PhUSE 2016

Paper AD03

QC of SDTM and aCRF using SAS

Rune Østergaard Pedersen, S-Cubed, Copenhagen, Denmark
Niels Both, S-Cubed, Copenhagen, Denmark

ABSTRACT

Processes to document traceability between SDTM, annotated CRF (aCRF), and define.xml are now supported by a variety of tools, but in most cases the processes are still partly manual. Often the creation of the aCRF is separated from the SDTM creation using prior knowledge of the trial data to annotate and iteratively (from the first draft to its final form) update the aCRF as well as the SDTM data. Because the creation of the aCRF is not fully automated, the deliveries can contain errors and data inconsistencies. Therefore, a thorough QC is required to ensure that define.xml and aCRF are fully aligned with the SDTM data. We have developed a program architecture to check consistency between aCRF annotations, define.xml, and SDTM datasets using SAS macros. We will present how to build such a tool, types of errors that can be identified, and options to further automate creation of aCRF and define.xml in order to reduce the risk of errors in define.xml and aCRF altogether.

INTRODUCTION

In clinical trials the study data tabulation model (SDTM) from CDISC has become the standard. When submitting a clinical trial the aCRF is used (among other entry sources e.g. electronic data transfer of external data) as documentation of questionnaires and clinical measurements, and their storage in SDTM datasets. Creation of the final aCRF from SDTM is in many pharmaceutical companies an iterative process, where annotations from other trials can be applied as templates. The annotation can then be copied to the PDF file of the template CRF, or imported in form of a FDF file (see reference 1). These raw annotations are adapted to the present study, using a number of rules for example: colors are used to group the SDTM domains, positions of labels on the CRF should ensure that CRF text is visible, all domains used on a page should be mentioned in the header of the CRF, the relations between domains could be linked using RELREC etc. In addition, sponsor defined rules may be applied to ensure simplicity of for example repeated forms. In many situations, the sponsor can use the reference aCRF file given by CDISC as part of the SDTM v1 & SDTM IG v3.1.1 Metadata Submission Guideline (reference 2). We also apply this aCRF in the current paper paper in our examples.

Two sources of errors exist that need to be covered for any QC-program of the aCRF. Firstly, the aCRF should match data, and secondly the data should match annotations.
In this paper we present a solution on how these two error sources can be identified by the SAS framework using a series of macros to create an output report. One example of a consistency check could be the hierarchical data structure of the SDTM data, which should also be reflected on the aCRF. For example the presence of an –SCAT label on the aCRF implies a –CAT value, or a relation in RELREC SDTM data, must also exist on aCRF and vice versa.

The advantage of such an electronic QC-procedure is that speed and accuracy of the electronic check of the aCRF by far exceeds the human. Also, a computer may reveal potential errors that are not detectable for the eye. For example, annotations that by mistake have been set too far outside the PDF-page, and are invisible. In Figure 1 we show the conceptual framework of the macros applied for the QC-process.

This paper will give the reader:

- An idea on what checks to include when doing QC of the aCRF.
- Tips on how to write efficient and transparent SAS-code.
- Suggestions on how to make annotations transparent and easy to QC.

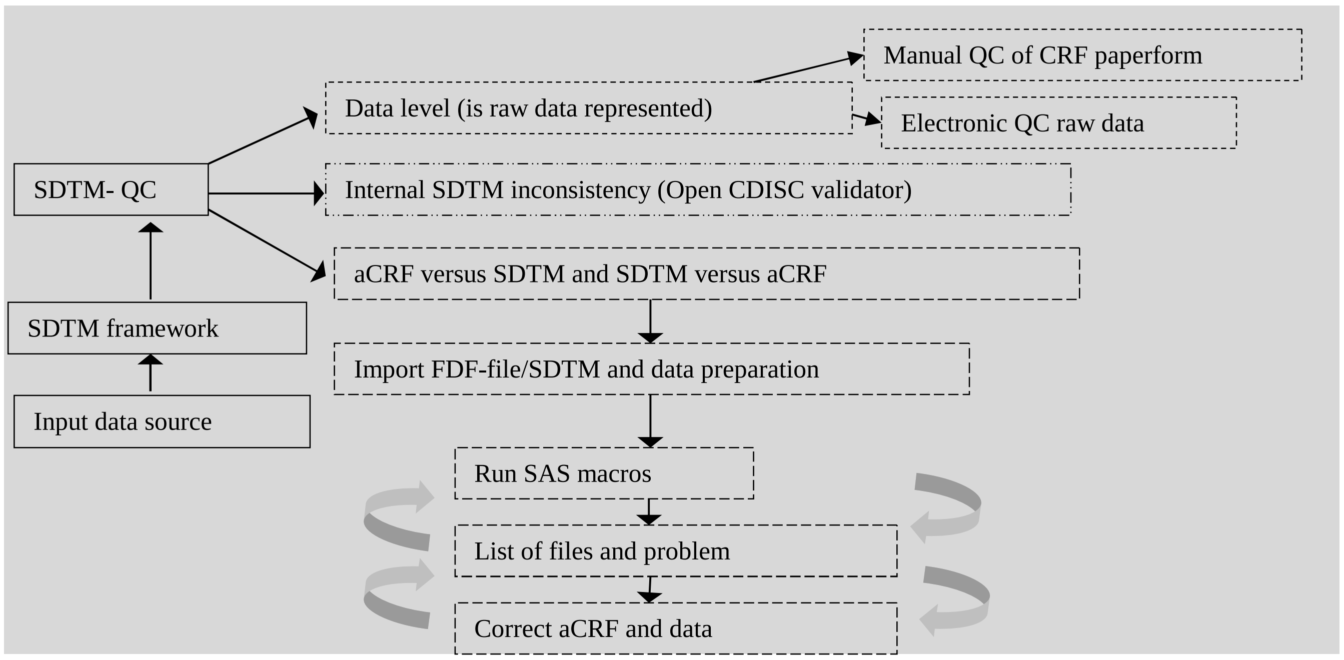
Manual QC of CRF paperform
Data level (is raw data represented)
Electronic QC raw data
SDTM- QC
Internal SDTM inconsistency (Open CDISC validator)
aCRF versus SDTM and SDTM versus aCRF
SDTM framework
Import FDF-file/SDTM and data preparation
Input data source
Run SAS macros
List of files and problem
Correct aCRF and data

Figure 1. The conceptual framework applied when working on SDTM QC. The dashed pattern of the box-border links related steps in the QC-process.

THE STEPS OF THE QC PROCESS

IMPORTING THE ANNOTATED CRF AND SDTM DATA INTO SAS

The annotations are put on as textboxes on the aCRF. Then the annotations can be exported to an FDF-file using for example ADOBE software (see reference 1). The FDF file is imported into SAS using an infile statement in the datastep. Each line is search for the string "/page" in order to identify the pagenumbers that the annotation originates from in the PDF-file. The color code is used to identify "groups" of annotations that belong to the same domain. Some instability in the exact color code number of the FDF-file may be avoided by application of an XFDF-file instead.

After importing the FDF file, the data is cleaned and notes and comments are excluded. For each label a domain annotation and a page number is assigned. The data is then ready for further analysis. In most cases the SDTM data is already available in the SAS data format, and one can simply apply a libname statement. If the data is available in SAS V5 TRANSPORT format the PROC COPY procedure and the XPORT engine may be applied.

MANUAL QC

A fully automated QC is not possible to create. Part of the process needs to be manual. Sometimes it is an advantage to test that mapping is correct by hand before QC, in order to avoid a very long list of errors from the automatic QC program. Some reported errors are, as we shall see later not true errors, but only a result of how the aCRF has been filled out. Furthermore, an aCRF may also contain pages which should be not submitted. In this case the annotator should verify that the "Not Submitted" data is in fact not needed for the submission, and consistent with SDTM (reference 2).

THE DATAFLOW OF THE PROGRAM

In Figure 2 a more detailed sketch of the dataflow is given. The data cleaning takes place before the step "Comparing the two input data sources".The cleaning insures that only data originating from aCRF is compared. For example, it is quite common that laboratory data is not visualised on the aCRF, but is mapped from Electronic Data Transfer (EDC). At this stage we have two datasets - one containing the SDTM data from SAS, and another with the aCRF annotations in a modified format. This allows comparison between the two input sources. Using Proc SQL all that the program does is to find and exclude the common variables, i.e. outputting only the variables not in common, which is the shaded area in the middle of the two overlapping circles on Figure 2. The list of variables actually present in SDTM are created using a call to the – vcolumn dataset in the SASHELP library. The potential variables are then categorised and output into a common Excel report.

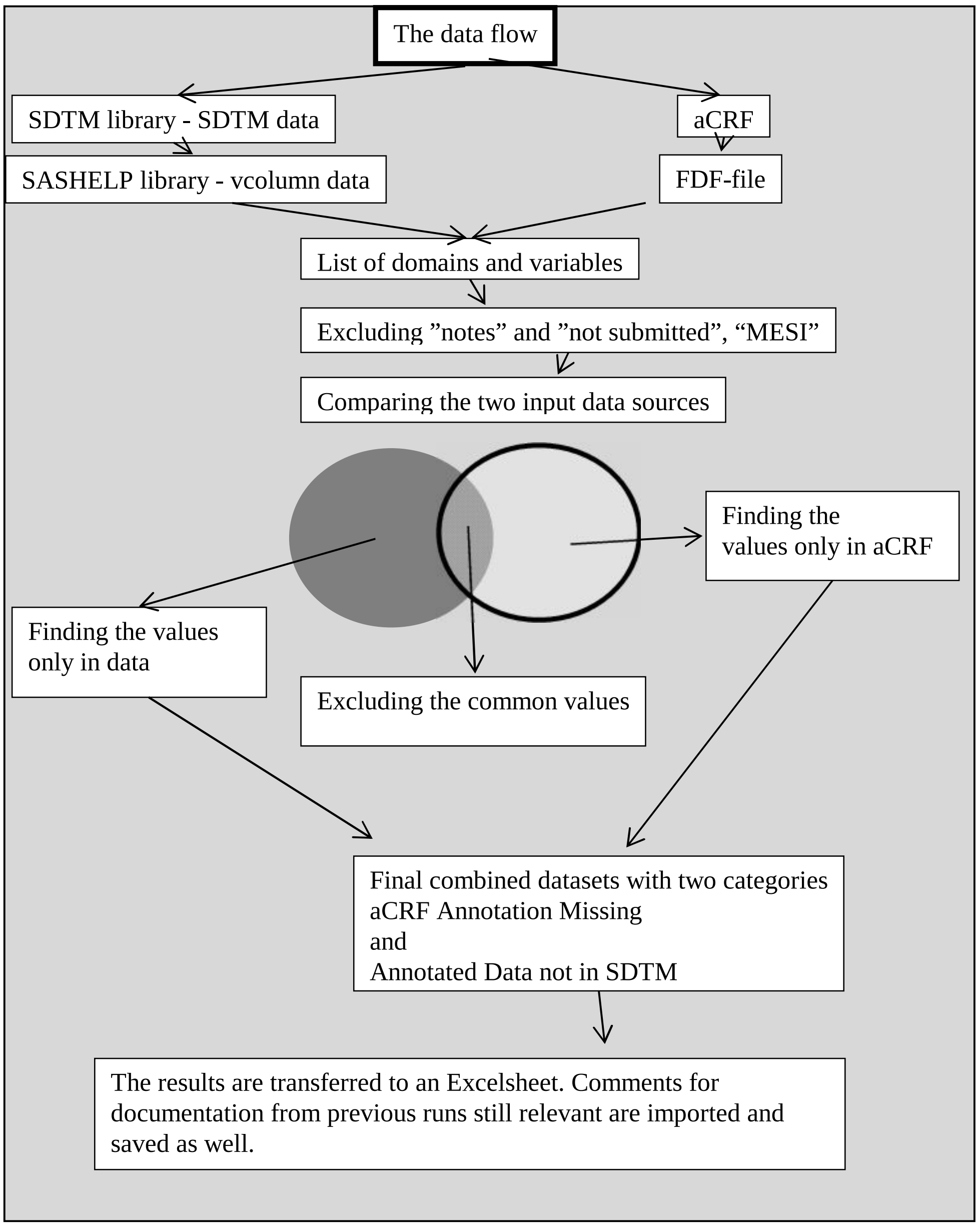
The data flow
SDTM library - SDTM data
aCRF
SASHELP library - vcolumn data
FDF-file
List of domains and variables
Excluding "notes" and "not submitted", "MESI"
Comparing the two input data sources
Finding the values only in aCRF
Finding the values only in data
Excluding the common values
Final combined datasets with two categories aCRF Annotation Missing and Annotated Data not in SDTM
The results are transferred to an Excelsheet. Comments for documentation from previous runs still relevant are imported and saved as well.

Figure 2. The detailed data flow of the SAS macros

UTILIZING THE STRUCTURE OF SDTM

The pyramid in Figure 3 illustrates the hierarchical structure in SDTM. On the top-level the domains are checked to make sure that they exist in data and on aCRF. Inside the data in the SDTM domains the variable level is checked. Similarly particular values of the variables mentioned in data are checked on aCRF e.g. --SCAT,--CAT and --TESTCD and in SDTM.

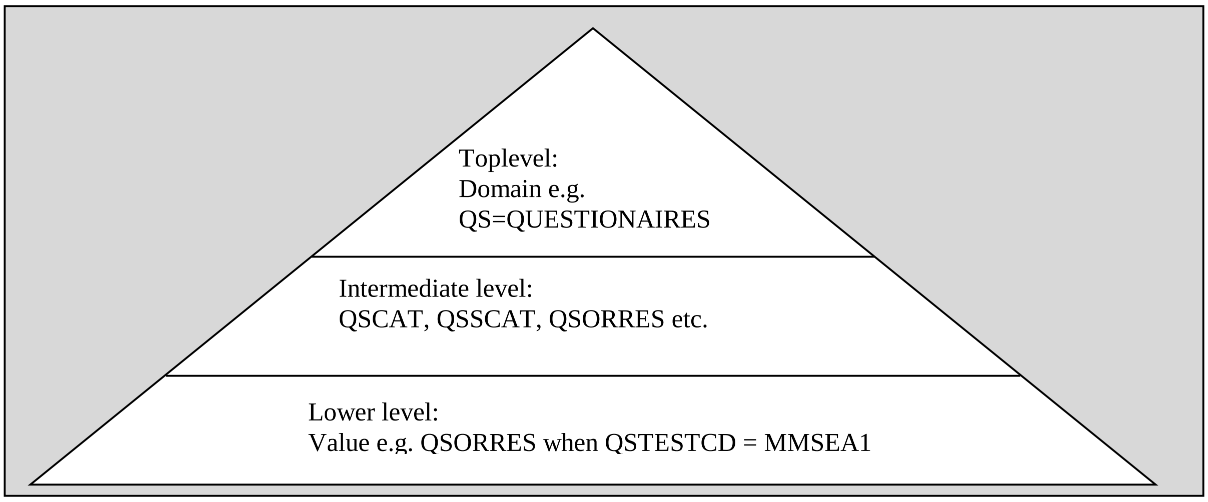
Toplevel:
Domain e.g.
QS=QUESTIONAIRES
Intermediate level:
QSCAT, QSSCAT, QSORRES etc.
Lower level:
Value e.g. QSORRES when QSTESTCD = MMSEA1

Figure 3. The hierarchical structure of the QC-process of SDTM and aCRF.

When debugging the errors found in SDTM and aCRF one should move from the highest level of resolution to the lower level of resolution. For example if a SDTM domain is not present in data, despite it is correctly present on the aCRF, one need to bring the domain in before debugging variables which are obviously also absent in the data, as the entire dataset is missing.

IDENTIFYING THE PROBLEMS BY STANDARDIZING CRF ANNOTATIONS

In order to make the data into a readable format we import the aCRF to SAS using the FDF format. It is important that the annotation of the aCRF follows some systematic business-rules making it possible to implement a systematic QC. In Table 1 we show some of the important characteristics of the aCRF annotation boxes that we implemented. As a general rule we want to make the aCRF linkage to the SDTM data as transparent as possible. For example, we chose to write the exact domains linked when using the RELREC domain. This allows a test to see if the relation is in fact represented in RELREC. The design of the annotation applied for identification is set under the "Rule of Inclusion". Some labels, though not all also have a rule of exclusion. For example EDC data is often applied as laboratory data, though it is not on the aCRF. In the rule of exclusion we have written data driven in parenthesis in Table 1 to indicate that exclusion is done on excess data in SDTM, and not excess annotation. For annotations that are not in the categories not submitted, notes or replications we always expect to find a match in the data. However, some few exceptions in automated list are expected in for example supplemental qualifiers.

Table 1

Level	Rule of inclusion	Rule of exclusion (data driven)
Domain	Grouping using colors of domain variable	
	On each page the domain should be mentioned e.g. AE=ADVERSE EVENTS	
Variable	The variable names are left of equal signs. In case of no equal sign the text is a variable. QNAM variables are separated with "-" and numbered successively like QNAM=AEACN1-AEACN6	Origin in the data that is not CRF e.g. EDC, could be adjudication data.

Value	Separation of values is done with “,”. Certain words are used to identify variables. For example LBORRES when LBTESTCD=TOT_CARB. The “when” and “=” are identifiers.	Origin in the data that is not CRF e.g. EDC, could be adjudication data.
Notes and other labels	None	Notes are always initiated with NOTE:, This allows to remove the notes from the SAS dataset originating from the FDF-file. “Not Submitted” labels are excluded.
Replicated forms (paper forms)	Replications of the same forms on the aCRF are written AS PAGE x (where x indicates a number).	
RELREC	The domains in RELREC are identified on the aCRF as RELREC: AE,DS. In case of more than two domains the list is expanded as RELREC: ZZ,YY,XX	None

MAKING EFFICIENT SAS MACRO PROGRAMS
The program used a combination of SAS macro language and the SAS Proc SQL procedure to loop through the data and obtain a list of potential errors. The whole program is made of five main parts. One macro checking the aCRF versus SDTM, one for SDTM versus aCRF, two more programs for the supplemental qualifiers and finally a macro for RELREC. For simplicity, and because the five parts are quite similar we only show a modified version of the macro testing the aCRF versus the SDTM data (macro AcrfvsSDTM), and the RELATION macro for RELREC. In the code examples, we explain the steps as we go along, with reference to the levels of the pyramid in Figure 3.

```
%Macro AcrfvsSDTM;
/* First step is a data cleaning step*/
   data ann_dom;
         set Raw_annotation(where=(compress(annotation) not
   in("QVAL","","NOTSUBMITTED") and not index(original_annotation, "RELREC") and
            domain ne ""));
/*Removing labels originating from aCRF that are not in SDTM*/
    label_SDTM=compress(label_SDTM,'"');
    original_annotation=compress(original_annotation,'"');
    /*This contains the original annotation used in the final excel report*/
   run;

/*Making a count used for looping over all domains and their variables*/
   proc sql noprint;
      select count(*) into:obs
         from ann_dom;
   quit;

   *This piece of the code makes a list of sdtm variables for a particular domain;
   proc datasets lib=work noprint;
      delete temp:;
   quit;

%*Making SAS macro variables from the labels on aCRF

   %do i=1 %to &obs.;

      data _null_;
         set ann_dom(obs=&i. firstobs=&i.);
         call symputx('domain',domain);
         call symputx('sdtmvar',annotation);
         call symputx('sdtmlabel',label_SDTM);
         call symputx('original_annotation',original_annotation);
         call symputx('page',strip(pageno));
      run;

%* Check if domain is present in data, this is the Toplevel in Figure 3;
      %let EXIST&domain=%sysfunc(exist(sdtm.&domain));

%if &&EXIST&domain.. %then %do; /*This is a loop on the Intermediate level in Figure
3*/

%*Make a list of the variables in the SDTM data for the current domain;
     proc sql noprint;
        select name into: varlist separated by ' '
           from dictionary.columns
              where libname eq "%upcase(SDTM)"
                 and memname eq "%upcase(&domain.)"
                 and memType eq 'DATA';
     quit;

/*Making a list of numeric/character variables –again the Intermediate level */
     proc sql noprint;
        create table num_char as
           select memname, type , name
              from dictionary.columns
                 where libname eq "%upcase(SDTM)"
                    and memname eq "%upcase(&domain.)"
                    and memType eq 'DATA';
     quit;

%*Check if the SDTM variable obtained from the aCRF exists in the list of variables
from the SDTM data;
            %let EXISTLIST&domain = %sysfunc(ifc(%index(&VarList,&sdtmvar.)
               ,%nrstr(1)
               ,%nrstr(0)
               ));

%*When the SDTMLabel is empty check only for the variable name;
            %if %bquote(&sdtmlabel.) eq %then %do;
                  %if not &&EXISTLIST&domain. %then %do;

%*SDTM variable does not exist in data;
      data temp&i.;
        LENGTH Original_annotation $ 200 RDOMAIN $ 20 qnam $ 200 pagenum $ 3 problem $
        500 checkdesc $ 200;
        Original_annotation="&original_annotation";
        qnam="&sdtmlabel";
        rdomain="&domain";
        pagenum=left(put(&page.,3.));
        problem="&sdtmvar not present";
        checkno=6;
        checkdesc="Find if a variable annotated on the aCRF does not exist in the SDTM
        data”;
      run;

                     %end;
               %end;

%*If the SDTMlabel is not empty, we look "into" the SDTM variable to find if the value
is there (eg for LBTESTCD, we look for the specified – this is the Lower level in
Figure 3. value from the aCRF, LBTESTCD=XXX);
            %else %if %bquote(&sdtmlabel.) ne %then %do;
%*Check that the SDTM var exists;
    %if &&EXISTLIST&domain %then %do;

%*This part of the code removed problems with num and char variables;
       data _null_;
         set num_char;
```

```
          where name="&sdtmvar";
          call symput("num_char",strip(upcase(type)));
        run;

                         %put &num_char;

        proc sql noprint;
           select count(*) into:valFound from sdtm.&domain
             %if &num_char="NUM" %then %do;
             where &sdtmvar = %nrbquote(&sdtmlabel);
             %end;
             %if &num_char="CHAR" %then %do;
             where &sdtmvar = "%nrbquote(&sdtmlabel)";
             %end;
        quit;

        %if not &valFound. %then %do;

/*Making an output datasets*/

    data temp&i.;
        LENGTH Original_annotation $ 200 RDOMAIN $ 20
        qnam $ 200 pagenum $ 3 problem $ 500
        checkdesc $ 200;
         Original_annotation="&original_annotation";
         qnam="&sdtmlabel";
         rdomain="&domain";
         pagenum=left(put(&page.,3.));
         problem="&sdtmvar. present but no value of
         &sdtmlabel. exists in data";
         checkno=7;
         checkdesc="Find if a variable with value (eg XXTESTCD=TEST) is annotated on
        the aCRF, but that value does not exist in the SDTM data";
     run;
                            %end;
                      %end;
                %end;
          %end;
%*If whole domain is missing in sdtm data;

      %else %do;

/*Making the final dataset with the different variables*/

    data temp&i.;
       LENGTH Original_annotation $ 200 RDOMAIN $ 20 qnam $ 200 pagenum $ 3 problem $
       500 checkdesc $ 200;
        Original_annotation="&original_annotation";
        qnam="&sdtmlabel";
        rdomain="&domain";
        pagenum=left(put(&page.,3.));
        problem="Missing SDTM-&domain. domain";
        checkno=8;
        checkdesc="Find if a domain is annotated on the aCRF, but that domain does not
       exist in the SDTM data";
    run;
         %end;

 %if %sysfunc(exist(work.temp&i.)) %then %do;

/*Appending the data from the different loops*/

    proc append base=tempAll data=temp&i. force;
    run;
         %end;
   %end;

   %if not %sysfunc(exist(work.tempAll)) %then %do;

    data tempAll;
```

```
        attrib pagenum length= $ 3 RDOMAIN length= $ 20 qnam length= $ 200;
        stop;
     run;

       %end;

   data ACRFvsSDTM;
      set tempAll;
      pag=input(pagenum,3.);
   run;

   proc sort data=ACRFvsSDTM;
      by pag rdomain qnam;
   run;

%mend AcrfvsSDTM;
```

The RELATION macro follows a similar structure than the code above. Only for RELREC we have a grouping variable linking observations called relid. Thus, for each relid we obtain a list of domains in the data that exist, i.e. domain names e.g. AE:DS. From the aCRF our annotation specifications seen in Table 1 specifies the linkage and this can be read in from the FDF-file. In the FDF data we search for labels containing the word RELREC, and we know that this word should be followed by domains.

```
%macro RELATION;
   *List of annotated domain links for RELREc AND Get them in alphabetical order;
   data relrec1;
      set RAW_ANNOATION;
      where index(original_annotation,"RELREC") and annotation ne "";
      comboN=_n_;
      dom=strip(scan(annotation,1,","));
      output;
      dom=strip(scan(annotation,2,","));
      output;
   run;

   proc sort data=relrec1;
      by comboN dom;
   run;

   data relrec2;
      set relrec1;
      by comboN;
      length combo $10;
      retain combo "";
      if first.comboN then
         combo=strip(dom);
      else combo=strip(combo)||","||strip(dom);

      if last.comboN then output;
   run;

   proc sql noprint;
      select count(*) into:relAnnoObs
         from anno_domains
            where index(original_annotation,"RELREC");
              select count(*) into:relObs
                from relrec2;
   quit;

   %* Check if RELREC domain is present in data;
   %if not %sysfunc(exist(sdtm.RELREC)) %then %do;
    data RELREConCRF_notDATA;
       LENGTH RDOMAIN $ 20 problem $ 500 checkdesc $200;
       rdomain="RELREC";
       problem="Missing SDTM-RELREC domain";
       checkno=12;
       checkdesc="Find if RELREC domain is missing from SDTM data";
      run;
      %end;
```

```
%else  %do;
 data RELRECinCRF_notDATA;
    LENGTH RDOMAIN $ 20 problem $ 500;
    rdomain="";
    problem="";
    delete;
 run;
      %* If we have a RELREC domain, make sure there is some kind of annotation;
      %if not &relAnnoObs. %then %do;

 data RELRECinDATA_notCRF;
    LENGTH RDOMAIN $ 20 problem $ 500 checkdesc $200;
    rdomain="RELREC";
    problem="RELREC exists, but is not annotated on aCRF";
    checkno=13;
    checkdesc="Find if existing RELREC domain has not been annotated on aCRF";
  run;
        %end;

%*If RELREC has been annotated, but without the domain links, we can’t check more;
      %else %if not &relObs. %then %do;

  data RELRECinDATA_notCRF;
    LENGTH RDOMAIN $ 20 problem $ 500;
    rdomain="RELREC";
    problem="RELREC exists, but annotation on aCRF does not include domain
    links";
    checkno=14;
    checkdesc="Find if RELREC annotations on aCRF do not include the domain link
    info (eg RELREC: AE, CM)";
  run;
        %end;

%* If we have annotations, check that the domain combos annotated, exist in the
      data;
      %else %do;
* Make a list of the unique combos in the RELREC data;
  proc sort data=sdtm.RELREC out=relIDs nodupkey;
    by relid rdomain;
  run;

  data relCombos;
     set relIDs;
     by relid rdomain;
     length combo $10.;
     retain combo "";
     if first.relid then combo=strip(rdomain);
     else combo=strip(combo)||","||strip(rdomain);
     if last.relid then output;
  run;

  proc sort data=relCombos out=uCombos nodupkey;
     by combo;
  run;

       * Find RELREC combos annotated, but not in data;
  proc sql noprint;
     create table RELRECinCRF_notDATA as
        select a.original_annotation,
           a.combo as qnam,
           "RELREC" as rdomain,
           "No link in RELREC for domains annotated on aCRF" as problem,
           pageno as pag,
           15 as checkno,
          "Find if a RELREC domain combination that has been annotated on the  aCRF
             does not exist in the RELREC SDTM domain" as checkdesc
            from relrec2 as a
             where (a.combo not in (select combo from uCombos));
  quit;
```

```
              * Find RELREC combos in data, but not annotated on aCRF;
     proc sql noprint;
        create table RELRECinDATA_notCRF as
           select a.combo as qnam,
              "RELREC" as rdomain,
              "No annotation on aCRF for domains linked in RELREC" as problem,
              16 as checkno,
              "Find if a RELREC domain combination that exists in the RELREC SDTM
                 domain has not been annotated on the aCRF" as checkdesc
              from uCombos as a
               where (a.combo not in (select combo from relrec2));
     quit;
 %end;
%end;
   %mend RELATION;
```

FIVE SMALL CASE STUDIES

In this section, we want to give five small examples of common errors on the aCRF and the error message output by the program. All of these steps can be covered using the principles of the code in the section "Making an efficient SAS macro program", though small modifications may be needed for your own trial. In addition, the output from the program may be studied in Appendix 1.

Example 1: A wrong color has been put on a label.

This error is very common, especially when reusing the annotations from other trials as templates. Often a box color can also differ from the remaining, due to missing domain on the top of the page. Both errors are important to remove in order to insure aesthetic and logical transparency of the aCRF.

Example 2: Often a supplemental qualifier is one in a series. To save space and for simplicity the annotation will often be written in a form like myqnam1-myqnam_nn in suppdomain. Each of the numbers here refer to a radio button that once activated will capture a fixed value ending in the qval variable of the supplemental qualifier. Programming wise a list of qnam from 1 to nn must be made in a loop.

One typical problem detecting errors from this list is false positive errors. For example the selection of supplemental qualifiers from the aCRF need not to contain all possible outcomes from 1 to nn. This needs to be verified by looking into the raw data source. Thus, an automated QC program cannot completely replace human investigation into data, but can give hints on where to look for potential problems.

Example 3: The RELREC domain is used to help the reader understand relations between data from different SDTM domains. As a standard the label RELREC is used on the aCRF. This can advantageously be extended containing information about the actual domains such as RELREC AE, DS. Such a labelling allows - contrary to the simple annotation - that the data is cross checked for relations. Furthermore, we believe it improves the reader's ability to understand the data flow when looking at the aCRF. We use an example where RELREC between adverse events and disposition is present in data but not found on the aCRF, i.e. the label RELREC: DS,AE on page 18 of the annotation (see reference 1 for aCRF).

Example 4: As a standard we recommend to use a label(s) on top of each page with a two letter domain abbreviation e.g.DM=Demographics. In addition, all labels from the same domain should be of same color.

Though one might argue that a form continuing over several pages need not the redundant information of a domain on top, it is technically a more safe way to insure that labels are grouped to the right domain than just using color. In our experience it can lead to errors simply relying on colors are systematic across domains from several pages, especially when the annotation uses templates from other trials. Furthermore, the exactness of the color codes seems to be unstable sometimes in FDF-files leading to errors.

Example 5: As an example of missing record data for a given domain we look at the DATESTCD=DISPAMT for the Drug Accountability domain on page 19 of the aCRF of Reference 1. The example assumes that the DA domain as well as the DATESTCD.

THE FINAL EXCEL REPORT

After isolation of the problems all data is joined together to create the final output report in Excel. The report allows comments to be made as a part of the QC-process. In order to preserve the comments when successively running the program we merge back on previous versions of reports maintaining comments of problems still appearing in the newest report. Such outputs can be used as a part of the documentation and justification of the aCRF and SDTM when submitting a trial.

An example of how an Excel report can look like is given below though slightly modified and is showed in Table 2 of Appendix 1, with link to the examples 1-5 above

CONCLUSION

The SAS based system in submission deliverables has proven to be an efficient and capable method of finding errors and inconsistencies.

The compilation time seems to be only a few minutes on powerful servers used by companies and a bit longer on PC, thus it seems feasible to apply it everywhere. The combination of the SAS macro language, Proc SQL and the SAS datastep procedure seems to be powerful and efficient for ensuring aCRF and SDTM data with a minimum of errors. The fact that that the system checks standardized uniform input means that it should be possible to use the same program in many different companies.

The concept of this kind of check is based on having the aCRFs stored as PDF files and exporting annotations as FDF (or XFDF) files (We refer to this as the "FDF-method"). The authors acknowledge that this method of working with aCRFs is beginning to become a bit outdated. Technology is changing and the trend seems to be that more and more companies strive to use MDR (Metadata Repository) technology, to produce the full aCRF (as well as Define.xml and input to drive SDTM creation). We deem that this newer MDR method is superior to the FDF-method

However, it is the impression that the MDR based concept is not yet matured and used to a great extent. The majority of companies seem to currently either annotate manually, or to use the FDF-method, and therefore for the moment and until the MDR based methods become fully developed, we think that the program we present in this article is a good method to use.

REFERENCES

1. Dirk Spruck, Monica Kawohl: Paper CC02. Using SAS to speed up annotating case report forms in PDF format – Pharmasuc 2004.
2. SDTM V1.1 & SDTM IG V3.1.1. Metadata submission guideline. Available on CDISC.ORG.

CONTACT INFORMATION

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

Rune Østergaard Pedersen
S-cubed APS
Lille Strandstræde 20C 5 th,
DK-1254 Copenhagen K, Denmark
Work Phone: +4529853672
Email: rp@s-cubed.dk or rune.pedersen82@gmail.com

APPENDIX 1

Table 2. Modified output from the SAS program.

Check No	Check Description	Category	Problem	Annotation	Domain	Page	Color	Variable	Value	addnote	comment
1 (Ex1)	Find if a variable has been annotated on a page, but with no matching domain annotation. This could be caused by color differences.	aCRF Annotation Problem	No domain annotated on aCRF	IEDTC		4	C[1.0 1.0 0.0]				The color has been corrected belongs to IE domain. Will be OK after rerun.
2 (Ex2)	Find if a supplemental qualifier variable has been annotated on the aCRF, but is not present in data	Annotated on aCRF, but not in SDTM data	QNAM annotated on aCRF, but not present in supp qual data	RACE1-RACE5 in SUPPDM	SUPPDM	6		RACE4			Ok, the value has never been filled out on aCRF
3 (Ex3)	Find if a RELREC domain combination that exists in the RELREC SDTM domain has not been annotated on the aCRF	In SDTM data, but not annotated on aCRF	No annotation on aCRF for domains linked in RELREC	No annotation on aCRF for domains linked in RELREC	RELREC	18		DS,AE			The annotation has been updated to reflect the data.
4 (Ex4)	Find if a variable has been annotated on a page, but with no matching domain annotation. This could be caused by color differences	In SDTM data, but not annotated on aCRF	No domain annotated on aCRF								The domain has been added
4(Ex 5)	Find if a value of an existing grouping variable (eg XXTESTCD of value TEST) has not been annotated on the aCRF	In SDTM data, but not annotated on aCRF	No annotation on aCRF for this value		DA			DATESTCD	DISPAMT		The testcd does origin from the CRF, the annotation has been corrected.