PharmaSUG 2018 - Paper DS-10

Designing Flexible Data Standards Models

Melissa R. Martinez, SAS Institute

ABSTRACT

When we think of CDISC submission data standards, we often focus on the tables and variables that the standards describe. We build products and business processes around creating these tables, but isn't there more to the story than just the submission data sets? This paper explores ways to design data standard models with data processing in mind. It also shows how to add to industry data standard models to define and standardize additional attributes, such as de-identification algorithms, variable calculations, and even blocks of SAS code. Finally, this paper describes features available in the forthcoming SAS Life Science Analytics Framework version 5.1 that facilitate these concepts.

INTRODUCTION

CDISC publishes several industry data set submission standards that provide guidelines for creating the data sets submitted to regulatory agencies. The CDISC data standards describe the end result, the final data sets, but there is so much more information that guides the process of creating the final submission data sets, the supporting documents and the processes around submissions. This paper describes how to make better use of data standards by adding information to them to drive more efficient processes and standardize more of the information that drives the programming and submission work.

DATA STANDARD MODEL, DEFINED

The model for a data standard is all the information, or metadata, that describes the basic data set information, rules for populating the data set, attributes to help create the define.xml, informational fields, and more. Basic information includes things like the variable name, description, type, and length. Examples of other metadata in the model include

- Order variable should appear in the data set
- Codelist used to standardize the variable's values
- Whether a variable is required, expected, or permissible
- CDISC notes from the Implementation Guide

To help visualize the model, you can consider the basic model to be the column headers in the data standard files provided by CDISC. The data standard is the metadata for the data sets to be created. The model is the metadata for the data standard, kind of like a Matryoshka nesting doll, however those column headers are not the complete story. There is more metadata that is used to populate define.xml files, such as the origin of a variable's value or the computational method used for a variable. These additional attributes are typically included as part of data standards in most metadata management systems. The additional attributes are extraneous for the purposes of creating a SAS data set; the data set has no concept of the variable's origin, but those additional attributes can drive the creation of other processes, most commonly the creation of the define.xml file. You can take the idea of additional attributes further to help drive other processes like automating parts of the data set programming, de-identifying data before sharing publicly, or implementing an SDTM-plus model.

ADDING ATTRIBUTES TO A DATA STANDARD MODEL

Additional model attributes are simply additional information about the variables in the data standard. Using the column header example for visualization, adding new attributes to a data standard model is like adding new columns to the data standard definition file. If Excel is used to manage data standards, then adding new attributes is as simple as adding new columns.

When adding attributes to a model, evaluate whether the values of the new attribute may be standardized and add values wherever possible to the data standard. Many of the attributes in the following examples can and should be standardized within an organization, which can be streamlined by managing them along with the data standard. It is easy to lose consistency when data standards are managed in one place, de-identification methodology is managed in another place, and common derivations are managed in yet an additional place.

DATA SET PROGRAMMING SPECIFICATIONS/STUDY METADATA

Most of the time, data set programming specifications look very similar to the data standard itself. All the information in the data standard is helpful for the creation of the data sets. Many of the attributes frequently included in the data standard are used for the creation of the define.xml. Many organizations will save a copy of a data standard in Excel, then add columns to describe programming details like source data sets and variables and the derivations to use to populate each variable. The Excel file is then used as an input to data set creation programs to obtain basic data set metadata like variable name, label, length, and text. It is also used as input to create the define.xml file.

If your organization does not use Excel for data standard management, there is likely a method available in the metadata repository system to extract data standards. The extracted metadata can similarly be used for seeding programming specifications and creating data sets and define.xml files.

The programming specification is what I also call the study metadata. Information that cannot be standardized or generalized at the data standard level will get populated at the individual study level. Data sets and variables not needed in the study will be removed. Referenced codelists and computational algorithms will be verified. References to supporting documents like the Case Report Form will be added to support the define.xml creation. Additional attributes also will be evaluated at the study level and populated or modified as necessary for the individual study.

EXAMPLE 1: DATA SET PROGRAMMING ATTRIBUTES

Most organizations are looking for ways to standardize and automate as much of their data set programming as possible. Adding data set programming attributes to your data standard model can help with that. Suppose we add two attributes to our model, Source Variables and Derivation.

Source Variables could be used to do one-to-one mapping by listing the source variable. This could simply reference the variable name (e.g. studyid) or include a reference to the source data set (e.g. demo.studyid). A SAS program can be written to map the Source Variable to the CDISC data set variable. If both a source data set and variable are referenced, the SAS program can parse the data set and variable from Source Variables, aggregate the list of input data sets and variables for the entire CDISC data set, and use that to build the data steps to read the data sets and variables into the program.

In a more complex application, populate Source Variables with all variables needed to derive the value. For example, if the USUBJID value is created by concatenating STUDYID, SITEID, and SUBJID, list all three in the Source Variables. Then use the Derivation attribute to place SAS code to carry out the derivation. A SAS program can parse Source Variables to determine the variables (and source data sets) required for input and pull the code from the Derivation right into the SAS program for execution.

There are several Source Variable and Derivation values that could be standardized within an organization. The remaining values can be added in the study metadata. It is unlikely this use case would allow you to populate an entire CDISC data set, but it can help get the simple mapping and derivation done with some SAS macro development to make use of the Source Variable and Derivation attributes. Programmers can then spend the bulk of their time programming the more challenging derivations.

EXAMPLE 2: DE-IDENTIFICATION ATTRIBUTES

Clinical trial data is being shared outside of original research organizations more and more frequently as regulators worldwide require more transparency and public interest is rising in performing further research on this publicized data. A critical piece of the transparency process is de-identifying the clinical trial data before it is shared publicly.

Jack Shostak wrote a fantastic paper called De-Identification of Clinical Trails Data Demystified that describes his methodology for data deiidentification and provides SAS macro examples. Using his macros as an example, an additional data standard attribute can make the de-identification process even simpler.

Jack's method requires identifying the subject identifier used throughout the trial data sets, all date and datetime variables, the reference start date for the study, and variables to be dropped from the de-identified data sets. The subject identifier will be replaced with a random value, dates and datetimes will be replaced with an integer number indicating the number of days away from the study reference start date it is, and variables to be dropped will be removed.

Jack's sample programs require passing in these variables to the macros. However, by adding a De-Identification attribute to your data standard, the value of De-Identification in the study metadata could be trigger values like USUBJID, DATE, and DROP. The macros Jack developed could then be modified to read the study metadata to find the variables to be used in the de-identification macros. Many of these values could likely be standardized and managed with the data standard, saving time for programmers and increasing consistency and standardization across studies.

EXAMPLE 3: SDTM-PLUS ATTRIBUTES

The SDTM data model is used for submitting collected data to regulatory agencies. As Barry R. Cohen points out in his paper SDTM, Plus or Minus, SDTM is a data submission standard and not an operational data standard. There are some cases where having data in exact SDTM format makes business processes harder than they need to be during the development period of a clinical trial. Many organizations use an SDTM-plus model during the development period to make the operation easier and will convert the data set to exact SDTM closer to submission time.

One example of SDTM-plus is keeping variables that belong in a supplemental qualifier domain in their parent data set. For example, adverse event collection pages sometimes have a clinical interest flag, which is not a standard AE variable in SDTM. In this case, the clinical interest flag value belongs in SUPPAE. In an SDTM-plus model, the flag values would be stored in AE as part of the parent observation.

Additional data standard model attributes can help by providing a flag for variables that need to be separated out into supplemental qualifier domains and by specifying the IDVAR and QLABEL. In Table 1 below, AECLINT is the clinical interest flag, and the additional attributes provide the information needed for a SAS program to extract AECLINT from the AE SDTM-plus data set and add it to SUPPAE. As with previous examples, some of the additional attribute values may be standardized within an organization.

Domain	Variable Name	SUPPQUAL Flag	IDVAR	QLABEL
AE	AETERM	N		
AE	AECLINT	Y	AESEQ	Clinical Interest

Table 1. Sample SDTM-plus Additional Metadata Model Attributes

HOW CAN SAS LIFE SCIENCE ANALYTICS FRAMEWORK HELP?

SAS Life Science Analytics Framework 5.1 is scheduled for release in late 2018. This major release includes a whole new clinical metadata management component allowing customers to manage their data standards, controlled terminology, and study metadata in the same application where they do their statistical programming work. One of the exciting features in this release is the ability to extend and customize the metadata models that SAS provides as part of the application. Working with their SAS Solutions OnDemand Industry Consultant, customers may request new attributes be added to their data standard models, including attributes discussed in this paper.

Users may extract data standards and study metadata into SAS data sets in SAS Life Science Analytics Framework, which they can then leverage to drive SAS programming processes. Automated processes can be made even more efficient and streamlined by making use of the SAS Life Science Analytics

Framework API to extract data standards and study metadata directly from the SAS programs that make use of the extracted metadata. With the flexibility offered in the metadata models in this next major release of SAS Life Science Analytics Framework, SAS can support a wide variety of metadata models to meet the diverse needs of life science industry organizations.

CONCLUSION

Metadata that enhances the information about data sets for data processing beyond submission data sets may be defined and stored with data standard metadata by adding attributes to the data standard model. This allows streamlined standardization and management of these complementary metadata attributes and more efficient programming.

REFERENCES

Shostak, Jack. 2006. “De-Identification of Clinical Trials Data Demystified.” *Proceedings of the PharmaSUG 2006 Conference*, Bonita Springs, FL. Available at https://www.lexjansen.com/pharmasug/2006/PublicHealthResearch/PR02.pdf.

Cohen, Barry R. 2007. “SDTM, Plus or Minus.” *Proceedings of the PharmaSUG 2007 Conference*, Denver, CO. Available at https://www.lexjansen.com/pharmasug/2007/fc/FC02.pdf.

ACKNOWLEDGMENTS

Thank you, Bill Gibson, Keith Appleby, Carl Haske, Kevin Alderton, Chandrika Nayak, Shari Forvendel, Ken Ellis, Steve Smith, Michael Kilhullen, Lex Jansen, Karen Chaney, Sarah Holshouser, Mandy Myers, Pete Armstrong, Gene Lightfoot, Michael Meadows, and Julie Maddox for soliciting my input for requirements, allowing me to participate in the development process, and for all of your hard work and dedication in building the clinical metadata management component of SAS Life Science Analytics Framework. I am grateful to have been a part of this team. Thank you, Bill Gibson and Sharon Trevoy for always having prompt, helpful feedback. Thank you, Angela Lightfoot for continuing to support me in my professional development and pursuit of my career goals.

CONTACT INFORMATION

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

Melissa R. Martinez
Principal Health and Life Science Industry Consultant
SAS Institute
Melissa.Martinez@sas.com