use and extend. Industry-developed Web technologies, such as XML, DOM, and RDF, contribute greatly toward this integration, and provide the basis for both enhanced Web products, and more productive Web-related research.

References

[BDHS96] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu, “A Query Language and Optimization Technique for Unstructured Data”, *Proc. SIGMOD ’96*, 505-516.

[Bos97] J. Bosak, "XML, Java, and the Future of the Web" <http://sunsite.unc.edu/pub/sun-info/standards/xml/why/xmlapps.htm>, 1997.

[CACS94] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl, “From Structured Documents to Novel Query Facilities”, *Proc. SIGMOD ’94*, 313-324.

[GB97] R. Guha and T. Bray, “Meta Content Framework Using XML”, <http://www.w3.org/TR/NOTE-MCF-XML/>, June 6, 1997.

[ISO96] International Standard ISO/IEC 10179:1996(E), Information Technology—Processing Languages—Document Style Semantics and Specification Language (DSSSL), 1996.

[KR97] R. Khare and A. Rifkin, “XML: A Door to Automated Web Applications”, *IEEE Internet Computing*, 1(4), July-August 1997, 78-87.

[Man97] F. Manola, "Towards a Web Object Model" <http://www.objs.com/OSA/wom.htm>, 1997.

[PGW95] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom, “Object Exchange Across Heterogeneous Information Sources”, *IEEE Intl. Conf. On Data Engrg*., 251-260, Taipei, March, 1995.

[Sol95] R. Soley (ed.), *Object Management Architecture Guide*, Third Edition, Wiley, June 1995 <ftp://ftp.omg.org/pub/docs/ab/95-05-05.ps>.