PhUSE 2015 - Paper CD11

Dataset-XML – A New CDISC Standard for Data Exchange

Lex Jansen, SAS Institute Inc., Cary, NC, USA
Julie Maddox, SAS Institute Inc., Cary, NC, USA

ABSTRACT

Dataset-XML defines a new CDISC XML standard format for transporting tabular data in XML between any two entities. That is, in addition to supporting the transport of data sets as part of a submission to the FDA, it may also be used to facilitate other data interchange use cases. For example, the Dataset-XML data format can be used by a CRO to transmit SDTM or ADaM data sets to a sponsor organization. Dataset-XML supports SDTM, ADaM, and SEND CDISC data sets but can also be used to exchange any other type of tabular data set. In 2014 the FDA conducted a pilot to evaluate Dataset-XML as a solution to the challenges of the SAS XPORT v5 transport format.

The metadata for a data set contained within a CDISC Dataset-XML document must be specified using the CDISC Define-XML standard. Each CDISC Dataset-XML document contains data for a single data set, but a single CDISC Define-XML file describes all the data sets included in the folder. Both CDISC Define-XML v1.0 and CDISC Define-XML v2.0 are supported for use with CDISC Dataset-XML.

This paper will introduce the Dataset-XML standard and present SAS based tools to transform between SAS data sets and Dataset-XML documents, including validation.

This paper assumes that the reader is familiar with some basic XML concepts, and also with the CDISC Define-XML standard for exchanging metadata. Reference [1] contains both a short overview of the XML needed to understand this paper, and also an overview of the structure of a define.xml file based on Define-XML version 1.0.0. A detailed overview of differences between CRT-DDS version 1.0.0 and Define-XML version 2.0.0 can be found in “Define-XML v2 - What’s New” [2].

INTRODUCTION

In the United States, the approval process for regulated human and animal health products requires the submission of data from clinical trials and other studies as expressed in the Code of Federal Regulations (CFR). The FDA established the regulatory basis for wholly electronic submission of data in 1997 with the publication of regulations on the use of electronic records in place of paper records (21 CFR Part 11). In 1999, the FDA standardized the submission of clinical and non-clinical data using the SAS Version 5 XPORT Transport Format and the submission of metadata using Portable Document Format (PDF), respectively. In 2005, the Study Data Specifications published by the FDA included the recommendation that data definitions (metadata) be provided as a Define-XML file.

It has been recognized that the ASCII-based SAS Version 5 XPORT Transport Format has some limitations:

Technical limitations

- Data set and Variable name length limitation (8)
- Data set and Variable label length limitation (40)
- Character variable data lengths limitation (200)
- Limited data types (Character, Numeric)
- Very limited international character support (only ASCII)

Structural limitations

- Two-dimensional “flat” data structure for hierarchical/multi-relational “round” data
- Lack of robust information model

On November 5, 2012, the FDA held a meeting entitled “Regulatory New Drug Review: Solutions for Study Data Exchange Standards”, the purpose of which was to solicit input regarding the advantages and disadvantages of current and emerging open, consensus-based standards for the exchange of regulated study data [3]. Dataset-XML was presented as an alternative for consideration.

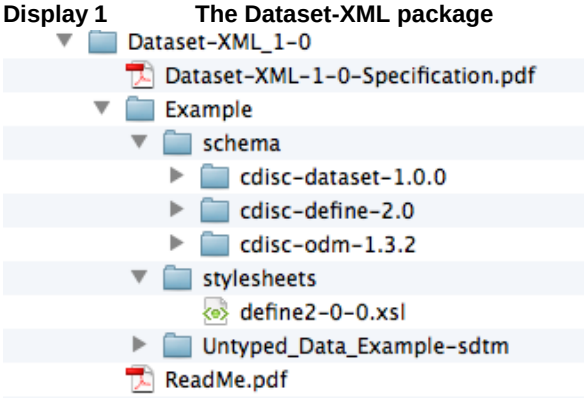
The following alternative standards were presented to replace the SAS Version 5 XPORT Transport format for the exchange of regulated study data:

- SAS Transport extensions (SAS Version 8 Transport format, available in SAS 9.3), addresses the character size issues

- CDISC Operational Data Model (ODM)
- HL7 Version 3 – including Clinical Document Architecture (CDA)
- Semantic Web Technologies:
    - Resource Description Framework (RDF)
    - Web Ontology Language (OWL)
- Analytic Information Markup Language (AnIML)

CDISC published draft version 1.0 of a new ODM based Dataset-XML standard (originally named StudyDataSet-XML or SDS-XML) on November 19, 2013 for public comment. In April 2015 CDISC published the final version 1.0 of the Dataset-XML standard. The Dataset-XML download (http://www.cdisc.org/dataset-xml) includes the Dataset-XML 1.0 specification, a ReadMe file and an example folder and supporting files [4]. Display 1 shows the organization of the Dataset-XML download package.

Display 1 The Dataset-XML package



In response to the development of Dataset-XML, the Center for Drug Evaluation and Research (CDER) and the Center for Biologics Evaluation and Research (CBER) of the U.S. Food and Drug Administration (FDA) released a notice on November 27, 2013 of their intent to begin a pilot project to evaluate Dataset-XML [5]. In the notice, it was highlighted that "although [SAS Transport] has been a reliable exchange format for many years, it is not an extensible modern technology," and that "FDA is announcing an invitation to sponsors to participate in this pilot project to evaluate the SDS-XML transport format." The objective of this pilot was to test the transport functionality of DS-XML, which included ensuring that data integrity was maintained and that DS-XML format would support longer variable names, labels, and text fields.

From May to July 2014 a number sponsor companies converted SDTM compliant datasets for a previously submitted phase 3 study to the FDA to the Dataset-XML format and re-submitted to the FDA. The data was accompanied by a Define-XML file. The FDA performed two types of tests:

- Data Processing Test
    - The XML data can be converted to a sas7bdat file.
    - The converted data is readable and can be viewed by FDA data analysis software (e.g., JMP).
- Data Matching Test
    - Data integrity can be preserved during transport from SAS to DS-XML and vice versa such that there is no loss of information, either metadata (data type) or variable values.
    - The converted data from the Data Processing Test matches the original sas7bdat file.

SAS partnered with the pharmaceutical company Novo Nordisk, who was one of the participants in the Dataset-XML pilot project, to develop tools for working with Dataset-XML files in the SAS System. Novo Nordisk used the SAS tools that are the topic of this paper to create the Dataset-XML files from SAS data sets. They also used the same tools for quality assurance: validating against the XML schema and converting the Dataset-XML files back to SAS to compare against the original data. The FDA used the SAS tools to validate the Dataset-XML files against the XML schema, convert the Dataset-XML files to SAS data sets and to compare the converted data sets to the original submitted data in SAS Version 5 XPORT Transport Format. The FDA was able to successfully convert the Dataset-XML files from the 6 sponsor companies to SAS data sets.

In April 2015 the FDA published a report to communicate the Dataset-XML pilot project findings [6]. The report mentions the following conclusion from the pilot was:

- Dataset-XML can transport data and maintain data integrity.
- The Dataset-XML transport format can facilitate a longer variable name (>8 characters), a longer label name (>40 characters) and longer text field (>200 characters).

- Dataset-XML requires stricter encoding in data.
- Dataset-XML requires consistency between datasets and Define.xml.
- Based on the file size observations, Dataset-XML produced much larger file sizes than XPORT, which may impact the Electronic Submissions Gateway (ESG) and may lead to file storage issues.

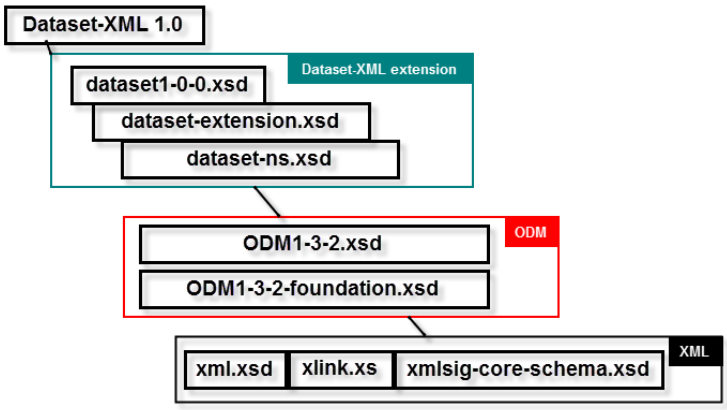
In the summary of the report it is mentioned that additional testing will be needed to evaluate cost versus effectiveness as an alternate transport format. The FDA envisions conducting several pilots to evaluate new transport formats before a decision is made to support a new format.

RELATIONSHIPS OF DATASET-XML TO OTHER CDISC STANDARDS

Operational Data Model (ODM)

The Dataset-XML standard is based on the CDISC Operational Data Model (ODM) XML schema [7].
The CDISC Operational Data Model is a vendor neutral, platform independent XML format for interchange and archival of clinical study data. The model represents clinical subject data along with its associated metadata, administrative metadata, reference data and audit information.
The ODM format is defined by an XML schema and a specification. One of the features of the ODM is a standardized mechanism for defining schema extensions to provide functionality needed to support interchange requirements for specialized use cases. To address the specific needs of data transmission in support of regulatory submissions, CDISC has developed the Dataset-XML model, which is implemented as an extension to the ODM foundation schema. These extensions follow the guidelines for Vendor Extensions provided in the ODM specification and comply with the W3C XML Schema 1.0 specification. The XML schema files for the Dataset-XML standard are available online [4].
Display 2 depicts the extension mechanism.

Display 2 Dataset-XML v1.0 as an ODM 1.3.2 extension



The XML schema files for the Dataset-XML standard are available online at http://www.cdisc.org/dataset-xml.
Since ODM already supports the transmission of clinical subject data, the Dataset-XML extension to ODM 1.3.2 is a very minimal extension:

- In an ODM file there is a hierarchy for clinical data:

```
<ODM>
  <ClinicalData>
    <SubjectData>
      <StudyEvenData>
        <FormData>
          <ItemGroupData>
            <ItemData>
```

  The Dataset-XML schema has simplified this to:

```
<ODM>
  <ClinicalData>
    <ItemGroupData>
      <ItemData>
```

- The Dataset-XML schema has added two attributes:
  - /ODM/@data:DatasetXMLVersion
    (The version of the Dataset-XML standard)

- /ODM/ClinicalData/ItemGroupData/@data:ItemGroupDataSeq
  (A unique sequence number for each ItemGroupData (record in the dataset))

Define-XML

Dataset-XML defines a standard format for transporting tabular dataset data in XML. A Dataset-XML document by itself does not have any metadata about data set name, data set label, variable name, variable label, variable data type or variable length. The metadata for a dataset contained within a Dataset-XML document must be specified using the Define-XML standard. The Define-XML must be contained within the same folder as the Dataset-XML files. Each Dataset-XML file contains data for a single dataset but a single Define-XML document describes all the datasets included in the folder.

As with Dataset-XML, the Define-XML model is implemented using extensions to the CDISC Operational Data Model (ODM) XML schema. The Define-XML v2.0.0 specification describes a model that defines CDISC SDTM, SEND and ADaM datasets as well as accommodating any other tabular dataset structure [9].

Define-XML v2.0.0 can be used to transmit metadata for any tabular dataset, including the following CDISC standards:

- SDTM Implementation Guide Versions 3.1.2 and higher
- ADaM Implementation Guide Versions 1.0 and higher
- SEND Implementation Guide Versions 3.0 and higher.
- SDTM Implementation Guide for Medical Devices Version 1.0 and higher

One of the key benefits to FDA reviewers is that the Define-XML standard provides both a machine readable format for use by the various FDA software applications and, through the provision of an XSL stylesheet, a browser-based report describing the contents of clinical study datasets.

Note, that XSL stylesheets are not a good solution for viewing Dataset-XML datasets due to performance issues caused by the large file sizes. Stylesheets work fine for the metadata in Define-XML, but would be problematic for Dataset-XML datasets. An application or import into a system as a SAS data set is a better option for viewing and filtering Dataset-XML datasets.

Define-XML v2.0 and later are recommended for use with Dataset-XML. The current production version of Define-XML is version 2.0 [9].

DATASET-XML DOCUMENT STRUCTURE

Object Identifiers (OIDs)

XML attributes whose names end with “OID” are used to uniquely identify specific metadata objects. The value of the OID attribute has no meaning by itself. For example, it would be incorrect to conclude from an ItemOID value like “IT.AE.AETERM” that the variable has a name “AETERM”. It would be equally correct to have a convention like "Item<n>", (e.g. "Item12") or even a completely random unique identifier like "bc3e3f8e-62aa-4be4-879b-f5eb747e0d9e". The only requirement for an object identifier is that it is a string with minimum length of 1. The OIDs used have to be unique, but only within certain contexts. For example, OIDs for each element type inside a MetaDataVersion (e.g., ItemGroupDef or ItemDef) must be unique within the scope of that MetaDataVersion.
Display 3 shows a perfectly valid Dataset-XML fragment. Without the metadata in the Define-XML document we would not know that this describes a record in an AE dataset with variables STUDYID, DOMAIN, USUBJID, AESEQ, AESPID, AETERM and AEDECOD. Also we would not know anything about variable labels, variable lengths or data types or other metadata to be able to reconstruct the original dataset from which this Dataset-XML document was created.

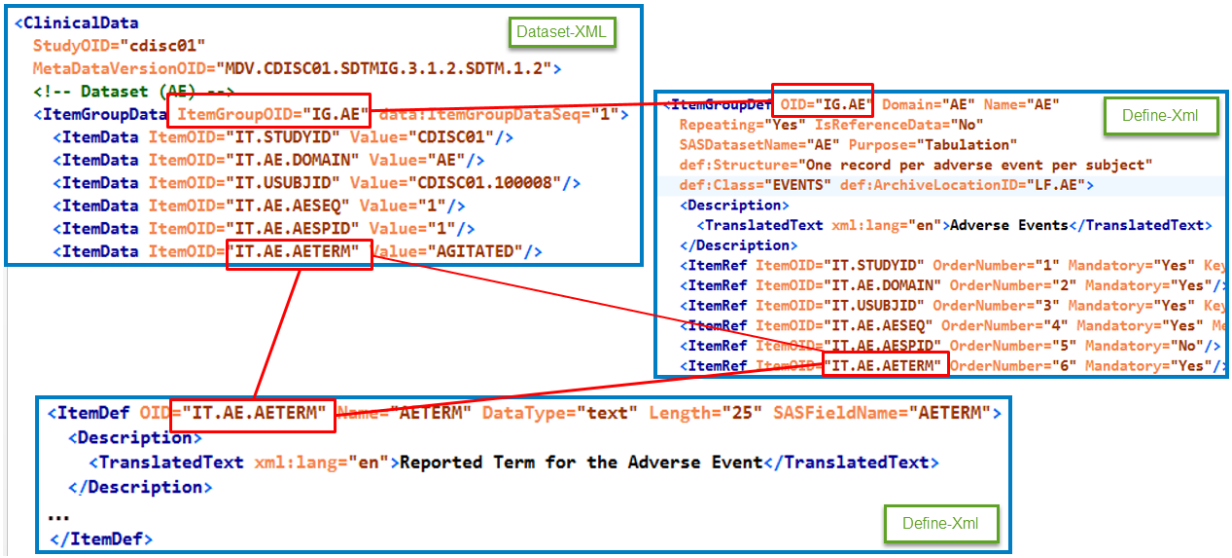
Display 3 Dataset-XML Object Identifiers

```
<ClinicalData
  StudyOID="cdisc01"
  MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
  <ItemGroupData ItemGroupOID="IG1" data:ItemGroupDataSeq="1">
    <ItemData ItemOID="IT1" Value="CDISC01"/>
    <ItemData ItemOID="IT2" Value="AE"/>
    <ItemData ItemOID="IT3" Value="CDISC01.100008"/>
    <ItemData ItemOID="IT4" Value="1"/>
    <ItemData ItemOID="IT5" Value="1"/>
    <ItemData ItemOID="IT6" Value="AGITATED"/>
    <ItemData ItemOID="IT7" Value="AGITATION"/>
```

Displays 4 and 5 show the relation between Dataset-XML and Define-XML through “OIDs”. For example, in the ItemData XML element (Display 4) the ItemOID attribute references a specific ItemDef in the define.xml file containing the variable metadata. We also see that the ItemGroupOID value “IG.AE” has to be the same as the corresponding ItemGroupDef OID attribute value. Finally, we see that all ItemOID attributes in ItemData elements in the Dataset-XML document have values identical to the values of corresponding ItemOID attributes in ItemRef elements that are child elements of the corresponding ItemGroupDef element in the Define-XML document. The ItemOID attributes in ItemData elements in the Dataset-XML document also have values identical to the OID attributes in ItemDef elements in the Define-XML document.

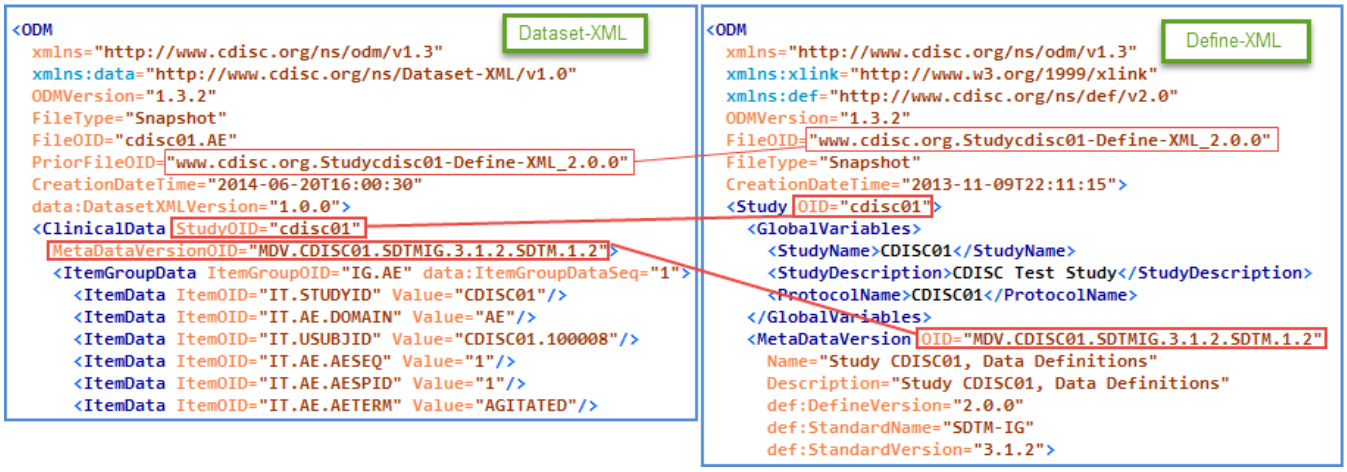
Note that XML is case sensitive, and thus an ItemGroupOID="IG.AE" does not correspond to an ItemGroupDef attribute OID="IG.ae".

Display 4 Relation between Dataset-XML and Define-XML



In Display 5 we see that the StudyOID attribute value "cdisc01" in the ClinicalData element of the Dataset-XML document has to be the same as the OID attribute value in the Study element in the Define-XML document. The ClinicalData element also has a MetaDataVersionOID attribute, whose value must be the same as the OID attribute value in the MetaDataVersion element in the Define-XML document. The Dataset-XML document containing the data may be linked to the Define-XML document containing the metadata by the PriorFileOID attribute on the ODM root element, which will then be the same as the FileOID attribute of the ODM root element in the Define-XML document. This last relation is not required, but optional.

Display 5 Relation between Dataset-XML and Define-XML



Validation of a Dataset-XML document

A valid Dataset-XML document must:

- Properly reference versions of the CDISC standards.
- Be well formed and conform to the XML schemas.
- Meet all of the requirements documented in this specification.

Once a Dataset-XML document is valid according to the schema, validation software should consider all Dataset-XML requirements in the specification. These include rules about conditionally required components and other business rules specified within this document. Schema validation can only enforce some parts of the standard, so this additional level of validation is required to determine whether a Dataset-XML document is fully compliant with Dataset-XML v1.0.0.
The correct ordering of elements within a document is an absolute requirement for the document to be valid with respect to the Dataset-XML schema.

Note that XML is case sensitive, and case sensitivity plays a role in creating a valid Dataset-XML file. For example, ItemgroupOID is not the same element as ItemGroupOID.

USING DATASET-XML

A Dataset-XML dataset provides the data for a single SDTM domain, an ADaM or SEND dataset, or a proprietary tabular dataset. A set of Dataset-XML files would be expected to represent the data for one study. The metadata for Dataset-XML is provided in an accompanying Define-XML file.

Each record in a Dataset-XML dataset is represented by an ItemGroupData element with a unique data:ItemGroupDataSeq attribute. The ItemGroupOID attribute references an ItemGroupDef element within the Define-XML document that contains the dataset metadata. Dataset-XML can represent any tabular dataset that can be described using Define-XML, including SDTM, ADaM, SEND, or non-standard legacy datasets. Each dataset variable value within a dataset record is represented with an ItemData element where the ItemOID attribute references an ItemDef within the Define-XML file that provides the variable metadata. Dataset variables with missing or null values are exceptions to this rule and are not included as ItemData elements. The following ItemData elements are incorrect in representing missing or null values:

```
<ItemData ItemOID="IT.AE.AEDECOD" Value="" />
<ItemData ItemOID="IT.AE.AEDECOD" />
<ItemData ItemOID="IT.AE.AEDECOD" isNull="Yes" />
```

The underlying ODM standard provides two ways to represent data - Typed and Untyped. Dataset-XML version 1.0 only supports the untyped representation of ODM data.

Subject-Level Data and Reference Data

Study subject data for SDTM, SEND or ADaM datasets should be placed under the **ClinicalData** element in the Dataset-XML documents. The IsReferenceData attribute in the ItemGroupDef element in the Define-XML document must in this case be set to “No”.

Reference data is data that is not about clinical study subjects. Examples include SDTM Trial Design datasets, a dataset with Lab Reference Ranges, or a dataset with reference information about a medical device such as Device Identifiers, Device Tracking or Device Properties. Reference data will be contained within the **ReferenceData** element instead of the ClinicalData element. In this cases the ItemGroupDef IsReferenceData attribute is set to “Yes”.

Correct use of the IsReferenceData attribute in the ItemGroupDef elements in the Define-XML document is needed, so that tools do not have to guess where to find ItemGroupData elements in the corresponding Dataset-XML documents.

Display 6 shows records 1 and 16 from an AE (Adverse Events) dataset. The records number is indicated by data:ItemDataGroupSeq attribute. Of the two records shown, the first contains 16 item data values and the second contains 15 item data values. The record with ItemOID="IT.AE.AEMODIFY" in the first record is missing in the second record shown. There are several other items with missing values that are not included as an ItemData element (AEENDTC, AEENDY). In the Define-XML document there must be an ItemGroupDef element with attribute IsReferenceData="No".

Display 6 Example of a Subject-Level Dataset

```
<ClinicalData
  StudyOID="cdisc01"
  MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
  <!-- Dataset (AE) -->
  <ItemGroupData ItemGroupOID="IG.AE" data:ItemGroupDataSeq="1">
    <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
    <ItemData ItemOID="IT.AE.DOMAIN" Value="AE"/>
    <ItemData ItemOID="IT.USUBJID" Value="CDISC01.100008"/>
    <ItemData ItemOID="IT.AE.AESEQ" Value="1"/>
    <ItemData ItemOID="IT.AE.AESPID" Value="1"/>
    <ItemData ItemOID="IT.AE.AETERM" Value="AGITATED"/>
    <ItemData ItemOID="IT.AE.AEMODIFY" Value="AGITATION"/>
    <ItemData ItemOID="IT.AE.AEDECOD" Value="Agitation"/>
    <ItemData ItemOID="IT.AE.AEBODSYS" Value="Psychiatric disorders"/>
    <ItemData ItemOID="IT.AE.AESEV" Value="MILD"/>
    <ItemData ItemOID="IT.AE.AESER" Value="N"/>
    <ItemData ItemOID="IT.AE.AEACN" Value="DOSE NOT CHANGED"/>
    <ItemData ItemOID="IT.AE.AEREL" Value="POSSIBLY RELATED"/>
    <ItemData ItemOID="IT.AE.AESTDTC" Value="2003-05"/>
    <ItemData ItemOID="IT.AE.AESTDY" Value="3"/>
    <ItemData ItemOID="IT.AE.AEENRF" Value="AFTER"/>
  </ItemGroupData>
  ...
  <ItemGroupData ItemGroupOID="IG.AE" data:ItemGroupDataSeq="16">
    <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
    <ItemData ItemOID="IT.AE.DOMAIN" Value="AE"/>
    <ItemData ItemOID="IT.USUBJID" Value="CDISC01.200002"/>
    <ItemData ItemOID="IT.AE.AESEQ" Value="3"/>
    <ItemData ItemOID="IT.AE.AESPID" Value="2"/>
    <ItemData ItemOID="IT.AE.AETERM" Value="PALPITATIONS INTERMITTENT"/>
    <ItemData ItemOID="IT.AE.AEDECOD" Value="Palpitations"/>
    <ItemData ItemOID="IT.AE.AEBODSYS" Value="Cardiac disorders"/>
    <ItemData ItemOID="IT.AE.AESEV" Value="MILD"/>
    <ItemData ItemOID="IT.AE.AESER" Value="N"/>
    <ItemData ItemOID="IT.AE.AEACN" Value="DOSE NOT CHANGED"/>
    <ItemData ItemOID="IT.AE.AEREL" Value="NOT RELATED"/>
    <ItemData ItemOID="IT.AE.AESTDTC" Value="2004-01-05"/>
    <ItemData ItemOID="IT.AE.AESTDY" Value="88"/>
    <ItemData ItemOID="IT.AE.AEENRF" Value="AFTER"/>
  </ItemGroupData>
</ClinicalData>
```

Display 7 shows records 1 and 9 from a TA (Trial Assessments) dataset. The record number is indicated by data:ItemDataGroupSeq attribute. Of the two records shown, the first contains 8 item data values and the second contains 9

item data values. Missing values are again not included as ItemData elements. In the Define-XML document there must be an ItemGroupDef element with attribute IsReferenceData="Yes".

Display 7 Example of a Reference Dataset

```
<ReferenceData
  StudyOID="cdisc01"
  MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
  <!-- Dataset (TA) -->
  <ItemGroupData ItemGroupOID="IG.TA" data:ItemGroupDataSeq="1">
    <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
    <ItemData ItemOID="IT.TA.DOMAIN" Value="TA"/>
    <ItemData ItemOID="IT.TA.ARMCD" Value="PLACEBO"/>
    <ItemData ItemOID="IT.TA.ARM" Value="Placebo"/>
    <ItemData ItemOID="IT.TA.TAETORD" Value="1"/>
    <ItemData ItemOID="IT.TA.ETCD" Value="SCREEN"/>
    <ItemData ItemOID="IT.TA.ELEMENT" Value="Screening"/>
    <ItemData ItemOID="IT.TA.EPOCH" Value="SCREEN"/>
  </ItemGroupData>    ...
  <ItemGroupData ItemGroupOID="IG.TA" data:ItemGroupDataSeq="9">
    <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
    <ItemData ItemOID="IT.TA.DOMAIN" Value="TA"/>
    <ItemData ItemOID="IT.TA.ARMCD" Value="WONDER20"/>
    <ItemData ItemOID="IT.TA.ARM" Value="Miracle Drug 20 mg"/>
    <ItemData ItemOID="IT.TA.TAETORD" Value="3"/>
    <ItemData ItemOID="IT.TA.ETCD" Value="EOS"/>
    <ItemData ItemOID="IT.TA.ELEMENT" Value="End of Study"/>
    <ItemData ItemOID="IT.TA.TABRANCH" Value="Termination from study"/>
    <ItemData ItemOID="IT.TA.EPOCH" Value="TREATMENT"/>
  </ItemGroupData>
</ReferenceData>
```

Data Types

The metadata for a dataset contained within a Dataset-XML document must be specified using the Define-XML standard. This metadata includes metadata about the data type of the variables in the datasets. In CDISC SDTM, SEND and ADaM variable data types are specified as "Char" (character) or "Num" (numeric)[1]. Since Define-XML is based on ODM, it supports a much richer set of data types. The following table shows the mapping between the allowed data types in Define-XML and the CDISC data types:

Define-XML Data Type	CDISC Submission Data Type	Length in Define-XML
Text	Char	Maximum allowable length.
integer	Num	The largest allowable integer width.
Float	Num	The largest allowable whole number width plus the maximum number of decimal digits.
datetime, date, time, partialDate, partialTime, partialDatetime, incompleteDatetime, durationDatetime	Char	N/A

The date and time related data types represent the planned specificity of the collected data, and not an interpretation of the actual collected values. Also, for the date and time related data types the length will not be specified in the Define-XML file.

Data precision when converting to XML

Each Dataset-XML ItemData element is defined by its corresponding ItemDef element in the associated Define-XML document. The ItemDef element has a DataType attribute that specifies how the corresponding ItemData Value attribute is to be interpreted for comparison and storage. When data is represented in a binary format, like a SAS XPT file or database table, it must be converted to text for inclusion as a value in a Dataset-XML document. Conversely, Dataset-XML ItemData values may need to be converted into a binary format, such as in a SAS data set. Typically, these values are converted with

[1] These are also the data types supported by SAS Version 5 XPORT Transport Format.

no loss or change in value. However, with the float datatype small differences in precision can be expected close to the machine precision. With the float datatype the number of significant digits must be included in the ItemDef element in the Define-XML document using the SignificantDigits and Length attributes to define the expected precision.

SAS® TOOLS FOR WORKING WITH DATASET-XML

SAS supports Dataset-XML 1.0 in several ways:

1. SAS Clinical Standards Toolkit 1.7 supports the following Dataset-XML features:
    1. Creation of Dataset-XML 1.0 files from study data, with examples from SDTM 3.1.2 and ADaM 2.1.
    2. Import of Dataset-XML 1.0 files into SAS data sets, with examples from SDTM 3.1.2 and ADaM 2.1
    3. Validation of a Dataset-XML 1.0 file against the XML schema definition, as published by CDISC
    4. Comparison of SAS data sets between original SAS study data and SAS study data that was imported from Dataset-XML 1.0 files

    SAS Clinical Standards Toolkit is a separately orderable component that is available at no additional charge to currently licensed SAS customers. Contact your SAS Account Representative to request the toolkit to be added to your Base SAS order.
    See: http://support.sas.com/rnd/base/cdisc/cst/index.html
2. A ZIP file with standalone SAS Macros to support Dataset-XML v1.0.0 can be downloaded that supports the same features as SAS Clinical Standards Toolkit 1.7 for Dataset-XML in a standalone mode (without having to install SAS Clinical Standards Toolkit): http://support.sas.com/kb/53/447.html
    Documentation is available in this document: http://support.sas.com/rnd/base/cdisc/cst/SAS-Dataset-XML-v1.0.0-support.pdf
3. SAS Clinical Data® Integration 2.6 (CDI) supports Creating Dataset-XML files from SAS data sets and also creating SAS data sets from Dataset-XML files.
    CDI uses the SAS Clinical Standards Toolkit 1.7 macros to support Dataset-XML.
    See:
    http://support.sas.com/documentation/cdl/en/clindiug/67579/HTML/default/viewer.htm#n0jixnkxvm9td4n1i91qk318r7bk.htm

The rest of this paper will discuss the macros that are part of SAS Clinical Standards Toolkit 1.7 and the standalone ZIP file that supports Dataset-XML v1.0.0.
Full description of the macros mentioned can be found in the SAS Clinical Standards Toolkit 1.7: Macro API Documentation:
http://support.sas.com/documentation/onlinedoc/clinical/1.7/cst1.7-macro-api/index.html

Creating Dataset-XML Files from SAS Data Sets

The **%datasetxml_write** macro can be used to convert a library of SAS data sets to Dataset-XML files. This requires a Define-XML file. The Define-XML file that describes the SAS data sets must contain metadata information about all SAS data sets and all variables to be converted. The Dataset-XML files by themselves do not have any information about the SAS data sets (name and label) or the SAS variables (name, label, data type, length, and display format). When the Dataset-XML file is converted back to SAS data sets, this information must be provided by the Define-XML file.
In the macro call below the SAS data sets in the **sdtmdata** library will be converted to Dataset-XML files in the **dataxml** library. The **defxml** filename statement defines the Define-XML file.

```
%datasetxml_write(
  _cstSourceLibrary=sdtmdata,
  _cstOutputLibrary=dataxml,
  _cstSourceMetadataDefineFileRef=defxml,
  );
```

Some of the other parameters that can be specified:

_cstOutputEncoding	The XML encoding to use for the Dataset-XML files to create (Default=UTF-8)
_cstCheckLengths	The actual value lengths of variables with DataType=text are checked against the lengths as defined in the metadata. If the lengths as defined in the metadata are too short, a warning is written to the log file (Y/N, default=N)
_cstIndent	Indent the Dataset-XML file (Y/N, default=Y)
_cstZip	Zip the Dataset-XML file to a zip file in the same folder and with the same name as the Dataset-XML file (Y/N, default:N)
_cstDeleteAfterZip	Delete the Dataset-XML file after it is zipped (Y/N, default=N)).

Displays 8 and 9 illustrate a Dataset-XML file and a Define-XML file.

Display 8 Dataset-XML Fragment

```
<?xml version="1.0" encoding="UTF-8"?>
<ODM xmlns="http://www.cdisc.org/ns/odm/v1.3"
  xmlns:data="http://www.cdisc.org/ns/Dataset-XML/v1.0"
  ODMVersion="1.3.2" FileType="Snapshot" FileOID="cdisc01.AE"
  PriorFileOID="www.cdisc.org.Studycdisc01-Define-XML_2.0.0"
  CreationDateTime="2014-06-23T13:18:18"
  data:DatasetXMLVersion="1.0.0">
  <ClinicalData StudyOID="cdisc01"
    MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
    <ItemGroupData ItemGroupOID="IG.AE" data:ItemGroupDataSeq="1">
      ...
      <ItemData ItemOID="IT.AE.AETERM" Value="AGITATED"/>
```

Display 9 Define-XML Fragment

```
<ODM ... >

  <Study OID="cdisc01">
    ...
    <MetaDataVersion OID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2"
      Name="Study CDISC01, Data Definitions"
      Description="Study CDISC01, Data Definitions"
      def:DefineVersion="2.0.0" def:StandardName="SDTM-IG"
      def:StandardVersion="3.1.2">
...
      <ItemGroupDef OID="IG.AE"
        Domain="AE" Name="AE" Repeating="Yes" IsReferenceData="No"
        SASDatasetName="AE" Purpose="Tabulation"
        def:Structure="One record per adverse event per subject"
        def:Class="EVENTS" def:ArchiveLocationID="LF.AE">
        ...
          <ItemRef ItemOID="IT.AE.AETERM" OrderNumber="6" Mandatory="Yes"/>
...
       <ItemDef OID="IT.AE.AETERM" Name="AETERM" DataType="text" Length="25"
         SASFieldName="AETERM">
```

SAS tables and columns are matched to the ItemGroup SASDatasetName attribute (or, if this value is not specified, the Name attribute) and the ItemDef SASFieldName attribute (or, if this value is not specified, the Name attribute). SASDatasetName and SASFieldName are optional attributes in a Define-XML file but the Name attribute is always available on the ItemGroupDef and ItemDef elements.

In case the ItemGroup or ItemDef cannot be found in the Define-XML file, the Dataset-XML is generated with the following pattern for the ItemGroupOID and ItemOID attributes:

ItemGroupOID = ”IG.<table>”
ItemOID = “IT.<table>.<column>”

Although the ItemGroupOID and ItemOID attributes are generated for missing ItemGroupDef elements or missing ItemDef elements, it is important to realize that this may lead to problems later when converting Dataset-XML files to SAS data sets. It is highly recommended to fix the Define-XML file for this missing metadata.

Warnings are written to the SAS log file and a results data set. Here is an example:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Columns not found in metadata: ADAE.AEDECOD
ADAE.AETERM
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Missing ItemData/@ItemOID for column=AEDECOD
has been set to IT.ADAE.AEDECOD
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Missing ItemData/@ItemOID for column=AETERM
has been set to IT.ADAE.AETERM
```

70	DATA0097	1	DATASETXML_WRITE	Metadata used from C:\datasetxml\xmldata\adam\define_adam.xml	Info
71	DATA0098	2	DATASETXML_WRITE	Columns not found in metadata: ADAE.AEDECOD ADAE.AETERM	Warning
72	DATA0098	3	DATASETXML_WRITE	Missing ItemData/@ItemOID for these columns will be generated as IT.<TABLE>.<COLUMN>	Warning
73	DATA0097	4	DATASETXML_WRITE	ADAMDATA.ADAE converted to C:\datasetxml\xmldata\adam\adae.xml in 0.39 seconds (106 records).	Info
74	DATA0097	5	DATASETXML_WRITE	Zip file C:\datasetxml\xmldata\adam\adae.zip was created	Info

The IsReferenceData attribute in the Define-XML file determines whether the data set is considered ReferenceData or ClinicalData:

```
<ReferenceData StudyOID="cdisc01"
  MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
  <ItemGroupData ItemGroupOID="IG.TE" data:ItemGroupDataSeq="1">
    <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
    <ItemData ItemOID="IT.TE.DOMAIN" Value="TE"/>
    <ItemData ItemOID="IT.TE.ETCD" Value="EOS"/>
    <ItemData ItemOID="IT.TE.ELEMENT" Value="End of Study"/>
    <ItemData ItemOID="IT.TE.TESTRL" Value="Study Termination"/>
    <ItemData ItemOID="IT.TE.TEDUR" Value="P1D"/>
  </ItemGroupData>

<ClinicalData StudyOID="cdisc01"
  MetaDataVersionOID="MDV.CDISC01.SDTMIG.3.1.2.SDTM.1.2">
  <ItemGroupData ItemGroupOID="IG.AE" data:ItemGroupDataSeq="1">
    <ItemData ItemOID="IT.STUDYID" Value="CDISC01"/>
    <ItemData ItemOID="IT.AE.DOMAIN" Value="AE"/>
    <ItemData ItemOID="IT.USUBJID" Value="CDISC01.100008"/>
    <ItemData ItemOID="IT.AE.AESEQ" Value="1"/>
    <ItemData ItemOID="IT.AE.AESPID" Value="1"/>
    <ItemData ItemOID="IT.AE.AETERM" Value="AGITATED"/>
```

Although not essential for the creation of Dataset-XML files, setting the _cstCheckLengths macro parameter to “Y” enables the macro to determine whether the lengths defined in the metadata are long enough for character data. Warnings are written to the SAS log file and a results data set. Here is an example:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short: __ItemGroupOID=IG.ADAE
__ItemOID=IT.ADAE.AETERM Length=20 _valueLength=24 value=HEARTBURN-LIKE DYSPEPSIA
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short: __ItemGroupOID=IG.ADAE
__ItemOID=IT.ADAE.AETERM Length=20 _valueLength=25 value=ACID REFLUX (OESOPHAGEAL)
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short: __ItemGroupOID=IG.ADAE
__ItemOID=IT.ADAE.AEDECOD Length=20 _valueLength=32 value=Gastrooesophageal reflux
disease
WARNING: [CSTLOGMESSAGE.DATASETXML_WRITE] Length too short: __ItemGroupOID=IG.ADAE
__ItemOID=IT.ADAE.AETERM Length=20 _valueLength=25 value=ACID REFLUX (OESOPHAGEAL)
```

8	DATA0097	1	DATASETXML_WRITE	Metadata used from C:\SASPlaypen\TK1.7\US7405_CreateDatasetXML\sourcexml_adam\define_adam.xml	Info
9	DATA0098	2	DATASETXML_WRITE	Check Log for potential length issues: ADAE.AEDECOD ADAE.AETERM	Warning
10	DATA0097	3	DATASETXML_WRITE	SRCDATA.ADAE converted to C:\SASPlaypen\TK1.7\US7405_CreateDatasetXML\sourcexml_adam\adae.xml	Info
11	DATA0097	4	DATASETXML_WRITE	Zip file C:\SASPlaypen\TK1.7\US7405_CreateDatasetXML\sourcexml_adam\adae.zip was created	Info

This check is important to avoid data truncation problems when importing the Dataset-XML files into SAS data set with the **%datasetxml_read** macro.

The **%datasetxml_write** macro also checks that numeric variables in ADaM data sets that represent Date/Time information have a DisplayFormat defined in the Define-XML file.

Creating SAS Data Sets from Dataset-XML files

The **%datasetxml_read** macro can be used to convert a library of Dataset-XML files to SAS data sets. This requires a Define-XML file. The Define-XML file that describes the Dataset-XML files must contain metadata information about all SAS data sets and all variables to be created. The Dataset-XML files by themselves do not have any information about the SAS data sets (name and label) or the SAS variables (name, label, data type, length, and display format). When the Dataset-XML file is converted to SAS data sets, this information must be provided by the Define-XML file.
In the macro call below the Dataset-XML files in the **dataxml** library will be converted to SAS data sets in the **sdtmdata** library. The **defxml** filename statement defines the Define-XML file.
Character variables that represent Date/Time related information in ADaM or SDTM data conform to the ISO 8601 standard and do not have a length specified in the Define-XML file. The cstDatetimeLength macro parameter specifies the length to use for these variables when they are converted to SAS data sets. The _cstAttachFormats parameter defines whether display formats, as defined in the Define-XML file, will be attached to the variables.

```
%datasetxml_read(
  _cstSourceDatasetXMLLibrary=dataxml,
  _cstOutputLibrary=sdtmdata,
  _cstSourceMetadataDefineFileRef=defxml,
  _cstDatetimeLength=64,
  _cstAttachFormats=Y
  );
```

If the lengths of character variables are too short to hold the data, warnings are written to the SAS log file and a results data set in the results folder. Here is an example of length issues:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too short for
ItemGroupDataSeq=12 IT.ADAE.AETERM value=HEARTBURN-LIKE DYSPEPSIA (length=
24)
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too short for
ItemGroupDataSeq=25 IT.ADAE.AETERM value=HEARTBURN-LIKE DYSPEPSIA (length=
24)
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too short for
ItemGroupDataSeq=28 IT.ADAE.AETERM value=ACID REFLUX (OESOPHAGEAL)
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] TRUNCATION occurred: Length=20 too short for
ItemGroupDataSeq=28 IT.ADAE.AEDECOD value=Gastrooesophageal reflux disease
(length=32)
```

71	DATA0097	2	DATASETXML_READ	Data read from C:\datasetxml\xmldata\adam\adae.xml	Info
72	DATA0098	3	DATASETXML_READ	Please check the LOG. There were data truncation issues for table ADAE	Warning
73	DATA0097	4	DATASETXML_READ	Data set adamdata.ADAE created in 0.92 seconds (106 records)	Info

Inconsistencies between the Dataset-XML file and the Define-XML file, which can lead to issues with matching data to metadata, are written to the SAS log file and a results data set in the results folder. Here is an example:

```
WARNING: [CSTLOGMESSAGE.DATASETXML_READ] Items not found in metadata: IT.ADAE.AEDECOD
IT.ADAE.AETERM
```

70	DATA0097	1	DATASETXML_READ	Metadata used from C:\datasetxml\xmldata\adam\define_adam.xml	Info
71	DATA0097	2	DATASETXML_READ	Data read from C:\datasetxml\xmldata\adam\adae.xml	Info
72	DATA0098	3	DATASETXML_READ	Items not found in metadata: IT.ADAE.AEDECOD IT.ADAE.AETERM	Warning
73	DATA0097	4	DATASETXML_READ	Data set adamdata.ADAE created in 0.96 seconds (106 records)	Info

In this example, even though the Dataset-XML file contained records identified by ItemOID="IT.ADAE.AEDECOD" and ItemOID="IT.ADAE.AETERM, the ADAE data set is created without the AETERM and AEDECOD variables, since there is no metadata available that defines these variables. This is correct behavior, since it is not allowed to derive the variable names from the Object identifiers.

Validating Dataset-XML Files Against an XML Schema

Here is an example that validates a Dataset-XML and a Define-XML file against the XML schemas as published by CDISC:

```
%cstutilxmlvalidate(
  _cstXMLStandard=CDISC-DATASET-XML,
  _cstXMLStandardVersion=1.0.0,
  _cstXMLPath=c:\cstSampleLibrary/cdisc-datasetxml-1.0.0-1.7/sourcexml/ae.xml
  );

%cstutilxmlvalidate(
  _cstXMLStandard=CDISC-DEFINE-XML,
  _cstXMLStandardVersion=2.0.0,
  _cstXMLPath=c:\cstSampleLibrary/cdisc-datasetxml-1.0.0-1.7/sourcexml/ae.xml
);
```

With some data step code this can be easily extended to a directory of Dataset-XML files:

```
filename xmlcode CATALOG "work._cst.xmlvalidate.source";

data _null_;
  length xmlfile $400;
  file xmlcode;
  rc=filename("fref", "&_cstDatasetXMLFolder");
  did=dopen("fref");
  memcnt=dnum(did);
  do i = 1 to memcnt;
    xmlfile=cats("&_cstDatasetXMLFolder", "/", dread(did,i));
    if scan(upcase(xmlfile), -1) = "XML" then
    do;
        %* A validation error would set _cst_rc=1 and stop further validation*;
        put '%let  cst rc=0;';
        put '%cstutilxmlvalidate(';
        if index(upcase(xmlfile), "DEFINE")=0 then do;
          put '   cstXMLStandard=CDISC-DATASET-XML,';
          put '  _cstXMLStandardVersion=1.0.0,';
        end;
        else do;
          put '  _cstXMLStandard=CDISC-DEFINE-XML,';
          put '  _cstXMLStandardVersion=2.0.0,';
        end;
        put '   cstXMLPath=' xmlfile;
        put '  );';
    end;
  end;
  rc=dclose(did);
run;

%include xmlcode;

data _null_;
  format  character ;
  set & cstResultsDS;
  if (resultflag ne 0) or (upcase(resultseverity) in ('WARNING' 'ERROR')) or
    index(upcase(message), 'XML FILE TO VALIDATE') > 0 or
    index(upcase(message), 'SCHEMA BEING VALIDATED AGAINST') > 0
  then do;
     resultseverity=upcase(resultseverity);
     putlog resultseverity +(-1) ": [CSTLOG%str(MESSAGE)] " message @;
     if not missing(resultdetails) then putlog ": "  resultdetails;
                                   else putlog;
  end;
run;
```

Here is an example of the messages that are written to the SAS log file. Notice that in this case we had some issues in a Define-XML file, where there were issues with the length attribute of a few ItemDef elements:

```
INFO: [CSTLOGMESSAGE] XML File to Validate: c:\cstSampleLibrary\cdisc-datasetxml-1.0.0-
1.7\sourcexml\ae.xml
INFO: [CSTLOGMESSAGE] Schema being validated against: c:\cstGlobalLibrary/schema-
repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
INFO: [CSTLOGMESSAGE] XML File to Validate: c:\cstSampleLibrary\cdisc-datasetxml-1.0.0-
1.7\sourcexml\cm.xml
INFO: [CSTLOGMESSAGE] Schema being validated against: c:\cstGlobalLibrary/schema-
repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
INFO: [CSTLOGMESSAGE] XML File to Validate: c:\cstSampleLibrary\cdisc-datasetxml-1.0.0-
1.7\sourcexml\da.xml
INFO: [CSTLOGMESSAGE] Schema being validated against: c:\cstGlobalLibrary/schema-
repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
INFO: [CSTLOGMESSAGE] XML File to Validate: c:\cstSampleLibrary\cdisc-datasetxml-1.0.0-
1.7\sourcexml\define.xml
INFO: [CSTLOGMESSAGE] Schema being validated against: c:\cstGlobalLibrary/schema-
repository/cdisc-definexml-2.0.0/define2-0-0.xsd
ERROR: [CSTLOGMESSAGE] (Line 3543/Column 98) cvc-enumeration-valid: Value 'Numeric' is not facet-
valid with respect to enumeration '[integer, float, date, datetime, time, text, string, double,
URI, boolean, hexBinary, base64Binary, hexFloat, base64Float, partialDate, partialTime
, partialDatetime, durationDatetime, intervalDatetime, incompleteDatetime, incompleteDate,
incompleteTime]'. It must be a value from the enumeration.
ERROR: [CSTLOGMESSAGE] (Line 3543/Column 98) cvc-attribute.3: The value 'Numeric' of attribute
'DataType' on element 'ItemDef' is not valid with respect to its type, 'DataType'.
ERROR: [CSTLOGMESSAGE] (Line 3569/Column 97) cvc-datatype-valid.1.2.1: '' is not a valid value
for 'integer'.
ERROR: [CSTLOGMESSAGE] (Line 3569/Column 97) cvc-attribute.3: The value '' of attribute 'Length'
on element 'ItemDef' is not valid with respect to its type, 'positiveInteger'.
ERROR: [CSTLOGMESSAGE] (Line 3586/Column 94) cvc-datatype-valid.1.2.1: '' is not a valid value
for 'integer'.
ERROR: [CSTLOGMESSAGE] (Line 3586/Column 94) cvc-attribute.3: The value '' of attribute 'Length'
on element 'ItemDef' is not valid with respect to its type, 'positiveInteger'.
WARNING: [CSTLOGMESSAGE] Document validation failed
ERROR: [CSTLOGMESSAGE] Errors were reported in the XMLTransformLog
INFO: [CSTLOGMESSAGE] XML File to Validate: c:\cstSampleLibrary\cdisc-datasetxml-1.0.0-
1.7\sourcexml\dm.xml
INFO: [CSTLOGMESSAGE] Schema being validated against: c:\cstGlobalLibrary/schema-
repository/cdisc-datasetxml-1.0.0/dataset1-0-0.xsd
```

Comparing Original SAS Data Sets to SAS Data Sets Created from Dataset-XML Files

After converting Dataset-XML files to SAS data sets, it is important to compare those SAS data sets with the original data sets from which the Dataset-XML files were created. The %cstutilcomparedatasets macro compares the original SAS data sets with the SAS data sets that were created from the Dataset-XML files. In the macro call below the SAS data sets in the **database** library will be compared with the SAS data sets in the **datacomp** library. The _cstCompareLevel parameter defines the minimum PROC COMPARE return code which is considered to be an error condition. Values greater than 0, but below the _cstCompareLevel value, are considered warning conditions. _cstCompDetail=Y will give a detailed PROC COMPARE output for data sets that were different. With the cstCompOptions parameter extra PROC COMPARE options can be specified.

```
%cstutilcomparedatasets(
  _cstLibBase=database,
  _cstLibComp=datacomp,
  _cstCompareLevel=16,
  _cstCompOptions=%str(criterion=0.00000000000001),
  _cstCompDetail=Y
  );
```

For every SAS data set that is compared, the macro reports the error code as returned by PROC COMPARE. Here are the error codes:

Bit	Condition	Code	Hex	Description
1	DSLABEL	1	0001X	Data set labels differ
2	DSTYPE	2	0002X	Data set types differ
3	INFORMAT	4	0004X	Variable has different informat
4	FORMAT	8	0008X	Variable has different format
5	LENGTH	16	0010X	Variable has different length
6	LABEL	32	0020X	Variable has different label
7	BASEOBS	64	0040X	Base data set has observation not in comparison
8	COMPOBS	128	0080X	Comparison data set has observation not in base
9	BASEBY	256	0100X	Base data set has BY group not in comparison
10	COMPBY	512	0200X	Comparison data set has BY group not in base
11	BASEVAR	1024	0400X	Base data set has variable not in comparison
12	COMPVAR	2048	0800X	Comparison data set has variable not in base
13	VALUE	4096	1000X	A value comparison was unequal
14	TYPE	8192	2000X	Conflicting variable types
15	BYVAR	16384	4000X	BY variables do not match
16	ERROR	32768	8000X	Fatal error: comparison not done

For example, code=40 (8+32) indicates that a format and a label were different.
This message is written to the SAS log file:

```
WARNING: [CSTLOGMESSAGE.CSTUTILCOMPAREDATASETS] Comparing srcdata.adqs and
trgdata.adqs  - Differences: FORMAT/LABEL (SysInfo=40)
```

When converting SAS data sets to Dataset-XML and then converting back to SAS data sets, the following differences can be expected:

- Date- and time-related columns do not have a length defined in the Define-XML metadata.
  The **%datasetxml_read** macro uses the parameter _cstdatetimeLength to define the length of the date- and time related columns, but can be different from the original lengths.
- SAS numeric variables are created with a length of 8 to avoid loss of precision, even when the original length or the length specified in the Define-XML file is less than 8.
- Character variables (DataType="text") that do not have a length specified in the Define-XML file are created with a length of 200.
- Small differences in precision can be expected around the machine precision for numeric variables that represent real numbers.
- Character data that contains leading spaces or trailing spaces loses the leading and trailing spaces.

By specifying PROC COMPARE options with the _cstCompOptions macro parameter, you can specify that the comparison be less precise (for example, _cstCompOptions=%str(criterion=0.00000000000001)). Lesser precision prevents differences close to machine precision from being reported as errors.

CONCLUSION

SAS supports Dataset-XML files in a convenient and efficient way without needing any XML knowledge.

REFERENCES

1. Lex Jansen (2012). Using the SAS® Clinical Standards Toolkit for define.xml creation. Proceedings of the Pharmaceutical Industry SAS® Users Group (PharmaSUG 2012, San Francisco, CA) (http://www.lexjansen.com/pharmasug/2012/HW/PharmaSUG-2012-HW02-SAS.pdf)
2. Lex Jansen (2013). Define-XML v2 – What's New. Proceedings of the 9th Pharmaceutical Users Software Exchange (PhUSE 2013, Brussels, Belgium) (http://www.lexjansen.com/phuse/2013/cd/CD05.pdf)
3. Information from the November 5, 2012 Solutions for Study Data Exchange Standards Meeting. Retrieved on April 11, 2015. http://www.fda.gov/Drugs/DevelopmentApprovalProcess/FormsSubmissionRequirements/ElectronicSubmissions/ucm332003.htm
4. CDISC Dataset-XML Specification Version 1.0, April 22, 2014 (http://www.cdisc.org/dataset-xml)
5. Transport Format for the Submission of Regulatory Study Data; Notice of Pilot Project. Retrieved on April 11, 2015 https://www.federalregister.gov/articles/2013/11/27/2013-28391/transport-format-for-the-submission-of-regulatory-study-data-notice-of-pilot-project
6. Test Report for DS-XML Pilot. Center for Drug Evaluation and Research (CDER), Center for Biologics Evaluation and Research (CBER), April 8, 2015. Retrieved on April 20, 2015 http://www.fda.gov/ForIndustry/DataStandards/StudyDataStandards/ucm380756.htm#CDISCDSXMLPilotEval
7. CDISC Operational Data Model (ODM), Version 1.3.2, December 1, 2013 (http://www.cdisc.org/odm)
8. Case Report Tabulation Data Definition Specification (define.xml), Version 1.0, February 9, 2005 (http://www.cdisc.org/define-xml)
9. CDISC Define-XML Specification, Version 2.0, March 5, 2013 (http://www.cdisc.org/define-xml)

ACKNOWLEDGMENTS

We would like to thank Mikkel Traun for the opportunity to work together in a successful collaboration during the Dataset-XML pilot.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Lex Jansen
SAS Institute Inc.
SAS Campus Drive
Cary, North Carolina, 27513, USA
Email: lex.jansen@sas.com

Julie Maddox
SAS Institute Inc.
SAS Campus Drive
Cary, North Carolina, 27513, USA
Email: julie.maddox@sas.com