

XML Basics for SAS Programmers

Yong Li, Sanofi-Aventis, Malvern, PA

ABSTRACT

XML is a simple dialect of SGML(Standard Generalized Markup Language). Originally designed to meet the challenges of large-scale electronic publishing, XML has evolved to play an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere. CDISC (Clinical Data Interchange Standards Consortium) has developed several XML-based models for the electronic acquisition, exchange, submission, and archival of clinical trials data. Using XML as the underlying technology for data formatting fits the FDA's strategic direction for electronic submissions. However, XML has not been widely presented at recent SAS user conferences. In this paper, we will describe XML and its basic structure, examine differences between XML and its cousin HTML, and evaluate potential advantages for XML-based submissions compared to the current XPT format. We will also consider other potential uses of XML and review new SAS tools available for reading and writing XML.

INTRODUCTION

Since the release of the first version of Extensible Markup Language (XML) by the World Wide Web Consortium (W3C) in 1998, XML has gained acceptance in many industries as an integrating platform for applications and data sources. As a result, momentum is growing to leverage XML technology in the content management and publishing systems used by the pharmaceutical industry and government agencies. In July 2003, the FDA announced that the Study Data Tabulation Model (SDTM) developed by CDISC would be the standard format for sponsors of human drug clinical trials when submitting data to the FDA. Several other models, such as ODM (Operational Data Model), LAB (Laboratory Data Model), SDS (Submission Data Model), and ADaM (Analysis Dataset Model) which are under development by CDISC are all XML based. ODM addresses data used during trials and analyses, LAB describes ECG and other laboratory data standards, SDS emphasizes data flow from database to regulatory submission, and ADaM defines safety domains and efficacy variables to facilitate statistical analysis. Several XML schemas for these models have been released. For example, Study Data Tabulation Model, Version 1.1 (Draft) was released in January and Case Report Tabulation Data Definition Specification (define.xml), Version 1.0 was just released in February. Furthermore, at the FDA's request, a CDISC-sponsored group has developed an XML-based replacement for the SAS version 5 transport files that are currently used to send case report tabulations from clinical trials to the FDA. Although the FDA has not stopped accepting transport files, it has explicitly stated that XML is one of the new formats accepted in its guidance for electronic submission. As XML vocabularies for clinical data are created, schemas are defined, and tools for creating and parsing XML documents are developed, the FDA as well as other global regulatory agencies will ultimately require submissions in XML. However, XML has not been widely presented at SAS user conferences, including SUGI and PharmaSUG. Moreover, the basics about XML were largely ignored in most recent presentations. It's time for SAS users to pick up some knowledge of XML and learn about XML programming. In this paper, we will describe XML and its basic structure, examine differences between XML and the more familiar form of HTML, and evaluate potential advantages for XML-based submissions compared to the current XPT format. We will also consider other potential uses of XML and review new SAS tools available for reading and writing XML.

WHAT IS XML

XML is a simple dialect of Standard Generalized Markup Language (SGML). SGML was formally approved as a standard for document structure (or really a description of how you can specify a document markup language or tag set) in 1986. SGML has been used by publishers, technical writers and library automation personnel to create catalogs and technical publications, and has been in use at IBM for several years in the production of their technical publications. However, as a markup language, SGML was never widely used due to its complexity. Hypertext Markup Language (HTML), which is a derivative of SGML, on the other hand, has been widely adopted by the Internet due to its simplicity. But an inadequate number of predefined HTML markup tags have limited the possibility of large-scale electronic publishing. To cope with this matter, XML was developed. Although XML was originally designed to meet the challenges of more complicated electronic publishing, it has evolved to play an increasingly extensive role in the exchange of a wide variety of data on the Web and elsewhere.

THE BASIC RULES OF XML

XML is actually a set of rules. Its syntax is really simple. Figure 1 shows an example of XML, and it illustrates most of the rules of an XML document.

```
<?xml version="1.0" encoding="windows-1252" ?>
<TABLE>
  <DEMOG>
    <SUBJID> 19581001 </SUBJID>
    <AGE> 58 </AGE>
    <SEX> Male </SEX>
    <RACE> Caucasian </RACE>
    <BIRTHDA> 1945-05-08 </BIRTHDA>
  </DEMOG>
  <DEMOG>
    <SUBJID> 19581002 </SUBJID>
    <AGE> 37 </AGE>
    <SEX> Female </SEX>
    <RACE> Black </RACE>
    <BIRTHDA> 1965-12-18 </BIRTHDA>
  </DEMOG>
  <DEMOG>
    <SUBJID> 19581004 </SUBJID>
    <AGE> 59 </AGE>
    <SEX> Male </SEX>
    <RACE> Other </RACE>
    <BIRTHDA> 1944-07-22 </BIRTHDA>
  </DEMOG>
</TABLE>
```

Figure 1. A simple example of XML

Each individual markup code is referred to as an element, but many people also refer to it as a markup tag. Each markup tag is embraced in “<” and “>” symbols. The ending tag of an element has to match the beginning tag, such as </AGE> to <AGE>, </DEMOG> to <DEMOG>, and </TABLE> to <TABLE>. Tag is case sensitive, i.e. <AGE> is not equal to <Age> or <age>.

All tags must be fully nested without over lapping, i.e.

```
<TABLE>
  <DEMOG>
    <SEX>Female</SEX>
  </DEMOG>
</TABLE>
```

But not:

```
<TABLE>
  <DEMOG>
    <SEX>Female</DEMOG>
  </SEX>
</TABLE>
```

An XML document must have a root tag, for example, <TABLE>, in this case. An empty tag, that is one without data, can be abbreviated as <, tag name, and />. For example, if race is missing, then <RACE> </RACE> can be written as <RACE/>.

Furthermore, a tag may include a value and/or attributes. As an example, we could add an attribute to the element of <BIRTHDA> to indicate its format as <BIRTHDA FORMAT="International">. Attributes are quoted in XML and are used to convey information that describes the elements.

In addition, <, >, &, ‘, and “ are the reserved characters for XML. They can be created by using <, >, &, ', and ", respectively. Comments are enclosed by <!-- and -->, and process instructions by <?xml and ?>.

XML AND HTML

Both HTML and XML are markup languages derived from SGML. HTML is a sub set of SGML code inserted in a file intended for display on a Web browser page. These tags tell the Web browser how to display a Web page's words, tables, and images for the user. But we don't always have to use markup tags to control the way the document is presented. In the case of XML, markup tags are used to describe the actual contents by telling us what it is and how it can be used. Since XML tags say what the content is, not what it looks like, this facilitates more precise declarations of content and more meaningful searching.

A predefined number of tag sets are available in HTML to conform the presentation of information on the web. For example, <P> is for paragraph and for bold text. In XML, on the other hand, we are not restricted by the number of tags that someone else has thought of, or defined. Users can create their own markup tags to describe the data: it can be any thing that you like, but of course we want the markup tag to be as descriptive as possible about its content. In the above example, male and female are marked up by <SEX>, but we could use <GENDER>, or any other name instead. In this context, XML is called extensible.

HTML was designed primarily for displaying information on the web with both the content and formatting of the data tied together in the HTML tags. XML was designed to separate the data from the presentation; it contains only the data. This separation of content and format enables XML to take advantage of various presentation forms. For example, you can alter the presentation of information by changing the style without the need to rerun programs.

In addition, rules in XML have to be closely followed. In HTML, if you do not close a tag properly, the browser can still display it properly. This is not the case in XML. Every element in XML must be closed correctly. Also tags may be nested (although not encouraged) in HTML, such as <P>text</P>, but it is not acceptable in XML. XML elements can have attributes like HTML, however the attribute value in XML must always be quoted.

XML AND XPT

The SAS version 5 transport file (XPT) format that is currently used to send case report tabulations from clinical trials to the FDA was released in the mid-1980s. Since then, technology in database structures and clinical database management systems, as well as data transport and exchange has advanced dramatically. Version 5 transport file formats, however, have not changed since then. The obvious limitations of version 5 transport files still exist: 8 characters for variable names, 40 characters for variable labels, and 200 characters for character data. Other limitations, such as the representation of dates and times and no standard international character sets and SAS specific metadata, have become noticeable and troublesome. A SAS version 5 XPT file also does not allow for extensibility. If you want to change the size of a variable name, you have to revise all your programs that produce the XPT file. Furthermore, formats are saved as a separate file with a SAS version 5 XPT file. With XML, it is possible to wrap data and formats together. XML has a tree like structure. This enables XML to construct hierarchical data, whereas the SAS version 5 XPORT Engine generates a flat XPT file. The hierarchical structure of XML and its embedded metadata allow data to be exchanged at any level. Our XML example in figure 1 was generated from a rectangular SAS file, it contains only two levels: table and demographic information. If we build a XML file for a family pedigree, then we could have many levels depending on how many generations we want to trace back. If you are only interested in your direct family information, you can just retrieve them from the branch that contains your immediate family data and ignore the rest of the tree. The FDA recognized these advantages of XML over XPT format and encouraged CDISC to develop an XML-based replacement for the SAS version 5 transport files, although no timetable was given.

XML AND THE PHARMACEUTICAL INDUSTRY

XML was originally intended for humans to read. Because of the simplicity of XML, computers can also access it, understand it and translate it into other useful information. Apart from taking Web contents to your cell phone and PDA, XML can also be used to carry streaming data and make your Web dynamic and real-time. From the clinical point of view, XML can be used in Electronic Data Capture (EDC) systems having data captured in electronic form, via the web or a handheld device. XML can also be used in electronic systems for patient diaries, where patients can periodically record their condition via Web, handheld, or interactive voice response (IVRS) systems. This data can be then transmitted electronically to doctor's offices or Clinical Data Management (CDM) systems. While the data collected for any given clinical trial may vary, there is a significant amount of commonality. The XML standards mentioned above, under the development by CDISC, are initiatives to standardize electronic capture, exchange, submission, and archival of clinical data. XML is free, non proprietary, and globally used. XML-based global initiatives, such as the Electronic Common Technical Document (eCTD) and Product Information Management (PIM) is just a couple of examples. The eCTD guidance defines the electronic submission delivery structure and

acceptance criteria for electronic submissions according to the international CTD format. The PIM is developed to leverage the power of XML to reduce the amount of time required by a pharmaceutical company to prepare a submission and to speed the review process by European regulatory agencies. These new standards will drive the need for XML programming.

XML AND SAS

EDC and CDM vendors, data warehousing and data mining providers, as well as statistical analysis products and solutions, such as SAS, are readily adopting new XML standards in order to enable data exchange and interoperability. Since an XML document is just a plain text file, a SAS programmer could use the familiar SAS data step to create an XML file. But for most SAS programmers who are not usually familiar with XML syntax, SAS has done an excellent job in developing tools to read and write XML documents. Both the Output Delivery System (ODS) and SAS XML Libname Engine (SXLE) can be used to write SAS data to XML, although the former cannot be used to import XML files back to SAS datasets. The syntax of the SXLE for generating XML files is similar to that of the SAS XPORT Engine.

```
libname xmlout xml 'c:\xml\demog.xml';
proc copy in=work out=xmlout;
  select demog;
run;
```

Running the above code on SAS 8.2 will produce the XML file listed in Figure 1. Another way to create an XML file from SAS data is to use the ODS markup tagset.

```
ods listing close;
ods markup file = 'c:\xml\demogbymkup.xml' tagset=default;
  proc print data=demog;
  run;
ods markup close;
ods listing;
run;
```

The above XML file can also be generated by using ODS with XML as an output destination.

```
ods listing close;
ods xml file = 'c:\xml\demogbyods.xml';
  proc print data=demog;
  run;
ods xml close;
ods listing;
run;
```

You can specify only one output file using XML as an output destination in ODS. While using ODS markup, you can create multiple output files with the same ODS statement by specifying different tagsets.

In all the above examples, SAS formats and variable labels are lost when translated to XML files. If you want to preserve SAS formats and labels, you can specify `xmltype` and `xmlschema` in the SXLE, as in the following example.

```
libname xmlout xml 'c:\xml\oimdbm.xml' xmltype=oimdbm xmlschema=yes;
proc copy in=work out=xmlout;
  select cusfmt;
run;
```

To import the `demog.xml` file generated above back to SAS data set, use the following code:

```
libname xmlin xml 'c:\xml\demog.xml';
data demog;
  set xmlin.demog;
run;
```

In this example, we use the SXLE to import a simple XML file back to a SAS dataset. XML can be much more complicated with many levels and elements which may contain further values and attributes. In fact, SAS datasets are neat rows and columns of data whereas XML documents are hierarchically arranged representations. To import

more complicated hierarchical XML files to flat SAS data, you need to specify the SAS XMLMAP option of the SXLE available in SAS version 8.2. XMLMAP is a special XML document, which contains information that the SXLE uses to parse the XML document. SAS versions 9 and up can build an XMLMAP syntax by a drag-n-drop approach through a user graphic interface. There are several excellent papers covering SAS XMLMAP and you can also consult XMLMAP documentation on the SAS Web site. You may still wonder how XML can be used natively and how to format XML. Just as HTML, we can use style sheets to format XML. The language that is created specifically to format XML files for display is the eXtensible Stylesheet Language (XSL). There are many Web sites on the Internet that introduce XSL, users can consult the Web if interested.

CONCLUSION

XML will clearly play an increasingly important role in clinical information management solutions throughout the pharmaceutical industry. Many XML-based initiatives are under way to formulate models and define standards to unify and speed up the practices. To maximize the profit potential, pharmaceutical information needs to be acquired and exchanged in a timely fashion. XML not only provides a platform for data exchange, it also enables electronic data capture and process automation. As companies recognize the business benefits derived from XML solutions, SAS users should prepare themselves for XML programming.

REFERENCES

Guidance for Industry: Providing Regulatory Submissions in Electronic Format -General Considerations
<http://www.cfsan.fda.gov/~dms/opaegui3.html>

Electronic Common Technical Document (eCTD)
<http://www.fda.gov/cder/regulatory/ersr/ectd.htm>

Operational Data Model, Final Version 1.2.1. ODM Final Version 1.2.1
<http://www.cdisc.org/models/odm/v1.2.1/index.html>

Case Report Tabulation Data Definition Specification (CRT-DDS, also called define.xml) Review Version 1.0.
<http://www.cdisc.org/models/def/v1.0/index.html>

An FDA-Requested XML Replacement for SAS Version 5 Transport Files in U.S. Regulatory Submissions.
<http://www.lexjansen.com/pharmasug/2004/FDACompliance/FC08.pdf>

XML and SAS®: An Advanced Tutorial. Greg B Nelson
<http://www2.sas.com/proceedings/sugi25/25/aa/25p013.pdf>

XML Primer for SAS® Programmers. Jack N Shoemaker and Greg B Nelson
<http://www2.sas.com/proceedings/sugi28/193-28.pdf>

<XML> at SAS® - A More Capable XML Libname Engine. Anthony Friebe
<http://www2.sas.com/proceedings/sugi27/p179-27.pdf>

XML? We do that! Anthony Friebe
<http://www2.sas.com/proceedings/sugi28/173-28.pdf>

CONTACT INFORMATION

Yong G. Li
Sanofi-Aventis
9 Great Valley Parkway
Malvern, PA 19355
(610) 889-6330
(610) 889-6932
yong.li@sanofi-aventis.com