Paper SM09

Utility Macro for Ensuring Consistent Updates of CRF Data in SDTM

Naren Surampalli, Pfizer, Groton, CT USA

ABSTRACT

The source data collected from the Case Report Form (CRF) pages is generally recorded in clinical trial databases. Various organizations may employ different tools to extract this source data and appropriately map it to the corresponding SDTM datasets. Commonly, not all CRF page data is available at the onset of a trial. As the trial advances, additional data is gathered from growing CRF pages. The updated information from the new CRFs must be accurately extracted and synchronized for both the source and SDTM data. This paper will outline a utility macro that incorporates multiple checks to ensure that updates to source data are reliably captured in the SDTM datasets from the initiation of a clinical trial until its completion, thereby reducing the likelihood of overlooking critical data.

INTRODUCTION

Most of the companies have tools to create SDTM data from the raw clinical trial data. At the beginning of the trial, not all CRF data is available. As the study progresses, new data from different CRF pages gets collected and there is a possibility that this CRF data might not get mapped to SDTM domains due to oversight. It becomes programmer's responsibility to verify and ensure all source data is captured in SDTM domains. It's very time consuming to check every time to ensure all source data is captured in SDTM. In this paper, we discuss the details of a macro, which helps to check the source data is consistently captured correctly from the beginning of the study to the end of the trial and avoid the risk of missing any source data in SDTM domains.

STUDY DESIGN SPECIFICATIONS (SDS)

Study design specifications are required for each study. It includes the list of CRF forms and their visits required for a study, which eventually help in building a database by central designer. The first place to get list of all CRF forms and their form names is SDS. Usually, SDS is in an excel format, which can be easily read into SAS. In the development of the macro, we first read the SDS for the study in scope to get the form names into a SAS dataset. Below is the snippet of the code to read the SDS.

```
** read the SDS into SAS dataset.

proc import datafile = "//&path. /&sds..csv"
dbms=xlsx replace out=allsubj;
sheet="Time and Events";
datarow=10;
getnames=yes;
run;
```

TEMPLATE FOR RAW OR SOURCE DATASET LIBRARY

After reading the form names from SDS into SAS dataset in the above step, next create a template for raw or source dataset library. This can be achieved by using the SDS created dataset as input and mapping all the form names to the relevant transformed raw/source dataset names. The newly created dataset will include the form names captured in a variable, expected mapped source dataset names and their relevant expected mapped SDTM names. This shell can be independent of therapeutic area and can include all expected source data. This step is done to ensure CRF form names get their relevant transformed raw/source datasets and helps us to check valid records for relevant form names.

** code below to create the template for raw/source dataset library.
Note: only few datasets are selected in the template to show as examples. Table 1 lists the form and the corresponding mapped raw/source memname and relevant SDTM name.

```
data temp_allsubj (keep=memname sdtmname form);
 set allsubj;
  length memname $32 sdtmname $32;
          if form = 'D_INFSCR001' or index (c, 'DS001') or form = 'D_INFENR001'
```

```
            then do; memname='SM_DS001'; sdtmname='DS'; output; end;
            else if index(form,'X_DS_SLOPD001') then do; memname='SM_SLOPD'; sdtmname='DS'; output; end;
            else if index(form,'SV001') then do; memname ='SM_SV001'; sdtmname='SV'; output; end;
            else if index(form,'DM001') then do; memname ='SM_DM001'; sdtmname='DM'; output; end;
            else if index(form,'IE002') then do; memname ='SM_IE002'; sdtmname='IE'; output; end;
             else if index(form,'CM00') then do; memname ='SM_CM001'; sdtmname='CM'; output; end;
            else if index(form,'MH00') then do; memname ='SM_MH001'; sdtmname='MH'; output; end;
            else if index(form,'LB001_3') then do; memname ='SM_LB001_3'; sdtmname='LB'; output; end;
             else if index(form,'LB001') then do; memname ='SM_LB001'; sdtmname='LB'; output; end;
            else if index(form,' BOX') then do; memname ='SM_LBOXY004'; sdtmname='LB'; output; end;
            else if index(form,'VS00') then do; memname ='SM_VS001'; sdtmname='VS'; output; end;
            else if index(form,'EG00') then do; memname ='SM_EG001'; sdtmname='EG'; output; end;
            else if index(form,'EC001_2') then do; memname ='SM_EC001_2';
              sdtmname='EC'; output;
              sdtmname='EX'; output;
             end;
run;
```

Table 1:

Obs	form	memname	sdtmname
1	AE001	SM_AE001	AE
2	AE001_1	SM_AE001	AE
3	BEETRK001_2	SM_BEETRK001	BE/CO
4	BEETRK001	SM_BEETRK001	BE/CO
5	BEETRK001_5	SM_BEETRK001	BE/CO
6	BEETRK001_3	SM_BEETRK001	BE/CO
7	BEETRK001_4	SM_BEETRK001	BE/CO
8	CECISR004	SM_CECISR004	FACE
9	CECISR004	SM_CECISR004	CE
10	CECISR004_1	SM_CECISR004	FACE
11	CECISR004_1	SM_CECISR004	CE
12	CECISR004_2	SM_CECISR004	FACE
13	CECISR004_2	SM_CECISR004	CE
14	CEIDT008	SM_CEIDT008	CE
15	CESOD006	SM_CESOD006	FACE

IDENTIFY VALID RECORDS IN SOURCE DATASET

After the source design specifications (SDS) and source dataset template are created, read the source/raw datasets in scope for a study and merge with the above created datasets. Keep only the datasets in scope for the study.

Below source code to read raw/source data and keep only records in scope:

```
proc sql noprint;
  create table rawv as
   (select memname from dictionary.tables
    where libname="MAPR" and memtype="DATA" and substr(memname,1,2) ='SM' and
index(memname,'SM_R_TRIG')=0
```

```
    and index(memname, 'VARD')=0)
  order by memname;
 quit;

data raw_check (keep=memname sdtmname form rename=(form=formref));
 merge allsubj1(in=i)  rawv(in=j);
  by memname;
 if j;
run;
```

To check valid records in source datasets, each source dataset has a field called notdonestate captured in the database, which helps us to know whether it's a valid record (notdonestate=0) or invalid record (notdonestate=1). Below code is used to check for valid records in each source dataset. The goal is checking only those source datasets with valid records and ensure the information is captured in their relevant SDTM datasets.

Macro check is used to check valid records for each source dataset. Formrefname field and notdonestate field are compared.

```
%macro check(din=);

proc freq data=mapr.&din. noprint;
 tables FORMREFNAME*notdonestate/out=freq0 ;
 where notdonestate=0;
run;
proc append BASE=final_valid DATA=freq0 force;
run;

%mend;

data _null_;
 set raw_check (where=(formrefname ne ''));
 call execute('%check(din='||strip(memname)||');');
run;

proc sort data=final_valid nodupkey;
  by formrefname ;
run;

proc sort data=raw_check ;
  by formrefname ;
run;

data final (drop=count percent);
  merge final_valid(in=i) raw_check(in=j);
   by formrefname;
    if i;
run;

 ** Table 2 lists all the source/raw datasets with valid records.
     Table 3 lists the datasets which has no valid records.
     Table 3 source data will not have relevant SDTM created.

proc sort data=final;
  by memname;
run;
proc sort data=rawv;
  by memname;
run;
data final_raw raw_miss;
  merge final(in=i) rawv(in=j);
```

```
   by memname;
    if j and not i then output raw_miss;
    if i and j then output final_raw;
run;

proc sort data=final_raw nodupkey;
 by memname;
run;
```

Table 2:

Raw datasets with valid records - CRF

Obs	memname
1	SM_AE001
2	SM_BEETRK001
3	SM_CECISR004
4	SM_CEIDT008
5	SM_CESOD006
6	SM_CM001
7	SM_CVECH001
8	SM_DD002
9	SM_DM001
10	SM_DS001

Table 3:

Raw datasets with no valid records - CRF

Obs	memname
1	SM_PRTFD002

IDENTIFYING VALID RECORDS IN SDTM DATASETS

Once the valid raw datasets are identified, half the battle is won. For the valid raw/source datasets, identify the valid expected SDTM datasets based on the template library. Read in the extracted SDTM datasets for a particular study and check if all the expected SDTMs are extracted based on the valid raw/source datasets. Flag any SDTMs which are not extracted or extracted but doesn't have a valid source data. Please note that there could be valid SDTM datasets extracted based on non-CRF data specifications like immunogenicity and reactogenicity.

Below macro is to read in the extracted SDTM datasets into a data step. Please note RELREC and SUPP. domains are not compared.

```
proc sql noprint;
   create table sdtm as
    (Select memname as sdtmname from dictionary.tables
     where libname="MAPD" and memtype="DATA" and index(memname,'CDASH')=0 and
     index(memname,'VOL3')=0 and index(memname,'_')=0 and index(memname,'SUPP')=0 and
index(memname,'RELREC')=0)
   order by memname;
quit;
```

Check valid SDTM datasets captured in final_sdtm data step and any SDTMs missed captured in sdtm_miss data step.

```
data final_sdtm sdtm_miss;
  merge final(in=i) sdtm(in=j);
   by sdtmname;
```

```
    if i and not j then output sdtm_miss;
    if i and j then output final_sdtm;
run;

proc sort data=final_sdtm nodupkey;
  by memname sdtmname;
run;
```

Table 4 below lists all valid raw datasets with relevant SDTM datasets extracted.

Table 4:

Raw datasets with relevant SDTM datasets with valid records - CRF

Obs	memname	sdtmname
1	SM_AE001	AE
2	SM_BEETRK001	BE/CO
3	SM_CECISR004	CE
4	SM_CECISR004	FACE
5	SM_CEIDT008	CE
6	SM_CESOD006	CE
7	SM_CESOD006	FACE
8	SM_CM001	CM
9	SM_DM001	DM
10	SM_DS001	DS
11	SM_EC001_2	EC
12	SM_EC001_2	EX
13	SM_FAFUC009	SV
14	SM_FAVSD011	CE
15	SM_FAVSD011	FACE

Table 5 below lists all the valid raw datasets but no SDTM extracted/present:

Table 5:

Raw datasets present and its relevant SDTM datasets not present

Obs	memname	sdtmname
1	SM_CVECH001	CV
2	SM_DD002	DD
3	SM_EG001	EG
4	SM_CESOD006	FAAE
5	SM_MH001	MH
6	SM_PR001	PR

IDENTIFYING THE SDTM DATASETS WITH MISSING CRF DATA

In clinical trial study designs, we typically see multiple CRF pages collected to capture the same information but for different time periods. For example, disposition form is captured in different CRF pages for screening, randomization, treatment period and follow-up. All these CRF pages gets mapped to one SDTM called DS domain. For an ongoing study, not all CRF pages gets updated for the participant. Only during final CSR, we typically see the complete data.

The macro also checks if any of the new CRF page is collected but is missing in SDTM dataset. This check helps us to follow up with the mapping team to ensure all the information is correctly captured from source/raw to SDTM.

```
** check if any CRF page available in raw/source but missing in SDTM;

data final (drop= percent);
  merge final_valid(in=i) raw_check(in=j);
   by formrefname;
    if i;
run;

proc sort data=final;
 by sdtmname;
run;

data final_check1;
 set final;
  by sdtmname;
  retain sum 0;
  if first.sdtmname then sum=count;
  else sum = sum + count;
run;

proc sort data=final_check1;
 by sdtmname;
run;

data final_check2 (keep=memname sdtmname sum);
 set final_check1;
  by sdtmname ;
   if last.sdtmname;
run;

proc sql noprint;
create table chk as select memname as sdtmname, nobs from dictionary.tables
where libname='MAPD' and index(memname,'SUPP')=0 and index(memname,'REL')=0;
quit;

data crf_check;
 merge final_check2(in=i) chk(in=j);
  by sdtmname;
  if nobs ne sum then output;
run;
```

Table 6 lists the different CRF pages for the same unique form. Table 7 lists the unique SDTM and their number of observations. We can observe here that number of observations for all DS CRF pages don't add up to the final SDTM DS domain. One of the CRF page X_DS_SLOPD001 was not mapped to SDTM. This check helps us identify any information available in source/raw data but missed in SDTM domain.

Table 6

Multiple CRF forms for the same domain

Obs	FORMREFNAME	NOTDONESTATE	COUNT	memname	sdtmname	sum
1	AE001	0	41	SM_AE001	AE	41
5	CM001	0	1	SM_CM001	CM	1
6	CM001_2	0	648	SM_CM001	CM	649
7	CM001_3	0	685	SM_CM001	CM	1334
8	CM001_4	0	41	SM_CM001	CM	1375
9	DM001	0	354	SM_DM001	DM	354
10	DS001	0	354	SM_DS001	DS	354
11	DS001_3	0	352	SM_DS001	DS	706
12	DS001_4	0	353	SM_DS001	DS	1059
13	DS001_5	0	1	SM_DS001	DS	1060
14	DS001_6	0	5	SM_DS001	DS	1065
15	DS001_7	0	5	SM_DS001	DS	1070
16	X_DS_SLOPD001	0	353	SM_SLOPD	DS	1423
17	EC001_2	0	477	SM_EC001_2	EC	477
18	EC001_2	0	477	SM_EC001_2	EX	477
23	MH001	0	269	SM_MH001	MH	269
24	SV001	0	513	SM_SV001	SV	513

Table 7

Number of observations for each SDTM domain

Obs	sdtmname	nobs
1	AE	41
3	CM	1375
4	DM	354
5	DS	1070
7	EC	477
8	EX	477
11	MH	269
12	SV	513

A corrective measure to avoid the above situation would be to ensure all the unique forms applicable to the study design listed in study design specifications (SDS) should be reviewed by the programmer and ensure all the CRF forms are utilized in the SDTM mapping programs to avoid future mapping issues with SDTM datasets.

CONCLUSION

The macro discussed in this paper helps the study teams to ensure the valid raw/source and SDTM datasets are accurately captured for each study and ensure we report all data in the CSRs accurately. This macro should be executed after each extraction done by the study team to ensure all relevant source/SDTM datasets are captured and any missing datasets should be extracted or investigated to ensure no data is missed.

The macro doesn't check non-CRF source vs expected SDTM data but gives a summary of expected non-CRF SDTM datasets.

ACKNOWLEDGMENTS

I would like to express my gratitude to the Pfizer management and organization for giving me the opportunity to present the paper.

My sincere thanks to Richa Kala and Liping Zhang for their invaluable support during the development/review of the paper.

CONTACT INFORMATION

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

Author Name: Naren Surampalli
Company: Pfizer Inc.
Address: 280 Shennecossett Rd, Groton, CT 06340
Work Phone: 862-222-4479
Email: naren.surampalli@pfizer.com
Website: www.pfizer.com