Paper SAS3

Defining and Validating CDISC Data Standards to XML in SAS® Technology

Anthony Friebel, SAS Institute, Cary, NC
Edward Helton, SAS Institute, Cary, NC
Eugenia Bastos, SAS Institute, Cary, NC
Michael Kilhullen, SAS Institute, Cary, NC
Jane Boone(CDISC)

ABSTRACT

We wish to demonstrate the Base SAS technology and tools to use and report investigational data in XML according to the CDISC Operational Data Model (ODM) v1.2 schema in Base SAS. The Operational Data Model (ODM) is a vendor neutral, platform independent format for interchange and archive of data collected in clinical trials. The model represents study metadata, data and administrative data associated with a clinical trial. Program fragments used for forward and backward conversions of data from SAS to ODM v1.2 format will be presented, along with displays of the data before and after conversion. The implications of data transmission using this standard will be discussed, including how to display data in ODM v1.2 format, how to validate imported and exported data, and how to otherwise assess conformance of the data with the model. Special attention will be paid to the implementation of the SDM in regulatory submissions to carry out the process in Base SAS and to demonstrate those processes for the support of CDISC Standards in the most common analysis and reporting software in the industry. Validation of these processes are critical to compliance. Standard data sets of both efficacy and safety from Phase III trials will be used to demonstrate the conversion of content to format has been performed in compliant manner.

INTRODUCTION

The new FDA draft guidance entitled “Providing Regulatory Submissions in Electronic Format --- General Considerations” begins the industry discussion and evaluation of using XML in lieu of SAS XPORT transport format for future electronic submissions. The transition to the international or global use of XML for all electronic submissions began with e-CTD, HL7 and now includes the CDISC Data Standards.

In anticipation of this transition to XML, SAS R&D has enhanced the SAS XML Libname engine and developed PROC CDISC in SAS 9. Both of these new tools convert SAS data set content into the XML format specified by the CDISC Operation Data Model. The new PROC CDISC will also be “back ported” to SAS 8.2. At the CDISC Interchange meeting in Washington DC the week of Sept 29th, 2003 these tools were demonstrated with considerable interest. Since that time, SAS Foundation Technology has been very busy in developing other tools to assist the implementation and use of CDISC Standards in SAS. They include a new XML ODM Viewer, XML Libname engine and XMLMap extensions, and the XML Mapper.

CDISC is anticipating that the V 3.1 of SDM (SDS) will be the first data standard adopted for use by the FDA. The development of the Standard Data Tabulation Model has also been very supportive of this requirement for implementation as well by the Agency. The appearance of SDS in FDA Guidance will occur with approval of SDS by the HL7 (ANSI Accredited Standards Organization), which is responsible for standardizing language and terminology for use in Health Care. SDS V3.0 was previously approved by HL7 but due to the FDA Pilot of V3.0, changes to include domains of efficacy were added in V3.1.

The most significant changes since the prior version now in V3.1 are summarized below:

- Corrections and amendments to what was previously known in V3 as the General Study Information Model to improve consistency, including the incorporation of new variables
- Incorporation of a new “Trial Design” component to the SDTM
- A more thorough solution for defining relationships between datasets, between records in different domains, and between supplemental qualifiers and a parent domain
- Representation of all date/time variables in ISO8601 format, and the elimination of the concept of a separate Date/Time Precision variable for each date/time variable
- New domain variables to represent additional timing descriptions, flags, and descriptive attributes of an observation (e.g., --SCAT, --DOSRGM, --NRIND)
- Removal of some variables within domains (e.g., --INTP, --DESC, --BLRESC, --BLRESN) that were either deprecated in the prior version or were inconsistent with the intent of the model
- Numerous changes to variables, labels, formats, and notes to reduce ambiguity and improve consistency.

The HL7 balloting process for approval of Version 3.1 begins at the first of April 2004 and it is hoped after approval by HL7 it will soon appear in the FDA Guidance in May or June 2004. It is believed that SDS V3.1 will appear as a specification and is presently described by CDISC as follows: *Plan is to reference as specification (HL7-approved CDISC SDS V3.1) in Guidance for FDA Implementation of International Conference on Harmonization (ICH) eCommon Technical Document (eCTD).*

CDISC and SAS are working in partnership to demonstrate how to load the data standards into common industry software such as Base SAS and how to use the standards both for data warehousing or metadata management and analysis, reporting and submission to the FDA or other regulatory authorities. Similarly, HL7 and CDISC are working in concert to establish interoperability between the HL7 healthcare standard and the CDISC therapeutic product development standards. Again SAS is very interested in the interoperability modeling processes needed to make this occur.

METADATA MANAGEMENT - CDISC SDS V3 IN ANALYSIS/REPORTING AND BASICS PRINCIPLES OF CDISC SDS

The Submission Data Standards (SDS) model is developed to handle data for subjects who participate in clinical trials. At a high level, it allows us to organize data by domain, and apply standard business rules to transform operational data into the SDS. These rules and domains are well documented and available at WWW.CDISC.ORG. What is appealing about the model- regardless of whether you are focused on submissions or supporting other data consumers in your organization- is that it provides an approach to standardizing data that is not drastically different that what you may be using today.

MODEL MANAGEMENT TECHNIQUES

So then, we have a model that we want to implement in our systems. But there is a bit more to consider before you start rolling it out in your systems. How will we manage the model? That is, how can we store the information about the model in a way that can be easily integrated into our existing SAS processes? Whether you are a batch SAS programming shop or using sophisticated ETL tools, the most basic approach is to generate and store metadata about the model. In the SAS world, this is equivalent to storing the output generated by PROC CONTENTS, or even storing empty dataset shells. However, some organizations prefer to use data modeling tools. Data modeling tools offer a more structured and formal way of managing your model. Not only does each domain exist as an entity, but the relationship it has to other domains or data structures can also be managed. When the model is ready to be deployed, many packages create export files structures called data definition language (DDL) exports. This is basically metadata about the data tables.

How you choose to handle model management is up to you. Base SAS approaches get the job done. But modeling tools do a great job at encapsulating all information about the model. However, using modeling tools can be an overly complex task and often generates information that is not valuable to the SAS world.

POPULATING YOUR MODEL WITH ETL STUDIO

SAS ETL Studio is a visual design tool that helps organizations build, implement and manage ETL processes from source to destination, regardless of data sources or platforms. Users can perform in-depth data transformations with minimal programming to quickly meet enterprise data integration requirements and support business and analytic intelligence. ETL Studio is a full Java Client that builds upon a process-driven approach to building your model. This means less steps/clicks for the end user to perform the tasks typically associated with the ETL process. Moreover, ETL Studio supports multiple, concurrent users providing change management functionality (check-in\check-out) across your programming teams, and stores metadata in accordance with the Common Warehouse Metamodel (CWM).

REGISTER THE MODEL

SAS ETL Studio allows us to create a hierarchy of repositories that can depend on each other. At the highest level is our Foundation Repository. Here, we store information about our users, generic library assignments, standard data tables, etc. In our SDS implementation, we will create a standard library reference in the foundation repository to where we will import the data model from whichever format it is stored in. As we define other repositories- for example, drug, project, or therapeutic area- we ensure that we make the new repositories dependent on the foundation. By doing so, the CDISC data models are always available centrally. Moreover, if an administrator changes the model, the new information is immediately available to all repositories.

DEPLOYING THE MODEL

The process of populating the SDS model is to first create your target data tables. This is easily done using SAS ETL Studio's Target Designer. The Target Designer uses a wizard interface that allows us to specify all information about the data we are creating. By referencing the CDISC library, we are able to create physical tables based on the model.

REGISTER THE SOURCE DATA

In order to populate the target tables, we must also register the source data. Once again, SAS ETL Studio provides a wizard interface to do this. In this case, we use the Source Designer. Like the Target Designer, the Source Designer allows us to specify all information about the source data from one common interface. Moreover, the source data identified can be bulk registered allowing all data in the source area to become immediately available for use in one simple step.

MAPPING DATA USING ETL TOOLS

SAS ETL Studio provides interactive tools to build the processes that will map and load your SDS data. A number of predefined process templates are available for use. In addition, new processes can be easily defined using the Transformation Generator, a wizard based interface to turn standard SAS code into compliant process plugins. Mapping data is where the focus of your work will be. Here, we determine which data from the source data gets loaded into certain variables in the target data, how new data is created, and how we transform existing data into a new form. Once again, SAS ETL Studio provides an interactive interface for implementing your mapping rules. When all is done, the processes are submitted to populate the SDS data.

WE CREATED THE SDS DATA, NOW WHAT?

Most companies have elaborate macro libraries or software packages in place to analyze and report on the data, publish the output, or even post the information to internal websites. While SAS has many new tools that help in this process, what's important to mention here is that you have taken the first step in dramatically improving the efficiency and reusability of your submission process. Data from trial to trial will be stored in a common format, allowing any analysis or report built upon that model to be reused across your organization. Moreover, you've improved your ability to interact with other models being defined. The remainder of this paper will discuss one important interaction-the ability to produce ODM markup.

XML LIBNAME AND PROC CDISC USE IN BASE SAS 9.13

CDISC ODM V1.2

ODM MARKUP DESCRIPTION

The Operational Data Model (ODM) is a vendor neutral, platform independent format for interchange and archive of data collected in clinical trials. The model represents study metadata, data and administrative data associated with a clinical trial. In addition to archive, the format can also be used as a data transmission format. It is in this role that we see ODM poised as a replacement for SAS Transport format in submissions. Like SAS Transport format, the ODM's XML-based format allows expression of common metadata attributes and actual data representation to be included within the same transmission file, but unlike its more vintaged counterpart, the ODM removes eight character name restrictions present in the SAS transport format.

Application of the ODM, however, can also be a non-trivial exercise in creating and maintaining metadata and data relationships. Table content, column attributes, data primary and "foreign" key relationships all must be managed by the exporter of ODM markup. Likewise, input of an ODM markup file to an analysis ready dataset presents some of the same challenges of sorting through markup chaff on the way to obtaining our data kernel (content).

METADATA CONTENT AND REQUIREMENTS

Metadata and data are combined in a typical ODM file. A logical half of the markup is dedicated to describing the attributes and relationships of columns and tables, while the two-thirds remaining is dedicated to actual data content. Typical column (variable) attributes such as data type, length, label, and format can all expressed in the metadata fundamental level. Special constructions in the ODM, called CodeLists, enable the inclusion of user-created SAS format information along with the data content which utilizes those definitions.

DOMAIN DATA CONTENT AND REQUIREMENTS

SDS as a "standard" defines all tables, and all attributes of columns contained in each table. The SDS domains of data initially came from the FDA Guidance regarding Providing Regulatory Submissions in Electronic Formats – NDAs in Case Report Tabulations ODM (proper) is not so strictly limited, and can define and contain arbitrary content. In either case, it is the metadata section that contains the detail of what is "data" and how that data must be rendered in the markup.

XML LIBNAME OPTIONS FOR ODM (ONE FILE EXAMPLE)

The SAS Libname mechanism is a powerful addition to SAS data handling capabilities. Via the Libname non-native data formats are automatically converted to internal SAS representations. The SAS XML Libname permits XML-encoded data to be accessed directly from SAS programs, turning XML into just another data source for SAS.

INPUT EXAMPLE (SAS XML LIBNAME ENGINE)

```
/* SAS 9 (update 9.1.3) */
Filename Chicago ‘CTChicago.xml’;
Libname Chicago xml xmltype=CDISCODM formatActive=Yes       FormatReplace=No formatLibrary=Work ;

PROC DATASETS DD=CHICAGO; RUN;
PROC CONTENTS DATA=CHICAGO.AE; RUN;
PROC PRINT DATA=CHICAGO.AE; RUN;
```

ODM CODELISTS CONVERSION TO SAS FORMATS

One important feature of the new SAS XML Libname engine ODM type worthy of specific note is the automatic generation of SAS Formats directly from the information contained in the ODM CodeList markup. We can see the formats created by issuing

```
PROC FMTLIB; RUN;
```

OUTPUT EXAMPLE (SAS XML LIBNAME ENGINE)

Creating ODM markup from existing SAS data is equally straight-forward. All the required metadata and relationships are generated automatically via the Libname**, including** creation of ODM CodeLists from user-created SAS Formats referenced by the columns in the source data set.

```
/* SAS 9 (update 9.1.3) */

Filename WYSIWYG ‘AE.xml’;
Libname  WYSIWYG xml xmltype=CDISCODM formatActive=Yes       FormatReplace=No formatLibrary=Work ;

DATA WYSIWYG.AE; SET CHICAGO.AE; RUN;
```

PROC CDISC (MULTIPLE FILE EXAMPLE)

The Libname ODM type implementation has the restriction of a single data set being available per invocation. Suppose we want to include multiple datasets within a single ODM file (or we want to generate ODM markup from SAS version 8.2). The task complexity increases dramatically as we must now manage table relationships, including potential concurrent columns and conflicting attributes. Enter the CDISC procedure.

```
/* +------------------------------------------------------------+
   |                                                            |
   +------------------------------------------------------------+ */

FILENAME  ODMOUT  'EXAMPLE.XML';
PROC CDISC        MODEL=ODM
                  OUTREF=ODMOUT
                  ;
ODM                  ODMVersion = "1.2"
                  FileOID = "000.00.0000"
                  FileType = Snapshot
                  Description = "Multiple file ODM example"
                  ;
STUDY              DATA = CURRENT.STUDY(rename=(OID=StudyOID)) ;
GLOBALVARIABLES   DATA = CURRENT.GLOBALS ;
BASICDEFINITIONS  DATA = CURRENT.BASIC ;
METADATAVERSION   DATA = CURRENT.METADATA ;

PRESENTATION      DATA = CURRENT.PRESENT ;
USER              DATA = CURRENT.USERS ;
LOCATION          DATA = CURRENT.LOCATION ;
SIGNATURE         DATA = CURRENT.SIGNATURE ;

CLINICALDATA      DATA = CURRENT.AE
                  DOMAIN  = "AE"
                  NAME    = "Adverse Events"
                  ;

CLINICALDATA      DATA = CURRENT.CONMED
                  DOMAIN  = "CONMED"
                  NAME    = "Concomitant Medications"
                  ;
RUN;
FILENAME  ODMOUT ;
```

Unlike the SAS XML Libname engine ODM enhancement, PROC CDISC allows user-control of much of the metadata content placed in the destination file. The syntax permits definition of many sections of the ODM including Administration and Study dependent information. Metadata content may be stored in supplemental datasets or supplied on a per invocation basis via statement parameters.

ODM DATA VIEWER

The ODM Data Viewer is a stand-alone graphical user interface (GUI) tool, which assists in navigating metadata and data content. It displays file metadata in a meaningful, operator-oriented fashion, in addition to displaying file data in a neat, tabular format for ODM markup files and define.xml files(which are based on the ODM markup). SAS data set and SAS Transport file content can also be displayed by the Viewer. With the transition from SAS Transport to ODM format for submissions this new tool is seen as a future replacement for the SAS system viewer.

CONCLUSION

We have reviewed the regulatory events, standards, tools, applications and processes that SAS has developed to bring CDISC standards into operation for the pharmaceutical industry. This paper addresses the interactive use of two key standards, ODM and SDS, which serve as the operational data standard for archive/repository and the data standard for content/CRTs in regulatory submissions, respectively. We demonstrated how SAS applications provide model management techniques for registration and deployment of the SDS model that supports both registering the source data and mapping the data to the appropriate domain. SAS metadata management allows the bottom-up inter-conversion of SAS datasets to the new XML domain structure of SDS. ODM, on the other hand, allows the more flexible top-down fill of clinical data and provides a new dimension for metadata management for therapeutic product development. SAS tools clearly facilitate ODM as a replacement for SAS Transport format in e-submissions and provide the easy transposition to the Submission Data Structure central to the efficiencies that the FDA is pursuing. The demonstrated out and backing validation of this process with Good Clinical Practice pivotal trial data has shown the utility of the technology.

REFERENCES

CDISC Standards and Supportive Documentation
Guidance for Industry – Providing Regulatory Submissions in Electronic Format –NDAs(January 1999)
Draft Guidance for Industry – Providing Regulatory Submissions in Electronic Format – General Considerations(October 2003)

ACKNOWLEDGEMENTS

None

CONTACT INFORMATION

XMLEngine@ SAS.com – Anthony Friebel and the XML Development team
Ed.Helton@SAS.COM
jboone@webclin.com
Michael.kilhullen@sas.com
Eugenia.basto@sas.com