The impact of XML on library procedures and services

Eric van Herwijnen
January 16, 2000

CERN, Geneva, Switzerland

Contents

Introduction

In recent years, digital technologies such as the World Wide Web have revolutionized the way that libraries are used and organized.

For example, in the High Energy Physics world, the omnipresence of the TeX text processing system and the availability of the Internet have achieved that all preprints are now electronically distributed and retrieved via the Internet. This development took both publishers and libraries by surprise. Now that the distribution of information is being taken care of by the HEP community, scientific publishers are forcing themselves out of this market by continuing to increase their journal prices. The organization of peer review is the only role left for the publisher and even this role could soon be taken away from them. Paul Ginsparg already claimed in 1994 that "certain physics journals currently play NO role whatsoever for physicists. Their primary role seems to be to provide a revenue stream to publishers, a revenue stream invisibly siphoned from overhead on research contracts through library systems..." If there are no more printed journals to store on shelves, the role of the library in the preprint publication process also becomes less clear. Since preprints are stored in full-text form, servers offer very good search capabilities, as good as those of a conventional library. With the Los Alamos preprint server available on every physicist's desktop, I would argue that the flow of scientific information via preprints would not be seriously impacted without libraries.

More traditional, and non HEP-specific library activities have also changed. Acquisition, cataloging, searching and borrowing are done online using automatic library systems with Web interfaces these days. Of course these systems can and are also used to store information on preprints but, in view of the above, we feel that this is redundant. At CERN, the library has gone a step further and has made its system available for the archival of physics experiment's internal documents. One drawback of this database is that the documents are only available in PostScript or PDF and not in full-text form. It is not

impossible, but difficult and costly to recover meta data of the information contained in these documents, which is essential if a specific query from a search engine is to be successful. On the other hand, the current level of granularity (i.e. entire documents in PostScript or PDF form) seems to satisfy most users.

XML (the eXtensible Markup Language, see the XML web site) is a recent technology that has been hailed as the revolutionizer of the Web. Will XML have an impact on the library, and more generally on the flow of information in the HEP community?

XML and its family

The "X" family

XML (see its specification) is the language proposed by the W3 Consortium for describing documents that are to be delivered via the World Wide Web. It has the status of a W3 Consortium Recommendation (since February 1998). Although this is not the same as a full International Standard issued by the ISO, XML appears to be widely accepted today. A large amount of XML software is available (for free) from mainstream vendors such as Microsoft, IBM and others. The intention is that XML will add richer functionality to the Web where HTML is too limited to do so.

XML (see some literature) is a simplified subset of SGML (the "Standard Generalized Markup Language"), and as such, its use largely surpasses documents to be sent over the Internet. XML data is vendor and application independent. XML fits well into any software architecture that aims at re-using its components over a long period of time in a robust manner.

XML originated in 1996 from an initiative organized by some SGML software vendors. At that time, SGML was growing but struggling to become mainstream. The Web, on the other hand, had by that time become fully mainstream. HTML, the language that Web documents are written in, conforms to SGML, but the SGML vendors were almost unanimous in rejecting HTML because of its simplicity, its lack of flexibility and its too practical way of approaching hypertext links. In my opinion it was thanks to these "disadvantages" that HTML became such a succes! Anyhow, the SGML software industry had an impressive amount of products and services ready for the mainstream market: and so XML was born, a simplified subset of SGML designed to overcome the limitations of HTML.

Following XML, a number of other standards were developed, all starting with an "X".

In the past, the SGML community had also produced the HyTime and DSSSL standards, for describing hypertext linking and document formatting respectively. SGML, HyTime and DSSSL, although rigorously written and well thought out, all suffered from the fact that they were far too complex to understand or to implement. The titles of the standards themselves are already meaningless (actually, the "X" family doesn’t score much better on that account). Now the people who made XML tried to learn from the difficulties of SGML, and XML clearly is much simpler to understand and implement than SGML. The HyTime and DSSSL (Document Style Specification and Semantics Language) standards were also re-implemented for XML and became XLINK (XML Link Language for robust hypertext links) /XPOINTER (XML Pointer Language) and XSL (XML Stylesheet Language, the standard for describing the visual representation of XML documents). Documents in XML are, as is the case for SGML, independent of a particular application. This is good, if you want to maximize exchange

possibilities. If, however, you need to standardize on a particular application, you need a way of describing the semantics, which can be done using the XML Schema Language. Other standards exist, and more will undoubtedly follow.

The "X" activity has shown an eXplosive growth. Can this enthusiasm be indicative of a real change with a positive impact for users, not only consultants and software vendors?

Current users and activities

A measure of its success can be seen from the number of industry sectors and projects where XML is being employed. Some examples are:

- E-commerce. The largest number of XML applications can be found in this area.
- Business to business, EDI.
- Healthcare sector. Use of XML to describe health records.
- The IMS (Instructional Management Systems) project meta-data description. Use of XML in the "learning" sector.
- News feeds.
- Media delivery.
- Telephony and speech access to Internet.
- Delivery of XML to cellphones via WAP (Wireless Application Protocol).
- Search engines.
- Integration of CORBA and XML.
- Conversion of QuarkXpress to XML for Internet delivery (e.g. Reader's Digest books on health and cookery).
- At CERN, by the WIRED event display, by LHCb as a persistent way to store the detector geometry in an OO framework, and by ATLAS as a way to store event data.

However, I know of very few applications of XML in the conventional publishing or library sector.

Three global areas for XML applications

Analysing the places where XML is being used, there seem to be three areas of potential use for XML:

1. As the "lingua franca" for documents that are distributed electronically over the Internet, replacing PDF and PostScript today. It is hoped for the library that one day this will be the case. Although I don't believe that instead of writing HTML, we'll be writing XML with XSL stylesheets in the near future (mainly due to the complexity of XSL) it looks likely that there will be a lot more content available via XML. The general feeling is that use of XML will increase the number of documents in full-text form on the Web, and that it will facilitate standard library activities such as classification, cataloguing, searching and retrieval. It could allow a library's focus to change from conventional paper documents to electronic documents, thus allowing the library to become digital.
2. As a general purpose exchange language by computer systems that need to input and output data, for example a automatic library system, or a medical apparatus. The fact that XML data is in ASCII and that XML data is conveniently identified and delimited via tags, makes XML very suitable for any system that needs to store intermediate or permanent data. I am thinking of bibliographic records in the case of a library, or health records in the case of a medical application.

3. As a language for the description of data (in the case of CERN, by physics experiments). As a replacement of, or as a complement to, a database. Here I am thinking of certain types of event data (coordinates of hits and tracks) or the description of the geometry of a detector (coordinates of shapes and volumes, types of material). One can also imagine that information from the Particle Data Group (particlenames, branching ratios, lifetimes, masses, quantumnumbers) would be really useful if it could be accessed by Monte Carlo and analysis programs in XML.

In the remainder of this article, I will discuss the impact of XML data coming from the three areas mentioned above.

XML as the lingua franca for electronic document delivery

Advantages of XML for libraries

Despite its great potential, there are only a few places on the Web where the impact of XML on libraries is discussed. There are even less concrete examples of XML projects. XML is currently being used for literary textual documents with a fairly simple structure; it is being used instead of HTML because people can define their own tags. Some points in favour of XML (for libraries) are:

- The structure and markup of an XML document facilitates the creation of document databases, while at the same time XML content can be delivered on the Internet or on a CD-Rom.
- The "meta-data", or data about the document, can be explicitly read from the XML tags, provided that libraries can agree on a standard set of tags. For books and journals, the meta-data is captured in the form of a bibliographic "MARC" record, that has to be created by hand and is added to the library database. XML would thus obviate the need for MARC records.
- There are some standard DTDs (Document Type Definition, an inheritance from SGML meaning the collection of tags of a particular XML application) that were designed specifically for literary scholarly documents (the TEI, Text Encoding Initiative) and scientific articles and books (ISO 12083). These applications can be used with XML. This is also an SGML MARC DTD.

Some people believe that HTML will disappear and that the main impact of XML on a library is the choice of a standard DTD such as the TEI, or in the case of HEP, ISO 12083. I will argue in what follows that this view is highly optimistic. XML is clearly a step in the right direction, but it requires further simplification before it can be applied to the organization of documentation.

I should point out that ever since the beginning of the Web, there has been a misunderstanding of what HTML is about. The SGML community never understood that the use of HTML was a pure accident. HTML is a display language; any ASCII system could have replaced it, e.g. the 20 most common TeX commands, nroff/troff or even GML would have been adequate. A document’s structure is irrelevant from the point of view of displaying it in an Internet browser. However, the fact that it is so simple, and so small, meant that it was easy to implement fast browsers. I don’t think that the Web people consciously choose HTML because it was an SGML application (the majority of us didn’t understand what an SGML application was in 1989). The HTML DTD was written a posteriori, under pressure from the SGML vendors. And so there was a lobby in favour of displaying SGML - an SGML browser is a lot more complicated to make, and it is usually sold for big $$$’s. But HTML is just fine for online display in most cases.

Of course one can always think of ways to improve HTML, or reasons why some special effect is desirable. In the early days of the Web, 1994 to be precise, Java was introduced as the way to enrich HTML. Applets bouncing around all over the place, JavaScript and cookies to personalize your content. However, the number of incompatible browsers also rapidly increased, as well as the number of Internet users that can reduce the available bandwidth to a few hundred bp/s. Adding intelligence to the browser has to be done very carefully. My experience with running a large scientific website has shown that in the long run users only care about:

1. Reliability (=> don't use scripts or applets as they are impossible to make work for all browsers).
2. Speed (=> reduce images to the absolute minimum)
3. Information (=> be very careful with stylesheets)

The requirement for formatting comes at the very end of the list. So in my view, if there is a use for XML for documentation, it will initially have to be as a hidden format: HTML will remain the format for Internet delivery for some time.

What's stopping us today from using XML instead of PDF and/or PostScript?

As I have observed in the above, the HEP world likes to use PDF and/or PostScript as the format for distributing physics documents. The touchstone for a successful introduction of XML would be if we could replace PDF and/or PostScript by XML for storing and displaying HEP documents. The following major obstacles prevent us from doing this:

1. The difficulty of creating XML. There are quite a number of XML editors in the public domain, but no one is going to write a document with an XML editor, unless it has a really good formatting support, possibly through the use of XSL stylesheets. Even if a good HTML editor such as FrontPage, DreamWeaver or PageMill were to support XML formatting, I can't see people using such a tool for creating (printable) documents. On the other hand, support for XML with XSL stylesheets is not yet available in the common word processors. Experience with the formatting of SGML documents has shown that the typesetting of XML documents will require very complex and probably expensive software. And if support were available in common word processors, it would require a lot of effort and specialized expertise to configure these programs, particularly for non standard applications (DTDs).
2. Lack of browser support for rendering of XML documents. It is claimed that both Netscape and the Internet Explorer will one day understand XSL and be able to render XML documents. However, experience with the partial support for CSS, which is less complex than XSL has shown that this will not be as easy as it may sound on paper. So this support may take a little while in coming.
3. In a similar vein, but of particular interest for the scientific community is the lack of support for mathematical formulas. An XML application for math exists (Mathematical Markup Language, MML). I am not yet aware of stylesheets for MML. Representation of such a math style sheet would impose heavy constraints on browsers. Also, experience with math markup languages in SGML has shown that there is a large resistance from scientists to use structured document editing tools for mathematical equations and symbols in their articles. TeX is very firmly entrenched whenever mathematical typesetting is required. Any editor should be as intuitive as TeX, and any math supporting browser would have to include TeX to display the formula's as nicely as TeX. This has been done for SGML, but the result is very heavy, as TeX is based on an old fashioned, non interactive architecture. The result is 'clunky'. Despite TeX's disadvantages, the physics

community can not agree to standardize on more modern alternatives such as Framemaker or Word, because they lack TeX's high quality mathematical typesetting. This need has been underestimated by the developers of MML, who believe that making a formula reusable and searchable is more important than formatting it. LaTex2HTML still has a long life ahead of itself.

These obstacles are not new, they were the same 15 years ago for the adoption of SGML. This time, there seems to be more momentum behind XML. Will the software industry find a sufficient commercial drive to pull this challenge off?

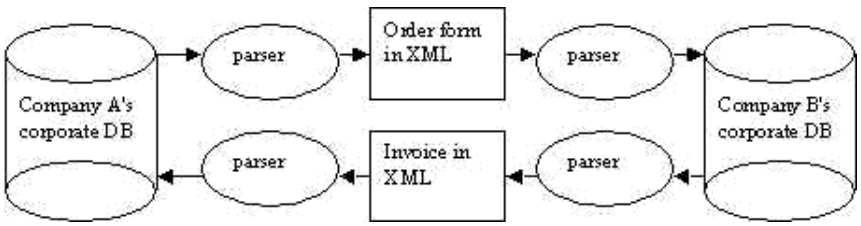
Provided that it will be sufficiently easy to create XML, both the library and its users will gain a significant advantage by using it, because XML documents will be fully searchable, and its meta data will be automatically extractable. Thus search engines will be able to profit from this by better responding to user's queries.

Even the XML Linking Language requires the endpoint of a link to be persistent, although it has some facilities for generating collections of links. Use of XML might make the results of a search engine more reliable, but this would not help for keeping it up to date. Use of the XML Linking Language would help to make hyperlinks inside static documents more robust. In any case broken links inside a document tend to be less of a problem as authors are fairly conscientious and Web authoring tools allow quick identification of broken links on the level of an entire Web site. However, the XML Linking Language will not solve the (more annoying) problem with search engines pointing to documents that no longer exist - because the search engine's database is not in sync with the status of the real data.

A tool such as FrameMaker, which already has full support for SGML, will undoubtedly soon have good support for XML, i.e. it will be as easy to extract XML from Frame as it is now to make PDF (or HTML from Word). As there is growing support using FrameMaker for complex technical documents at CERN (excluding mathematics), one can imagine that a lot of technical design reports for the LHC experiments will be available to the library in XML. As these documents are a priori not sent to a preprint server, the library could have an important role to play in this development. But for documents containing mathematical formulas TeX, PostScript and PDF will remain the norm.

XML as a general purpose exchange language

It is much simpler to use XML as a general purpose exchange language, for documents that only have to be understood by a system. Highly structured data such as purchase requests, orders, tax forms etc. can be coded using XML and processed by different systems. Using XML is only advantageous if the two systems are really different, such as the databases of two companies participating in an EDI (Electronic Document Interchange) exchange:



The interchange medium need not necessarily be the Internet. By standardizing on XML, the number of conversion programs (symbolized by the blob containing "parser"), is linear in the number of different

formats used by each respective company, instead of squared. By using XML as the exchange language, the exchanging partners become independent of each other, provided they can agree on a standard "Document Type Definition" (DTD, = a specific collection of XML tags). No wonder that the most commonly found applications for XML are in the domain of e-commerce and business-to-business applications.

But it's easy to get carried away by the hype. Surprisingly enough, XML is sometimes used as a layer inbetween a database and an HTML Web interface. Data is extracted from the database into XML, then converted into HTML. If one of the parties is a client on the Internet, and the data input is to be done from a Web browser, it's much simpler to use HTML forms. In this case, it seems to me that introducing XML is an extra, unnecessary complication.

In fact, one of the most often quoted applications of XSL is to convert documents from XML into HTML. Unless this is used as a temporary measure, awaiting the day when full XML support will be available in the common Web browsers, this use of XML seems to introduce an extra layer of complication.

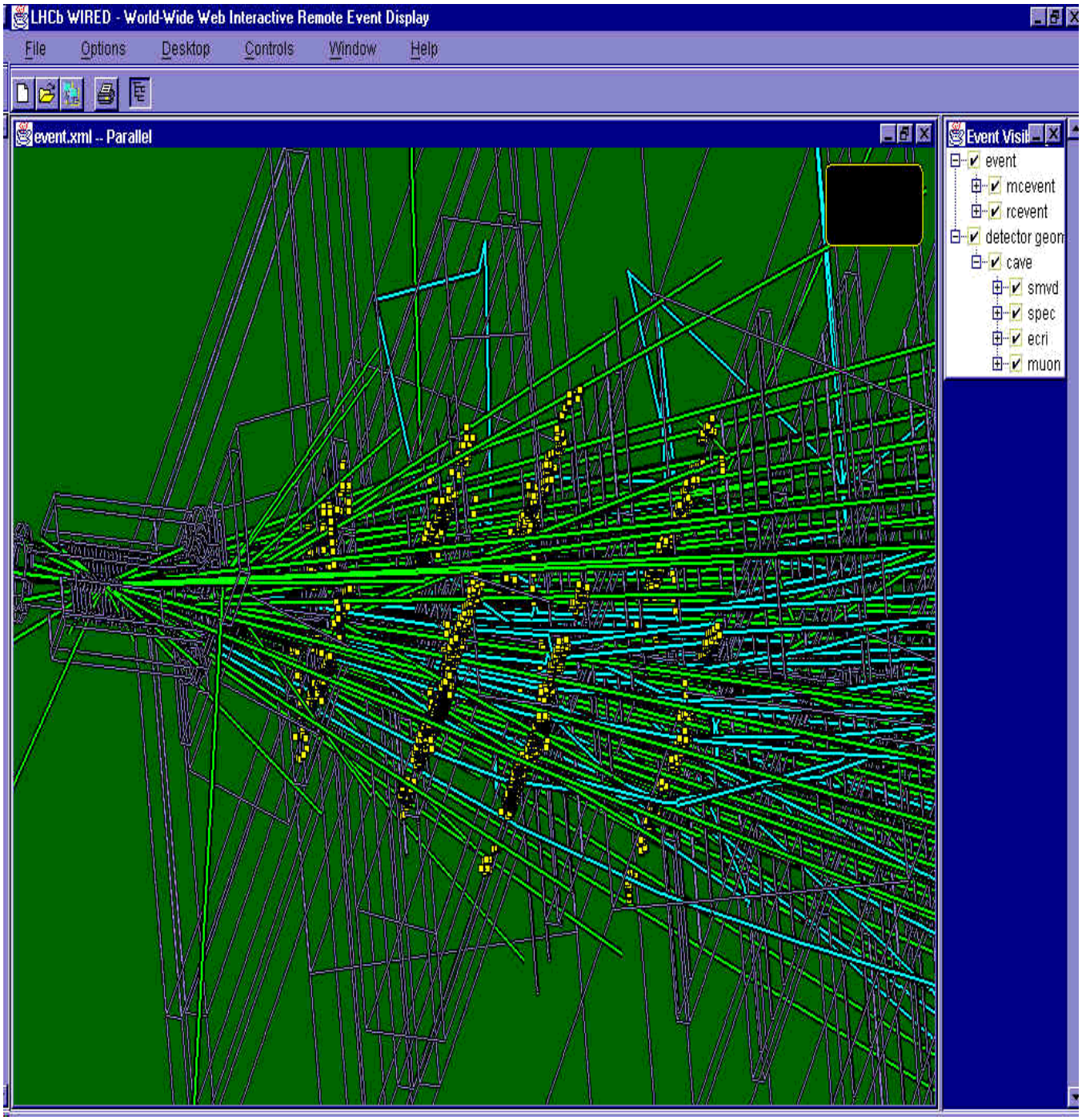
XML as a language for describing data

The simplest application of XML is to non textual, but numerical data (N.B. not binary data!). HEP is full of this, and it is no wonder that there are now many examples of physicists at CERN who are turning to XML as a general purpose data description language.

Some examples are described in the following.

WIRED

WIRED is a system for displaying events (i.e. particle tracks, hits) from a collision inside a detector. The data on the display can be manipulated in many ways: one can zoom in and out, look at specialized projections, hide parts from the display and so on. In the figure below you can see hits on wires in a tracking chamber with a few reconstructed tracks.



Tracks generated by a Monte Carlo simulation program can be distinguished from reconstructed tracks by changing their colour or representation. WIRED is written in Java. To get the event and detector data (i.e. the coordinates of points, lines and polygons) into WIRED, WIRED has to be integrated into an analysis program (which then needs to be written in Java), or the data should be supplied in an application independent form such as XML. An XML loader has to be written that knows how to represent a <polygon> on the screen.

Traditionally, experiments have built their own event displays. The big advantage of this way of

working is that the WIRED developers can concentrate and apply their expertise to the very specialized domain of event visualization, while the problem of getting the data into the system is handled once, by XML. The WIRED client, an experiment, only has to write its data out in XML form (possibly via a Web server). In principle any experiment can do this.

Detector Description Persistency in Gaudi

Within the Gaudi framework, the LHCb experiment is using XML as a persistent way to store its detector description. The detector material and 3D geometry C++ objects (following the Geant 4 structure) are converted to XML for saving in a transient data store, to facilitate independent editing and creation by graphic XML tools. The following mapping between C++ objects and XML was used:

C++	**XML**
Class	Element
data members	Attributes
Composition	Element embedded in content model
Reference	Element reference with hyperlink and class ID of a target data object embedded in content model

The advantage of using XML is that the detector geometry can be easily modified (e.g. adding a new subdetector) using an XML editor; and it can be easily mapped in both directions onto programmatic datastructures. A drawback is that there is no equivalence of neither methods nor inheritance in XML. This means that if a new C++ subclass is added, the XML DTD needs to be modified, and it is not easy to automate this.

The impact of these developments on library procedures will be indirect, as it will enhance people’s awareness about XML, and reduce the learning curve for using XML for textual documents.

Conclusions

XML is a very interesting technology that is being applied in a number of places to describe data in an application-independent way. Applications of XML to documents will bring clear benefits to the library, but in the absence of easy tools to save documents as XML with a corresponding stylesheet, people will continue to use PDF and PostScript for distributing formatted documents on the Web. For mathematical texts, despite the existence of MML, it may take even longer.

Notes

1. The development of the W3 Consortium’s "Recommendations" is an important and laudable activity. However, we should not forget that, although the process by which a "Recommendation" is made would seem democratic enough (by virtue of the use of the Internet), only paying members of the W3 Consortium are allowed to vote. For International Standards, the procedure seems more democratic (although ISO gets its funding through the sale of printed standards - a procedure that does not always enhance their use).