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
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.

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

%let domain=%upcase(&domain);

%if %upcase(&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),"&STUDY.")>0);
  run;

%macro end;
%mend creshell;

%macro creshell;

%mend;
```

**Macro parameters:**

- **DOMAIN** – name of domain;
- **DEFLIB** – name of SAS library reference for storing the metadata;
- **DEFNAME** – name of metadata dataset;
- **VNAME** – 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 *

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

**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.
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.

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

*** Create macro domvar **;
%reshell(domain=dm, deflib=deflib, defname=gs427051, varname=vname, typename=type, lenname=length, labname=label);

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

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

proc sql nolog;
   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.

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

*** append to ADMI 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 subj id;
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.

```sas
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.

```sas
*** program initialization;
%include MAC_1(common.sas);

*** Create macro domar ***;
%creshell(domain=ds, deflib=dslib, defname=ps4270sl, varname=vname, typename=type, lengname=length, labname=label);

*** Beginning of ADS 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

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. © indicates USA registration.

Other brand and product names are trademarks of their respective companies.