PharmaSUG 2015 - Paper AD02

A Way to Manage Clinical Project Metadata in SAS® Enterprise Guide

David Wang, Sanofi, Bridgewater, New Jersey

ABSTRACT

This paper will show how we use an existing standard (SDTM, ADaM, or Company standard ADS) as a starting point to build and maintain a project metadata, which can be used across all studies under a same project. This paper will use a real submission project experience to demonstrate how this approach was used during ADS development and how it saved a lot of time in a submission project. It can guarantee the SAS variable contributes (e.g. label, length and type) to be consistent across studies. It also helps the project programming lead to manage all studies in the creation of ADS and its define documents as well as ADS pooling process. All changes in the metadata were managed with SAS Enterprise Guide without third party software. It has been a quite convenient tool to use by including the same SAS macro in each of ADS programs. Major SAS code will be included in the paper.

INTRODUCTION

We use a company standard ADS metadata as a starting point to build and maintain a project metadata, which can be used across all studies under a same project. This paper will use a real submission project experience to demonstrate how this approach was used during ADS development and how it saved a lot of time in a submission project. It can guarantee the SAS variable contributes (e.g. label, length and type) to be consistent across studies. It also helps the project programming lead to manage all studies in the creation of ADS and its define documents as well as ADS pooling process. All changes in the metadata were managed with SAS Enterprise Guide without third party software.

METADATA PREPARATION

We used a company standard ADS metadata in excel file per domain to build a project metadata in SAS (see figure 1). We saved the metadata in a central area, in which only the project programming lead has a write access.

Figure 1. Metadata Structure

MYPROJECT

Filter and Sort | Query Builder | Data | Describe | Graph | Analyze | Export | Send To

	VORDER	INFLAG	VNAME	VLABEL	VTYPE	VLENGTH	DOMAIN	STUDY	DEFINE	STANDARD	NOTE	POOL
1	1	y	DOMAIN	Domain Abbreviat...	Text	2	AE	all	all			y
2	2	y	AESEQ	Sequence Number	Integer	8	AE	all	all			y
3	3	n	AEREFID	Reference ID	Text	6	AE	all	all			
4	4	n	AESPID	Sponsor-Defined...	Text	6	AE	all	all			y
5	5	y	AETERM	Reported Term fo...	Text	200	AE	all	all			y
6	5.1	y	AETERM1	SAE-Specify 1	Text	200	AE	STUDY1	STUDY1			y
7	5.2	y	AETERM2	SAE-Specify 2	Text	200	AE	STUDY2, STUDY3, STUDY4	all			y
8	6	y	AEMODIFY	Modified Reporte...	Text	200	AE	all	all			y
9	6.1	y	AEMREASN	Modified Reason...	Text	30	AE	all	all	n	myproject	y
10	7	y	AEDECOD	Dictionary-Derive...	Text	120	AE	all	all			y
11	8	n	AECAT	Category for Adve...	Text	200	AE	all	all			
12	9	n	AESCAT	Subcategory for A...	Text	200	AE	all	all			
13	10	n	AEPRESP	Pre-Specified Adv...	Text	1	AE	all	all			
14	11	y	AEBODSYS	Body System or...	Text	120	AE	all	all			y
15	12	y	AESOCCD	System Organ Cl...	Text	8	AE	all	all			y
16	13	n	AESOC1FL	BODSYS Primary...	Text	1	AE	all	all			
17	14	y	AEPTCD	Preferred Term C...	Text	8	AE	all	all			y
18	15	y	AELLT	Lowest Level Term	Text	120	AE	all	all			y
19	16	y	AELLTCD	Lowest Level Ter...	Text	8	AE	all	all			y
20	17	n	AELLTCUR	LLT Current	Text	1	AE	all	all			
21	18	y	AEHLGT	High Level Group...	Text	120	AE	all	all			y
22	19	y	AEHLGTCD	High Level Group...	Text	8	AE	all	all			y
23	20	y	AEHLT	High Level Term	Text	120	AE	all	all			y
24	21	y	AEHLTCD	High Level Term...	Text	8	AE	all	all			y
25	22	y	AEORDER	Body System or...	Integer	8	AE	all	all			y

Here are the definitions for each of above variables:

VORDER - order of variable;

INFLAG – indicator of variable active or inactive;

VNAME - variable name;

VLABEL - variable label;

VLENGTH - variable length;

DOMAIN - domain short name in 2 or 3 letter, e.g. AE, DM, CM etc.;

STUDY - name of individual study;

DEFINE - name of define document;

STANDARD – indicator of standard variable or not;

NOTE - anything, e.g. project name;

POOL - indicate if it is included in integrated datasets.

DATASET SHELL

We can use above metadata to build the dataset shell for each ADS through a macro call. The main code of the macro is shown below. The main function of the code is to filter the metadata to the specific study.

```
options source2;
%macro creshell(domain=, deflib=, defname=, varname=, typename=, lengname=, labname=, fmtname=);

*************************************************************************;
* Get customized data for the specific domain                           *;
*************************************************************************;
%let domain=%upcase(&domain);

%if %upcase(&w_analysis.)=MAPP %then %do;
    /*for MAP folder named MAPP only otherwise MAPP has to the actual name*/
    data work.def_&domain.;
        set &deflib..&defname.;
        if upcase(inflag)="Y" and upcase(domain)="&domain."
           and (index(upcase(study),"ALL")>0 or index(upcase(study),"&W_MSTUDY.")>0);
    run;
```

Macro parameters:

DOMAIN – name of domain;

DEFLIB – name of SAS library reference for storing the metadata;

DEFNAME – name of metadata dataset;

VARNAME – name of variable name in metadata;

TYPENAME – name of variable type in metadata;

LENGNAME – name of variable length in metadata;

LABNAME – name of variable label in metadata;

FMTNAME – name of variable format in metadata.

From above code we can see how to filter the data for a specific domain. If variable INFLAG='Y' mean the variable is active and variable STUDY ='ALL' or the name of this study means only the variables applicable to this specific study

are picked. Once we filter the metadata to the specific domain and specific study we can start to build a SAS code to create the dataset shell as below based on all information in the metadata.

```
*************************************************************************;
*  Write sql script file for extracting data from Oracle DB             *;
*************************************************************************;

    data _null_;
      file shelloc;
      put ;
      put @4 '*-------------------------------------------------------------------------------*;' ;
      put @4 "*    Create a file containing all domain specific variables "          @83 '*;' ;
      put @4 "*    Domain name      :  &domain "                                     @83 '*;' ;
      put @4 "*    Compound         :  &w_compound"                                  @83 '*;' ;
      put @4 "*    Study            :  &w_study"                                     @83 '*;' ;
      put @4 "*    Creation date    :  &sysdate9"                                    @83 '*;' ;
      put @4 "*    User ID          :  &sysuserid"                                   @83 '*;' ;
      put @4 "*    SAS version      :  &sysver"                                      @83 '*;' ;
      put @4 '*-------------------------------------------------------------------------------*;' ;
      put ;
    run;

    data _null_;
      file shelloc mod;
      set temp end=eof;
      if _n_ = 1 then do;
          if upcase(domain)='COM' then do;
             put @2 '%macro comvar;' ;
          end;
          else do;
             put @2 '%macro domvar;' ;
          end;
          put ;
          if upcase(type) in ("TEXT", "DATETIME", "CHAR") then do;
             if format ne ' ' then do;
                put @10 ' ' variable '    char(' length ')    label = "' label '"  format = ' format ;
             end;
             else do;
                put @10 ' ' variable '    char(' length ')    label = "' label '"';
```

DATASET CREATION

The code below shows how to start the ADS program with a include SAS program and macro call, which create the actual common variables and domain variable shell. Common.sas is used to create the common variable in SAS work area. The macro call is to create the dataset shell for demographic ADS.

```
*** program initialization;
%include MAC_A(common.sas);

*** Create macro domvar **;
%creshell(domain=dm, deflib=deflib, defname=gz427051, varname=vname, typename=vtype, lengname=vlength, labname=vlabel);

*** Beginning of ADDM core program;
proc sql;
  create table work.addm
  (%comvar
  ,
  %domvar
);
  quit;
run;

*** Create keep variables ;
proc contents data=work.addm out=temp2;
run;

proc sql noprint;
  select name into: keepvar separated by ' '
  from temp2;
quit;

%let keepvar=&keepvar.;
%put &keepvar.;
```

The code below shows the end section of ADS creation, in which proc append and proc sort are used to append the final temporary data for the domain to the dataset shell, and sort the data and output it to a permanent area.

```
data final;
     merge dm(in=a) adcommon bgroup;
     by patid;
     if a;
  run;

  *** append to ADDM shell ;
proc append base=addm data=final(keep=&keepvar) force;
  run;

  *** Sorting and labeling of the final data set;
proc sort data=addm out=adsd.addm (label="Demographics");
       by studyid subjid ;
  run;

  **** End of addm.sas;
```

INTEGRATED DATASET

The code below shows how to use the same approach to create integrated dataset quickly.

Below is the partial code of macro which is used to create the dataset shell. The variable POOL decides what variables are chosen for the integrated ADS.

```
options source2;
%macro creshell(domain=, deflib=, defname=, varname=, typename=, lengname=, labname=, fmtname=);

*********************************************************************;
* Get customized data for the specific domain                       *;
*********************************************************************;
%let domain=%upcase(&domain);

data work.def_&domain.;
    set &deflib..&defname.;
    if upcase(pool)="Y" and upcase(domain)="&domain." ;
run;
```

The code below shows the macro call in the ADS to create the integrated datasets.

```
*** program initialization;
%include MAC_A(common.sas);

*** Create macro domvar **;
%creshell(domain=ds, deflib=deflib, defname=gz427051, varname=vname, typename=vtype, lengname=vlength, labname=vlabel);

*** Beginning of ADDS core program;
```

CONCLUSION

Through managing the project metadata we can make sure all variable attributes to be consistent and compliant with FDA submission requirements (e.g. length of variable and label) across a whole project. The way we present in this paper is very easy to use during the ADS programming development. It also helps the project programming lead to manage all studies in the creation of ADS and its define documents as well as ADS pooling process. All changes in the metadata were managed with SAS Enterprise Guide without third party software.

CONTACT INFORMATION

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

David Wang
Sanofi
55 Corporate Drive,
Bridgewater, NJ 08807
david.wang@sanofi.com or david_2_wang@yahoo.com