
THE BIG BANG - XML EXPANDING THE INFORMATION UNIVERSE

S. Rutt, M Chamberlain, G Buckley
STATS-NNC Limited, United Kingdom

Email address of main author: Stephen.Rutt@stats-nnc.co.uk

THE BIG BANG – XML EXPANDING THE INFORMATION UNIVERSE

XML (eXtensible Mark-up Language) is a part of a family of mark-up languages that date back to 1969. It was then that Charles Goldfarb, working for IBM, developed a means of indexing and cataloguing data by means of inserting descriptive mark-up tags around the data content. This was called the Generalized Mark-up Language and was to become the basis for the ISO standard defined in 1985 and now known as SGML (Standard Generalized Mark-up Language). Although SGML has been a success within large-scale organisations it is complex, expensive to implement and does not fit into our modern world of browsers and mobile devices.

One successful SGML application that has been developed is HyperText Markup Language (HTML). HTML was originally developed by Tim Berners-Lee working for CERN, the Swiss based particle accelerator, as a means of swapping information and ideas across a network. It provides a set list of simple descriptive mark-up tags and has since become hugely successful as the language of the World Wide Web. However, HTML does suffer from a few shortcomings; principally the ability to search content since the limited number of descriptive tags does not adequately describe the content; plus many web-authors tend to use the tags to style content rather than describe it.

In 1996 the World Wide Web Consortium (W3C) began discussions to address the inadequacies of HTML by defining a new mark-up language with the power and extensibility of SGML but with the simplicity of HTML. This language was called the eXtensible Mark-up Language (XML) and became an approved specification in 1998.

XML soon became a great success with many standard XML applications being developed such as Scalable Vector Graphics (SVG) for defining illustration and animation, mathML for defining mathematical equations and, more importantly, eXtensible Stylesheet Language (XSL) used for styling and transforming XML documents. Many other XML application exist; a useful resource is <http://www.w3.org>.

Industries are adopting XML as a means of making disparate systems talk with each other or as a means of swapping information between different organisations and different operating systems by using a common set of mark-up. More important to this discussion is the ability to use XML within the field of Technical Documentation and Publication.

Traditional Technical Documentation Methods

Traditional technical documentation is usually authored within a Word Processor environment. Although modern Word Processors provide the end users with great flexibility and richness of application they suffer some severe shortcomings when used to produce consistent technical documentation. Not only is consistency within one particular document difficult to maintain but also across entire suites of documentation where numerous authors have provided input. This amounts to common documents with different styling; different numbering systems and inconsistent headers and footers. Even with use of specified templates authors often tend to drift towards their own formatting styles; often applying style because it meets with their own aesthetic appreciation rather than document specification.

An author also needs to have formatting skills since all content needs to be styled and formatted. It has been estimated that up to 50% of an authors time can be taken with styling and formatting and these issues remain with the author right up to the time of final

publication; often hampering delivery deadlines. Once the style has been applied, it is then locked in to the content, so re-use is accompanied with re-styling.

Reuse of common information within the Word Processing environment is not easily done and even more difficult to manage. Take, for instance, a common Warning: this has to be inserted by either copy and paste or Autotext, then possibly styled for each instance. If a change is needed to this Warning someone has to search out each instance and physically change the content. If the Warning has been used across an entire system the management of such a task is colossal. If we take an example of a company signoff, and the company name or address is amended, it is not difficult to imagine how many items of documentation need to be changed.

As well as style being locked into content, content is locked into the software application and its proprietary file format which in most cases is binary and unreadable by anything other than the host application. This means that when the software is upgraded, your documentation needs to be upgraded. Even worse is when there is a shift in industry preferred software such as when Microsoft Word became the dominant software over WordPerfect – even to this day companies are having to convert their old documents due to this legacy. And if delivery of the published document is required in different formats i.e., Word, PDF and HTML, then an independent document instance has to be created and restyled. Any updates to the content then have to be carried out in each instance greatly increasing the workload.

The Promise of XML Documentation

When using XML as a method of producing documentation we release ourselves of many of the burdens afflicting the Word Processor. Within an XML environment, content, and style are separated and structure is enforced. An XML document is purely content - but not just plain content; it is marked up content, which makes it a rich data source that can be reused and repurposed easily and efficiently.

Because style is independent of the content the author no longer has to worry about applying style; all they have to do is input the data which, after all, is what they are being paid to do. So, as well as utilising 100% of their time in creating content they spend no time in the 11th hour addressing formatting issues which mean deadlines can be met. Style and format is supplied by an independent stylesheet and can be applied across all documentation, thus maintaining consistency.

When it comes to reuse of common information, XML has major advantages over the traditional publication methods. Take for instance the example of the common Warning as mentioned previously. If we define this in XML as a separate file entity, we can then link this into each instance within each appropriate document. This makes it easier to insert; it means that consistency is maintained, it also means that it will take the appropriate style for the appropriate place in the document, plus, management is easy – change the file entity and it automatically updates across the entire documentation. Of course sometimes we might need this versioning, and because XML is describing its data we can easily call up a list of instances and apply the appropriate version to the appropriate documents.

As well as reuse, we can repurpose. Use the same data, apply a different stylesheet and our document can be published in a completely different file format without having to restyle or create new document instances.

The XSL stylesheet language that is used is much more than just a styling language; it can manipulate and transform XML documents, then output them in a variety of software formats. It is more like a programming language providing string manipulation and mathematical calculation.

As well as being freed of the tie between content and format, XML also frees us from software application. XML is an internationally accepted standard as defined by the W3C.

It is a text based format so that it can be read by a simple text editor. This means that we are no longer locking ourselves into proprietary binary formats. If our preferred XML editor upgrades, our documents still apply. If we choose to change our editing software, our documents do not need conversion.

The XML Documentation Process

In order to define an XML application we first have to analyse existing documents and processes. Structure and format are looked at independently, common usage and final deliverables are defined and from all these parameters we can build our application.

A Document Type Definition (DTD) or XML Schema is written which will enforce the structure of our documents. Stylesheets are programmed for providing styling to the editor environment and for output to the required formats and presentation. Profiling can be put in place to provide a mechanism for outputting different versions of the same document with content applicable to specific end users. Customisation of the an XML editor interface using common scripting languages can also be applied to make authoring tasks easier.

Once all set in place, our first instance can be created letting the DTD guide the author through the document as they create it. All they have to do is enter the content. No styling. No formatting. No headers or footers to worry about. No automatic numbering to go astray.

Once the content has been approved and checked we can publish the document using the appropriate stylesheet and have confidence that the output will be to the desired specification.

The process is front-end loaded so a lot of time is spent building the application for the first instance. But even here the time and costs involved are comparable to using traditional methods, but give added benefit because your data can be reused and repurposed. Future instances see a huge time saving over the traditional process.

The neutral Zone

If an XML solution is applied across an entire enterprise we can automate the publication processes with use of a web-enabled interface. Conversion rules and mapping files can be set up so that suppliers and customers who use traditional publication software can convert their content to XML to be stored in a central repository. Data can be tracked, searched and reused.

Case Study: Dungeness B Power Station.

Dungeness B power station is located in Kent, in the south-east corner of England and operated by British Energy (BE). It is a 1200MW AGR which was commissioned in 1966, though did not become fully operational until 1982. Because of this extended time between commissioning and operation, the SOI documentation was in a terrible state, having been authored in different software by thousands of different authors and suffering from major inconsistencies in style and format as well as structure. A requirement was made to recast all SOIs to a common and consistent format, to be delivered in Microsoft Word.

A joint workshop between BE and STATS-NNC was held to define structure and format standards. An XML application was then developed and all legacy documents were converted to XML using an automatic process and then reviewed. Inaccuracies and repetition was removed and Technical Authors provided updates. BE conducted a final review and the documents were then output to Microsoft Word.

Each SOI consisted of a series of individual sub-documents, all having common data and all being referenced in a contents document. Using XML each SOI was authored as a single document so that common information only had to be entered once. It was only at the final publication stage that the document was output as individual sub-documents. This was automated using an XSL stylesheet.

Finally, BE staff were trained to provide any future updates and amendments. At the first training session they stressed that a requirement had been made to globally change the font size of all documentation, that a common logo had to be deleted and a reference code changed

– it had took them one day to apply this to 3 Word documents. It was easy to demonstrate that this could be done a lot more efficiently within XML. A few simple changes to the stylesheet, then republish each document to Word allowed hundreds of documents to be done in a day!

Case Study: Manual vs XML Production of a Sales Tender

Many potential customers express a sense of wonder when they initially see the potential of using XML. However that can soon turn to doubt when it comes to implementing an XML solution to their own documentation. One such case was the Sales Tender Team at NNC – they could see that XML could potentially speed up their tender production but had critical reservations about using it since a very tight time schedule was specified for the delivery for each tender.

The problems they had encountered producing these tender documents in Word 2000 included:

- Inconsistent formatting – the document was authored by 20-30 authors and compiled by a head author. General structure inconsistencies dominated the content.
- The documents needed reviewing. Because of the inconsistencies the document had to be reviewed by an independent Document Control Team which added time and effort and the end result still suffered from minor inconsistencies.
- An excessive time was experienced in printing the documents to a Xerox colour printer. This was essentially down to the misuse of graphics and images within the Word environment. None of the graphics were optimised for output and typical print time for a 500 page document was 10 hours.
- Post print compilation was needed since subcontractor data sheets needed including. This always gave page number anomalies.
- A separate document had to be created containing no financial information for delivery to all subcontractors

To allay their fears it was agreed that STATS would compile and format a traditional Word document alongside developing an XML application and then compiling the Tender document in XML.

For production of the Word document, the authors sent their Word fragments which were then copy and pasted into a master document based on an existing template. The content was then formatted accordingly and delivered back to the customer for assessment and approval.

On the XML side, a document analysis was conducted from which both DTD and stylesheets were developed. As this was a direct comparison we decided that we should copy and paste from the Word fragments to XML rather than use an automated process which was available. As style is not intrinsic to XML no formatting was involved. There was no Document Control review as the stylesheet maintained consistency. By using selective profiling, two individual documents were published from the XML source, one complete and one without the financial information. These were delivered on time as optimized PDFs which took 1 hour to print on their Xerox machine. There was no compilation since the XML linked in all provided data sheets. They were also able to provide the customer with a PDF of the complete document.

The Word document, however, was delivered 5 hours late and still suffered from some minor inconsistencies.

When times for the project were audited it revealed that the XML version, including all development time, was less than the total amount of time taken to produce the Word version. The major time-saving that enabled this was down to not having to style or format the document. Future use of XML would produce much larger savings, plus other areas were identified for extra time savings, such as automating the production of XML from the Word fragments; using a web-based authoring tool so that authors could directly input into XML and creating common XML CV files for direct inclusion.

Conclusion

A summary of the benefits of using an XML solution:

- Precisely match your requirements at no more initial cost
- Single Source Dynamic Content Delivery and Management
- 100% of authors time is spent creating content
- Content is no longer locked into its format
- Reduced hardware and data storage requirements
- Content survives the publishing lifecycle
- Auto-versioning/release management control
- Workflows can be mapped and electronic audit trails made