PharmaSUG 2021 - Paper QT-111

A Unique Way to Identify the Errors in SDTM Annotations Using SAS®

Xingxing Wu, Bala Dhungana, Eli Lilly and Company, Indianapolis, IN

ABSTRACT

The annotated Case Report Form (aCRF) is a vital file of SDTM development and FDA submission. Therefore, it is very important to maintain the correctness of SDTM annotations in the aCRF. However, a single study would usually have several thousands of SDTM annotations and integrated database (IDB) would have over ten thousands of SDTM annotations. This makes it very difficult and time-consuming to check all the annotations manually. In addition, some errors in the SDTM annotations are not obvious and it is very easy to overlook these errors if checking them manually. To overcome these issues, this paper proposes a unique way to check the errors in SDTM annotations using SAS. This approach uses SAS to extract all the SDTM annotations from aCRF and then compares them with the pre-defined rules. The SDTM annotations with issues could then be identified if they do not follow these rules. The proposed approach can greatly improve the quality of aCRF in an efficient way.

INTRODUCTION

SDTM is a standard way to format and organize data and is the standard for pharmaceutical companies to submit data to FDA. Before creating SDTM, we first need to create the aCRF which is the document to map the collected clinical data to the SDTM variables. The aCRF is an important reference file during SDTM development, it is also an important file in the Case Report Tabulation (CRT) package submitted to FDA. The accuracy plays a critical role in maintaining the quality of SDTM. However, in most cases, the SDTM annotations are added to the blank CRF manually. It is inevitable to have some mistakes in the annotations. In addition, a single study would have over thousands SDTM annotations and a IDB would have over ten thousands of SDTM annotations. The following is an example of the SDTM annotations on a single CRF page. Furthermore, the submission to FDA might include several studies in addition to the IDB.

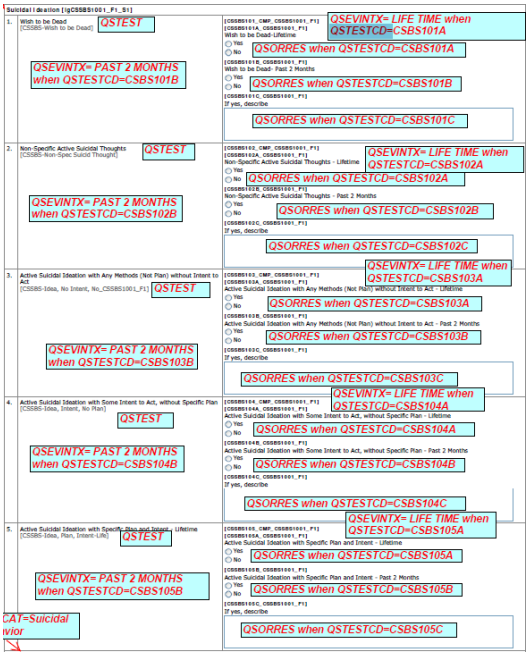


Figure 1. Annotated Case Report Form

From Figure 1, we can see it would be a very difficult and time-consuming task to find the issues with the SDTM annotations if we just check these annotations totally manually. Also, some mistakes are not obvious, it is very easy to overlook these mistakes. In order to overcome all these difficulties and improve the accuracy of the aCRF, the paper proposes a unique approach to check the SDTM annotations using SAS. The proposed approach can greatly improve the quality of aCRF in an efficient way. This paper will first briefly introduce the whole process of this approach using flowchart, and then demonstrate how to extract the SDTM annotations from aCRF into SAS datasets. Finally, this paper will discuss how to identify the errors in the SDTM annotations based on pre-specified rules.

HOW TO IDENTIFY ERRORS IN SDTM ANNOTATIONS USING SAS

As mentioned above, we can easily overlook some mistakes in the SDTM annotations if we just check the SDTM annotations manually. Furthermore, the manual check would take a lot of time. If fact, we can use SAS programs to help check the SDTM annotations and identify the errors in a more efficient way. The SDTM annotations can be extracted from aCRF PDF file into a SAS dataset. After this step, we can use SAS to manipulate these SDTM annotations, identify and output the SDTM annotations with errors. The following is the flowchart of the process.

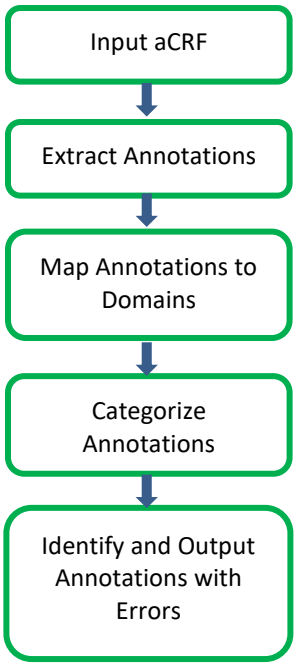


Figure 2. Identify Errors in SDTM Annotations Using SAS

First, we need to read aCRF into a SAS dataset, and then identify and extract the SDTM annotations and related information, such as background color, from the aCRF. The next step is to identify the domain annotations and associate each variable annotation with related domain. We can then divide the annotations into different categories. This will help identify the errors in the SDTM annotations. If the annotations belong to 'NOT SUBMITTED' or 'NOTES', we then do not need to check whether these annotations have issues or not. The last step is to check these SDTM annotations with pre-specified rules based on their categories. The SDTM annotations that do not follow these rules will be considered as

SDTM annotations with issues and will be put into a report. We will introduce the key steps in more details.

INPUT CRF INTO SAS DATASET

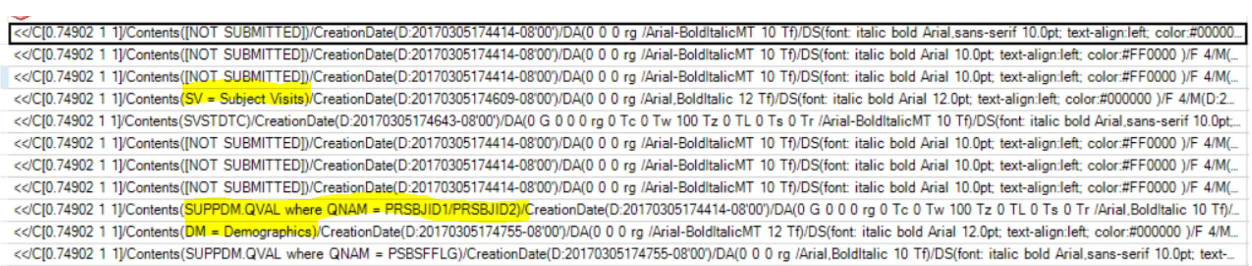
aCRF is a PDF file. In order to read the SDTM annotations into SAS datasets, we first need to output all the SDTM annotations into a forms data format (FDF) file which is a formatted ASCII file. We can then use the following SAS code to read the FDF file into SAS datasets.

```
data acrf_0;
   length conts0 $ 32767;
   infile inacrf lrecl=32767;
   input;
   conts0=_infile_;
run;

data acrf_01;
  retain conts objno;
  length conts $32767;
  set acrf_0;
  if compress(scan(strip(conts0), 2, ' '))='0' and 
upcase(compress(scan(strip(conts0), 3, ' ')))='OBJ' then do;
     conts='';
       objno=input(scan(strip(conts0), 1, ''), best.);
  end;
  else if objno not in (., 1) and upcase(compress(conts0)) ne 'ENDOBJ' 
then  conts=catx (' ', conts, conts0);
  if upcase(compress(conts0))='ENDOBJ' and conts ne '';
  keep conts objno;
run;
```

EXTRACT ANNOTATIONS

The SDTM annotations are embedded in the text of FDF file. The text itself is very messy. However, we can still locate these SDTM annotations based on certain patterns. Figure 3 is an example. From these texts, we can still find the SDTM annotations. After reading FDF file into a SAS dataset, we can then use the following SAS regular expression to extract these annotations and related attributes, such as page numbers, background colors, and assign them to related variables.



```
<</C[0.74902 1 1]/Contents([NOT SUBMITTED])/CreationDate(D:20170305174414-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial,sans-serif 10.0pt; text-align:left; color:#00000...
<</C[0.74902 1 1]/Contents([NOT SUBMITTED])/CreationDate(D:20170305174414-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial 10.0pt; text-align:left; color:#FF0000 )/F 4/M(...
<</C[0.74902 1 1]/Contents([NOT SUBMITTED])/CreationDate(D:20170305174414-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial 10.0pt; text-align:left; color:#FF0000 )/F 4/M(...
<</C[0.74902 1 1]/Contents(SV = Subject Visits)/CreationDate(D:20170305174609-08'00')/DA(0 0 0 rg /Arial,BoldItalic 12 Tf)/DS(font: italic bold Arial 12.0pt; text-align:left; color:#000000 )/F 4/M(D:2...
<</C[0.74902 1 1]/Contents(SVSTDTC)/CreationDate(D:20170305174643-08'00')/DA(0 G 0 0 0 rg 0 Tc 0 Tw 100 Tz 0 TL 0 Ts 0 Tr /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial,sans-serif 10.0pt;...
<</C[0.74902 1 1]/Contents([NOT SUBMITTED])/CreationDate(D:20170305174414-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial 10.0pt; text-align:left; color:#FF0000 )/F 4/M(...
<</C[0.74902 1 1]/Contents([NOT SUBMITTED])/CreationDate(D:20170305174414-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial 10.0pt; text-align:left; color:#FF0000 )/F 4/M(...
<</C[0.74902 1 1]/Contents([NOT SUBMITTED])/CreationDate(D:20170305174414-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 10 Tf)/DS(font: italic bold Arial 10.0pt; text-align:left; color:#FF0000 )/F 4/M(...
<</C[0.74902 1 1]/Contents(SUPPDM.QVAL where QNAM = PRSBJID1/PRSBJID2)/CreationDate(D:20170305174414-08'00')/DA(0 G 0 0 0 rg 0 Tc 0 Tw 100 Tz 0 TL 0 Ts 0 Tr /Arial,BoldItalic 10 Tf)/...
<</C[0.74902 1 1]/Contents(DM = Demographics)/CreationDate(D:20170305174755-08'00')/DA(0 0 0 rg /Arial-BoldItalicMT 12 Tf)/DS(font: italic bold Arial 12.0pt; text-align:left; color:#000000 )/F 4/M...
<</C[0.74902 1 1]/Contents(SUPPDM.QVAL where QNAM = PSBSFFLG)/CreationDate(D:20170305174755-08'00')/DA(0 0 0 rg /Arial,BoldItalic 10 Tf)/DS(font: italic bold Arial,sans-serif 10.0pt; text-...
```

Figure 3. SDTM Annotations in FDF File

```
data acrf_02;
  retain re1 re2;
  length colvar $200 anno $2000;
  set acrf_01;
  if _N_=1 then do;
     re1 = prxparse("/(\/Page +)(\d+)/i");
     re2 = prxparse("/^\<\<\/(C\[.+\])\/Contents/i");
  end;
  if prxmatch(re1,conts) then do;
     pageno = input(prxposn(re1, 2, conts), best.);
  end;
  if prxmatch(re2,conts) then do;
     colflg=1;
     colvar = prxposn(re2, 1, conts);
  end;
  fpos=index(conts, '/Contents');
  lpos=index(conts, '/CreationDate');
  if fpos ne 0 and lpos ne 0 then anno=strip(substr(conts, fpos+10,
lpos-fpos-11));
  else anno='';
  if index(anno, '\r')>0 then anno=strip(tranwrd(anno, '\r', ' '));
  if colvar ne '' then colvar1=round(input(substr(scan(colvar, 1, '
'), 3), best.), 0.01);
  if colvar ne '' then colvar2=round(input(substr(scan(colvar, 2, '
'), 1), best.), 0.01);
  if colvar ne '' then colvar3=round(input(substr(scan(colvar, 3, '
'), 1, length(scan(colvar, 3, ' '))-1), best.), 0.01);
  keep anno colvar colvar1 colvar2 colvar3 pageno objno;
run;
```

MAP ANNOTATIONS TO DOMAINS

In order to identify potential issues with SDTM annotations, we need to associate each valid SDTM annotation with its domain. As shown in Figure 1, the domain annotations are also included in the aCRF, and are usually located at the top of CRF pages. Therefore, we can assign all the SDTM annotations to the domains located on the same CRF pages. There are also two special cases we need to handle. In some CRF pages, there might be more than one domains in one page. In this case, we also need to use the background colors of the SDTM annotations to match the annotations with their domains. The second case is that the annotations with a single domain might be on more than one CRF pages. In this case, we need to carry forward the previous domains in order to map them.

```
data acrf_05;
  length variable $200 domain $10;
  set acrf_04;
  if index(anno, "=") then do;
     variable=strip(scan(anno, 1, "="));
      fposw=index(upcase(variable), 'WHEN');
     if fposw>0 then variable=strip(substr(variable, 1, fposw-1));
     if compress(upcase(variable)) in (&domain) then
domain=compress(upcase(variable));
```

```
  end;
  else variable=anno;
  keep domain variable anno_org anno colvar colvar1 colvar2 colvar3
pageno objno;
run;

********check irregular annotations*********;
data acrf1 chk_irreg_anno (keep=anno_org pageno);
    set acrf_05;
     if index(variable, '/')=0 and length(variable)>8 or variable=''
then output chk_irreg_anno;
    else output acrf1;
run;

proc sort data=acrf1 out=acrf2 nodupkey;
   by pageno colvar1 colvar2 colvar3 domain variable anno;
run;

/*obtain domain name*/
data domain;
  set acrf2;
  where domain ne '';
run;

proc sort data=domain; by pageno colvar1 colvar2 colvar3; run;

data acrf3;
  merge acrf2 (in=a rename=(domain=domain_o)) domain (in=b keep=pageno
colvar1 colvar2 colvar3 domain);
  by pageno colvar1 colvar2 colvar3;
  if a;
run;

proc sort data=acrf3; by colvar1 colvar2 colvar3 pageno; run;

data acrf4;
  retain domain;
  set acrf3 (rename=(domain=domain_o1));
  by colvar1 colvar2 colvar3 pageno;
  if domain_o1 ne '' then domain=domain_o1;
  if variable='STUDYID' then domain='';
  if domain_o='';
  keep domain variable pageno domain_o domain_o1 anno anno_org objno;
run;
```

CATEGORIZE ANNOTATIONS

We can put the SDTM annotations into different categories. This will help us locate the errors in the annotations, because different categories have different requirements on the annotations.

The following are some of the main categories of SDTM annotations:

- SDTM domain annotations
- SDTM variables in parent domains
- SDTM variables in supplemental domains
- TESTCD/TEST related annotations
- Not Submitted annotations
- Notes annotations

Take “SDTM domain annotations” as an example. These are the annotations to indicate the domain names. For this type of annotations, the annotations should have a length of 2, and the annotations should be a valid SDTM domain names.

If the annotations are “Not Submitted” annotations or “Notes” annotations, we can then ignore these annotations.

IDENTIFY ANNOTATIONS WITH ERRORS

After the steps mentioned above, we can then define a number of pre-specified rules and check the SDTM annotations against these rules. The annotations that do not follow the rules will be identified as the annotations with errors and will be put into reports. We can define these rules based on the requirements on the SDTM annotations. The following are some examples of these rules:

- The lengths of domain names should be 2
- The domain names should be valid SDTM domain names
- The lengths of the SDTM variable names should not be longer than 8
- The SDTM variables should have the correct domain names
- The SDTM variable names should be valid
- The SDTM annotations should have required formats

The following are some sample codes to identify the SDTM annotations with errors

```
********check irregular annotations*********;
data acrf1 chk_irreg_anno (keep=anno_org pageno);
    set acrf_05;
     if index(variable, '/')=0 and length(variable)>8 or variable=''
then output chk_irreg_anno;
    else output acrf1;
run;


*************check domain*************;
data chk_domain;
  set acrf4;
  if domain='APMH' and variable not in (&apmhvar) then output;
  else if domain='DM' and variable not in (&dmvar) then output;
  else if (domain not in ('APMH' 'DM') and domain ne substr(variable,
1, 2)) and variable ne 'STUDYID' then output;
run;
```

```
**************check crf page numbers of annotations********;
data chk_anno_pages;
  set acrf8;
  var2=compress(substr(compress(variable), 1,2));
  if length (variable)>8 or var2 ne compress(dataset) or variable in
(&err_var);
  if dataset='APMH' and variable in (&apmhvar) then delete;
  if dataset='DM' and variable in (&dmvar) then delete;
  if index(variable, 'STUDYID') then delete;
  keep dataset variable origin anno anno_org;
run;


********check suppxx annotations**********;
data chk_supp;
  retain variable pageno anno;
  set supp1;
  if index(variable, 'SUPP')=0 or variable ='' or
length(compress(variable)) ne 11;
  keep variable pageno anno;
run;


*******check relrec annotations*******;
data relrec1 chk_relrec (keep=anno pageno);
  length variable $200;
  set relrec;
  if index(upcase(anno), 'RELREC')>0 and index(upcase(anno),
'BETWEEN')>0 then do;
     variable=anno;
      output relrec1;
  end;
  else do;
     variable='';
      output chk_relrec;
  end;
run;


******check value level annotations**********;
data vl1 chk_value_level;
  length tmp1 tmp2 tmp3 cond $2000;
  set vl;
  dataset=domain;
  tmp1=strip(substr(anno, index(upcase(anno), 'WHEN')+4));
  tmp2=strip(scan(compress(tmp1), 1, '='));
  tmp3=strip(scan(compress(tmp1), 2, '='));
  cond=compress(dataset||'.'||tmp2||'.EQ.'||tmp3);
  if dataset='' or cond='' or variable='' or length (variable)>8 or
index(tmp1, '=')=0 or length(tmp2)>8 or
index(compress(upcase(anno_org)), '=NOTSUBMITTED')>0 then output
chk_value_level;
```

```
  else output vl1;
  keep dataset variable cond pageno anno anno_org objno tmp1 tmp2
tmp3;
run;


**********check 'where' annotations**********;
data suppvl chk_supp_where;
  set supp;
  if index(upcase(anno), 'WHERE')>0 and index(upcase(anno), '=')>0 and
index(upcase(anno), '.QVAL')>0 and index(upcase(anno), 'QNAM')>0 and
substr(compress(upcase(anno)), 1, 4) = 'SUPP' then output suppvl;
  else output chk_supp_where;
run;


*********check SUPPxx value level annotations*********;
data suppvl3 chk_supp_vl;
  set suppvl2;
  if dataset='' or length(dataset) ne 6 or index(dataset, 'SUPP')=0 or
variable ne 'QVAL' or index(tmp1, '=')=0 or tmp2 ne 'QNAM' or
length(tmp3)>8 or index(tmp1, '/')>0 then output chk_supp_vl;
  else output suppvl3;
run;
```

The following are two examples of SDTM annotations with errors identified using this tool. In the left CRF page, the SDTM variable is not correct. The correct variable name related to item 6 should be 'FAORRES' instead of 'FORRES'. In the right CRF page, the variable PRSTDTC is a variable in PR domain rather than FA domain, so the correct background color of this annotation should be yellow instead of blue. These types of errors are not obvious, and we can easily overlook them if the annotations are checked totally manually.

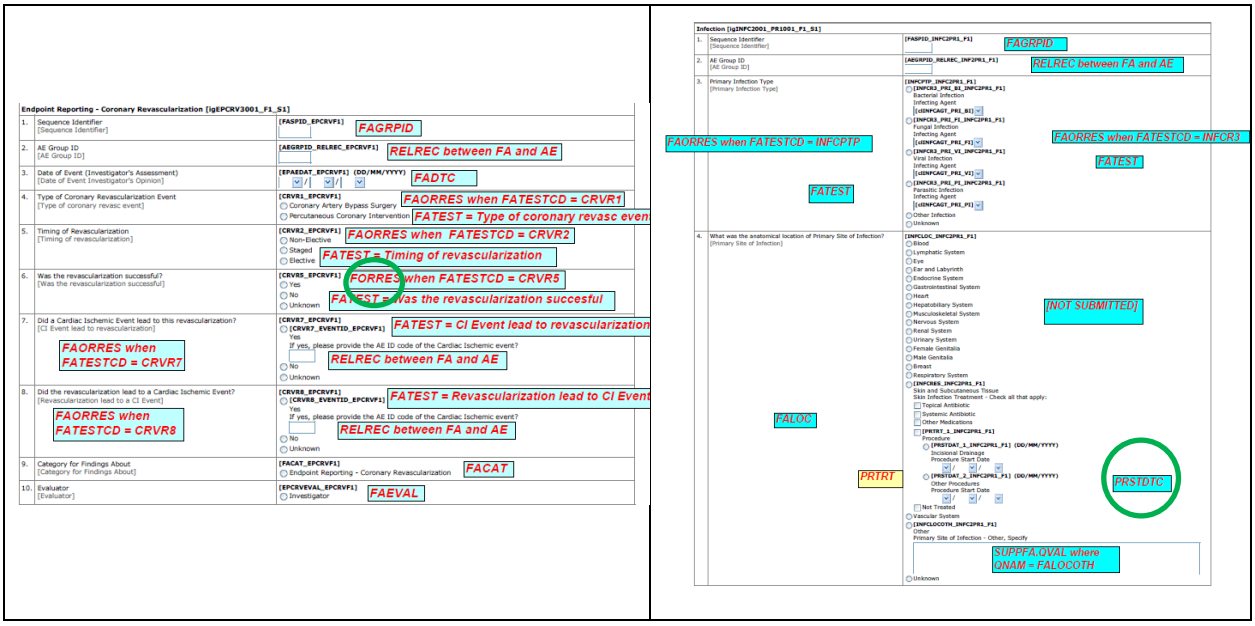


Figure 4. Annotations with Errors

CONCLUSION

aCRF is a very vital file during SDTM development and FDA submission. Therefore, it is important to maintain the correctness of the aCRF. For this purpose, we usually put a lot of efforts to check each annotation to ensure the raw variables are mapped to the correct domains and SDTM variables, the annotations are following required formats, and so on. It would be very time-consuming if we check the SDTM annotations totally manually. In addition, it is very easy to overlook the errors in the SDTM annotations if the errors are not obvious. In fact, some of annotation issues can be checked by programming. This paper proposes a unique way to check these types of issues with SDTM annotations using SAS. The proposed approach is a great addition to the quality check process to improve the efficiency and support to have a quality submission package.

REFERENCE

CDISC. 2013. “Study Data Tabulation Model Implementation Guide (SDTMIG) v3.2.” Accessed November 2, 2017. https://www.cdisc.org.

CONTACT INFORMATION

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

Xingxing Wu
Eli Lilly and Company
wu_xingxing@lilly.com