

PLAIN
WORDS

Enter
→

An XML Briefing

One of a series of free
articles from Plain Words



An XML Briefing

John L. Mobbs

Executive Summary

This very brief document cannot possibly do justice to a subject as vast as XML and can be seen only as a very brief overview. This implies that many concepts are merely stated rather than explained with examples although explanations and example are readily available.

The document sets out to describe XML and some of its associated technologies, lists some of the current applications, looks at some issues related to implementing an XML documentation system and offers some thoughts on legacy conversion.

Description

What is XML?

The eXtensible Markup Language (XML) is a markup language with superficial similarities to the Hypertext Markup Language (HTML) which has been the language of the web since its inception. XML is web compatible and was developed to provide capabilities which HTML could never provide. To that extent, it is the next development beyond HTML. Both have been developed by, and the standards are maintained by, the Worldwide Web Consortium (W3C).

Formally, the roots of XML are in the Standard Generalised Markup Language (SGML) of which XML is a subset. An XML document is, therefore, SGML compliant and transition between the two formats is relatively straightforward. HTML is, by contrast, an instance (or implementation) of SGML. An HTML document is not SGML compliant.

How XML Represents a Step Beyond HTML

The crucial characteristic of XML is that the structure of an XML instance is constrained by an external (or can be internal) definition. This definition specifies what content can be included and in what order. With this aspect of an XML document under control, it then becomes possible to read the XML instance with software which is able to distinguish between, and manage, the different parts of the content. One way of explaining this is to contrast it with HTML.

There are two fundamental differences between HTML and XML:

- The HTML standard specifies the markup (tags) which may be used; HTML 4 has in excess of 100 tags and every browser must be able to interpret all of them. The XML standard does not specify any tags; a tagset containing as many or as few tags as are required can be specified separately for each individual document. The tags are specified in the Document Type Definition (DTD) or, more recently,



the Schema. It is the DTD or Schema to which the XML instance must conform to be described as “valid”.

- HTML markup is concerned only with formatting, not with content; the only use a browser makes of HTML markup is how to render the document on a screen. XML markup, by contrast, is concerned primarily with content and only peripherally with formatting. This is why it is not possible to “read” an HTML document with software and make sense of it in terms of its content.

There are, officially, 10 goals of XML and they can be examined at <http://www.w3.org/TR/REC-XML> .

XML Architecture

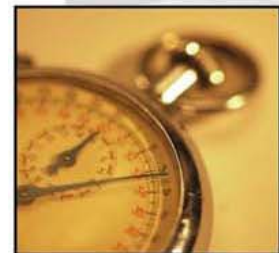
There are several building blocks which figure in the XML world. One approach is to list them in the sequence in which they might be important.

- The DTD has already been mentioned. Common practice is use a standard DTD for a whole suite of XML instances. If two separate sites use exactly the same DTD, a valid XML document prepared on one site will be valid on any other participating site. In view of this, whole industries have standard DTDs so that they can exchange documents freely.

The DTD is a legacy, albeit a very useful one, from SGML and it has limitations. It is written in a language of its own and so a designer must learn that language (not difficult) to write a DTD in the first place. Another limitation is that it does not generally support data typing. Schemas, however, support data typing, tighter control of structure and are, themselves, written in XML.

- The XML instance comes next. The word “instance” is used to avoid limiting the field merely to documents and publications. XML has been used to describe scaleable vector graphics (SVG), financial information, mathematical formulae, molecular structures, DNA and protein sequencing, astronomical data, musical notation, weather observation reports, theological information and marking up liturgical texts, the list goes on.

There is a basic set of syntax rules which must be observed. HTML is very lax and permits imprecise coding. XML is wholly intolerant. An instance which obeys the rules but is not necessarily valid is called “well formed”. To be valid to a DTD/Schema, an instance must also be well formed.



- An XML document (staying with documentation) is of little use as it stands and must be “transformed” into a useful format. This is the role of the XML processor of which there are several excellent freeware products available. The processor takes its instructions from both the XML document and the eXtensible Stylesheet Language - Transformation (XSLT). Two popular output formats are HTML and PDF. The XSLT for HTML consists of a set of templates which the processor populates with content from the instance. HTML represents a very useful and widely used output which can be produced automatically from the same (single) source as any other output required. Direct production of PDF from XML involves another incarnation of XSL known as XSL FO (formatting objects). At present this is less developed than XSLT. (The use of XSL FO is not essential to producing PDF; there is a quite viable workaround using Word.)
- An XSLT is able, where required to locate a particular node in the instance and this technology is called XPath. It is possible to query an XML instance using the eXtensible Query Language (XQuery). The list goes on.



Benefits of XML

Use of XML confers the following advantages:

- The data is held in a format which is governed by a world standards body (Worldwide Web Consortium - W3C) and is therefore independent of any proprietary formats such as MS Word, FrameMaker, Interleaf and others.
- It is a subset of SGML (Standard Generalised Markup Language) and has been designed for interoperability with HTML, specifically for delivering information over the web.
- The content can be read by software and be sorted, filtered and re-arranged for different audiences.
- A single source of data can be output to any number of different formats.
- Multiple languages can be held in a single source file thus virtually eliminating the possibility of mistakes in compiling foreign language documents. (The different languages are tagged with language-specific tags.)
- Data is both application-independent and platform-independent.
- It is possible to exchange data with other users in certain knowledge that if they are using the same Document Type Definition (DTD), the files will be 100% compatible.
- This is the way in which all manner of types of documentation are developing worldwide and so provides the best form of future-proofing which is currently available.



Some Issues concerning Implementation

Skill sets

There seem to be two basic skill areas associated with XML and both are quite separate from the skill set for a Technical Author. The minimum set of skills seems to be:

- **DTD/Schema/XSL.** This is the ability to write DTDs/Schemas and XSL (both XSLT and XSLFO). DTD/Schema skills and XSL skills are grouped together because the two documents are so closely related that is unrealistic to separate them and also that a person who does not have both skill areas is unlikely to be worth engaging.
- **Data entry.** The ability to use a high level tool to enter text (paragraphs, lists and tables) and graphics into an XML document so that it conforms to the specified DTD/Schema.
- **Authoring Skills.** Information gathering, interviewing, information structuring, explaining, generating content etc.

Staff

- It is possible that the Technical Author and the DTD/Schema/XSL writer can be the same person but it is possible for these roles to be carried out by different people.
- The skill set which is identified above as Data Entry could probably be carried out by a “technician level” person who is intelligent enough to handle the XML tool(s) and has keyboard skills but not necessarily any of the other skills identified.

Tools

- The DTD/Schema/XSL writer needs, at least a colour-highlighted text editor (such as XMLwriter) and possibly a higher level tool such as XML Spy.
- The technician-assistant may wish to have a WYSIWYG XML content editor such as XMetaL. It should be acknowledged that a DTD/Schema/XSL tool, such as XML Spy, will be complicated and far from easy to learn. It seems unlikely that a technician-assistant would be comfortable with such a tool. If they are, this can bring advantages of economy, commonality and training.
- The Technical Author needs, as a minimum, only a word processor but if they use an XML content editor (ideally but not necessarily the same one as the other staff), this will avoid a lot of coding which will probably have to be carried out (or at least edited) manually.
- A WYSIWYG editing environment is a mixed blessing. It is superficially attractive but always seems to impose limitations on the editing which is required.



Types of work

- If much of the work is concerned with converting legacy material, it may be cost-effective to engage a technician-assistant as well as the DTD/Schema/XSL writer. If most of the work is generating new content, a multi-skilled person can, if need be, work alone or as part of a team whose members all have Technical Authoring and XML Data Entry skills.



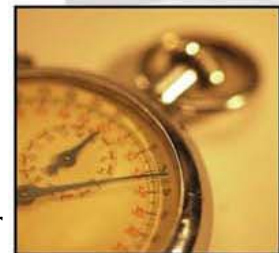
In both cases, DTD/Schema/XSL writing should be essentially a one-off job (albeit a substantial one) which should be substantially completed early on in an XML implementation. The Authoring and/or Data Entry tasks do not necessarily need to be dependent on completion of the DTD/Schema/XSL suite. In reality, it will probably turn out that a draft DTD or Schema can be used initially to enable Data Entry to start and the XSL and (inevitable) refinements to the DTD/Schema can run in parallel with other work.



A Possible Procedure

One possible, and arguably quite cost-effective, procedure could be:

1. Identify which project (development or legacy conversion) is to be the launch project for XML.
2. Engage a consultant who has, at least, the skills identified above as **DTD/Schema/XSL**. The consultant will, hopefully, also have wide experience as a Technical Author across several different industries.
3. The consultant will:
 - Carry out a documentation audit, make appropriate proposals according to the brief and reach agreement on an implementation plan.
 - Prepare a DTD or Schema in line with the expected end use of the documents.
 - For product development, optionally create the content in XML or train an in-house author to do so, possibly using a high level tool (WYSIWYG interface) according to the available skill levels of in-house staff.
 - For legacy conversion, advise on the use of proprietary tools or develop macros where possible to automate the process.
 - Prepare an XSLT to produce the required output format.
 - Provide continued support, possibly on a part-time basis to develop in-house skill to a level of complete independence.



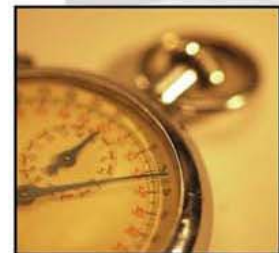
A Note about Legacy Conversion

There are tools on the market which claim to output XML from various applications and forms of input. Some of the more expensive ones even claim to make an attempt to convert unstructured documents into XML. What is less clear about such tools is how much additional work would be required on the resulting XML document to make it valid to a specified DTD/Schema.

An alternative, and arguably more cost-effective, approach is to semi-automate a conversion process using macros in Microsoft Word. In this way it is possible to write the macros to match a specified DTD so minimising any “clean up” work on the resulting marked up file. The presence of a human at every stage of the process also helps to ensure that the conversion which takes place is what was expected so preventing a process from producing wholly unexpected results.

Porting a source document into Word from whatever format is generally quite straightforward.

This technique is proven and works reliably.



Plain Words – The Big Picture

Technical Writing & Course Development

Planning, writing and producing Help systems, HTML, on-line material, traditional manuals. Designing and writing bespoke courses for your products and services, whether tutor-led, CBT or e-learning. Presenting 'train the trainer' or end user courses.



Specialist Search & Selection

Helping you to recruit permanent or contract documentation staff:

- Technical Authors & Editors
- Trainers
- Localisation Specialists
- Information Managers
- Documentation Managers
- Copy Writers
- Project Managers
- Project Coordinators
- Instructional Designers
- Bid & Proposal Writers
- Web Content Developers
- Knowledge Managers

Writing Skills Training Courses

- Structuring & Writing Reports
- Designing & Writing Help Systems
- Writing Winning Bids & Proposals
- Writing Effective Letters & Emails
- Designing & Writing Technical Documents
- Indexing On-line & Printed Material
- Estimating & Planning Technical Documents
- Writing for the Web



Bids Consultancy & Training

Bidworker, a division of Plain Words, is dedicated to providing bids consultancy and training. Writing, editing and formatting of bids and proposals. 'How to Write Winning Bids and Proposals' training course. Plain Words own **bid**worker™ software to automate much of the process of compiling proposals.

"Excellent service, highly skilled authors. All deadlines have been met!"
Giovanni Calamida, European Patent Office

Catalogues

Plain Words **catalogue**worker™ enables you to create high quality printed or online catalogues in minutes. This is a simple and speedy way to create large catalogues that would otherwise need to be built, fully typeset and formatted. It also caters for multi-currency, multi-languages and different discount schemes.



Publicity & Copywriting

Inject selling power into your communications with Plain Words' advertising and PR services. Writing or editing sales copy, web content, articles and newsletters. Publicity campaigns and promotion via the media or through your website.

Other Services

Graphic Design • Printing • Localisation & Translation

Technical Writing • Training • Search & Selection • Bids Consultancy
Call us on +44 (0)1635 202013

©Plain Words Limited 2004. All rights reserved.

