USING PROC TEMPLATE TO CONVERT SAS DATA TO DEFINE.XML

Lucius A. Reinfelt, Steven Kirby, Matthew Wiedel, Aleksandra Stein, Vanessa Huang and Nancy Wang

Celerion

DEFINE.XML OVERVIEW

Well formed data contains all information needed to understand study results, but that information is typically not easily accessible to end users, and combining the data with extensible data documentation goes much further in making it easy to understand. The DEFINE.XML format is an extensible stylesheet language (XSL) format that is commonly used in the pharmaceutical industry for creating data documentation for submission to the FDA. It is a flexible approach that allows for non-representative data sets in various forms. When combined with SAS data, it results in well-formed define.xml is reduced. Code list names and attributes from the variable (unique values by variable and value) levels are integrated into the data sets. The result is back-end XML data. The PROC TEMPLATE code implemented in the DEFINE.XML module is simple to alter and run, allowing more data to be used in the process.

TWO-STEP DATA-DRIVEN DEFINE.XML CREATION USING PROC TEMPLATE

A two-step data-driven process is used to create the define.xml documents. The first step is the production of six well-formed define data sets. The second step is integration of the SAS® define data into an XML formatted document through a robust approach driven by PROC TEMPLATE. Sorted define data sets are restructured and transformed into a well-formed XML format. This process is driven by SAS® PROC TEMPLATE. Sorted define data sets are restructured and transformed into a well-formed define.xml format. The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

DEFINE DATA GENERATION USING DATA MINING

Simple iteration through each submission domain is used to mine the data for the information needed to create the define.xml. The data is read in as six define data sets, with metadata included in each define data set. The metadata includes unique values by variable and value. Well-formed define data sets are created. The information is then transformed into the well-formed define.xml format. The attribute fields are populated by the define datasets. These nested tagsets then are processed by the metadata versions of the tool. The leaf nodes match the data set for the information. The data set is output as six SAS data sets. The result is well-formed define.xml format. This process is driven by SAS® PROC TEMPLATE. Sorted define data sets are restructured and transformed into a well-formed define.xml format. The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

ACKNOWLEDGEMENTS

We want to thank Celerion for the support received for this work, and we want to express our gratitude to the CDISC community for the wealth of public information on the application of data standards.

DEFINE.XML LEVELS OF INFORMATION

XML format allows for multiple levels of information to be hierarchically linked together. The Diagram below shows the different level of information. The data set contains information about the study, the domain, the variable, the variable level, and the variable value. The Diagram below shows the levels of information. The data set contains information about the study, the domain, the variable, the variable level, and the variable value. The DEFINE.XML module is simple to alter and run, allowing more data to be used in the process.

DEFINe.XML LEVELS OF INFORMATION

Variable Level Define data

The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

CREATE DATA TO DEFINE.XML WITH PROC TEMPLATE

Creating a well-structured define.xml file is a multi-step process. The first step is the production of six well-formed define data sets. The second step is integration of the SAS® define data into an XML formatted document through a robust approach driven by PROC TEMPLATE. Sorted define data sets are restructured and transformed into a well-formed define.xml format. The attribute fields are populated by the define datasets. These nested tagsets then are processed by the metadata versions of the tool. The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

READ IN TAGNAMES AND DATA VALUES

The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

FORMAT AND PRINT TAGSET (DefValueListDef tag)

The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

TAGS PRINTED TAGSET DATASETS AS DEFINE.XML

There are metadata levels of information in the define that correspond with each of the input define data sets. Not all information available in the define is presented using the macro, but all information can be stored in various DEFINE.XML modules using the macro application.

DEFINE.XML - VARIABLE LEVEL

The attribute fields are populated by the define datasets. These nested tagsets then are processed by the metadata versions of the tool. The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.

TRIGGER TAGSET EVENTS

The attribute fields are populated by the define datasets. These nested tagsets then are processed by the metadata versions of the tool. The leaf nodes match the data set for the information. The data set is output as six SAS data sets. There is some flexibility in how the datasets are formed. The PROCESS TEMPLATE code is simple to alter and run, allowing more data to be used in the process.