PharmaSUG 2019 - Paper AP-125

Auto-Annotate Case Report Forms with SAS® and SAS XML Mapper

Yating Gu, Seattle Genetics, Inc., Bothell, WA

ABSTRACT

A common approach to annotating blank case report forms (CRFs) is to manually create and draw text boxes in PDF, and then type in the annotations that document the location of the data with the corresponding names of the SDTM datasets and the names of those variables included in the submitted datasets. The CRF design is similar across many studies, and there are usually only minor updates in different CRF versions within studies, however, it is very time consuming to manually copy and paste annotations across CRFs. Therefore, an approach that could help auto-annotate a new blank CRF based on an existing annotated CRF could save a tremendous amount of time and effort. This paper introduces a method to import annotations from an annotated CRF to SAS using SAS XML Mapper, manage annotations, and incorporate annotations back into a new blank CRF using SAS. Some relevant programming details and examples will be provided in this paper.

INTRODUCTION

A case report form (CRF) is a printed or electronic document designed to collect the protocol-required information to be reported by the Sponsor on each subject in a clinical trial. An annotated CRF (aCRF) is a blank CRF with the annotations that document the location of the data with the corresponding names of the datasets and the names of those variables included in the submitted SDTM (Study Data Tabulation Model). Figure 1 shows an example of an annotated Demographics CRF page.

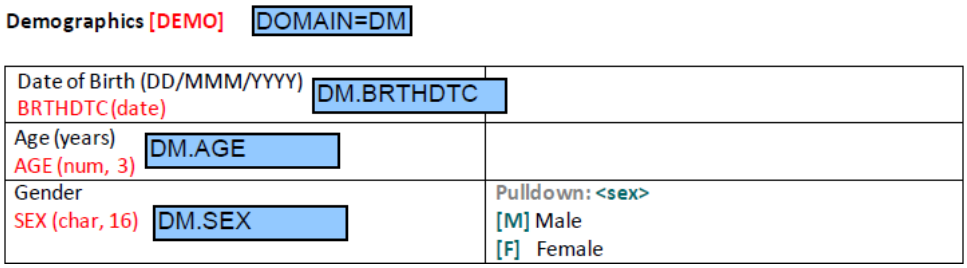

Figure 1. Annotated CRF Reflects How Variables Collected in CRF are Mapped into SDTM Datasets

The annotation box was manually created in the PDF file and adjusted to the proper position. An aCRF can help the programmers quickly identify the origin of variables in the submitted SDTM datasets while they map variables captured from CRF into SDTM. As there may be protocol amendments and new CRF forms added to the study, it would be very time consuming to manually copy all existing annotations from the old aCRF and paste to the new blank CRF. So, this paper introduces a way to help programmers auto-annotate a new blank CRF using an existing aCRF, which could save a lot of effort. Figure 2 explains the work flow of the auto annotation process.

Figure 2. Work Flow of Auto-Annotate New Blank CRF Using an Existing aCRF

Both XML Forms Data Format (XFDF) files and Forms Data Format (FDF) files can be used as containers for annotations that are separate from the PDF file. The annotations are exported and stored in an XFDF file. The annotations are then captured from the XFDF file to SAS datasets by SAS XML Mapper, which automatically analyzes the structure of the XFDF file and creates basic metadata. The annotations can be modified and built from SAS datasets and exported to an FDF file, which can then be imported back to a new blank CRF.

METHODOLOGY

EXPORT ANNOTATIONS FROM AN ACRF PDF FILE TO A XFDF FILE

The aCRF annotated by the programmer is saved in a PDF file. Figure 3 shows the process to export the aCRF annotations and save in a stand-alone XFDF file.

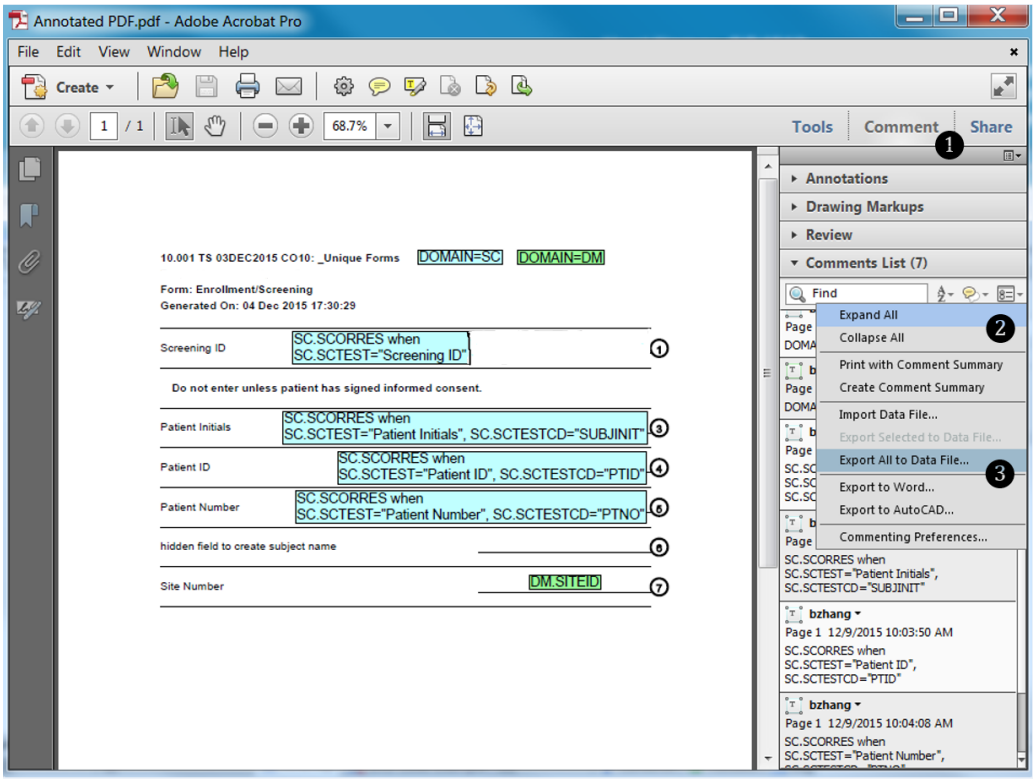

Figure 3. Steps of Exporting Annotations from an aCRF PDF File

Firstly, click on “Comment”. From the drop down menu within “Comments List”, select “Export All to Data File…”. Then choose XFDF as the file type so that the annotations can be saved in a separate XFDF file.

CONVERT XFDF FILE TO SAS DATASETS WITH SAS XML MAPPER

Next, with the help of SAS XML Mapper, the annotations in XFDF file can be migrated to SAS datasets. Figure 4 shows an example of the SAS XML Mapper panel.

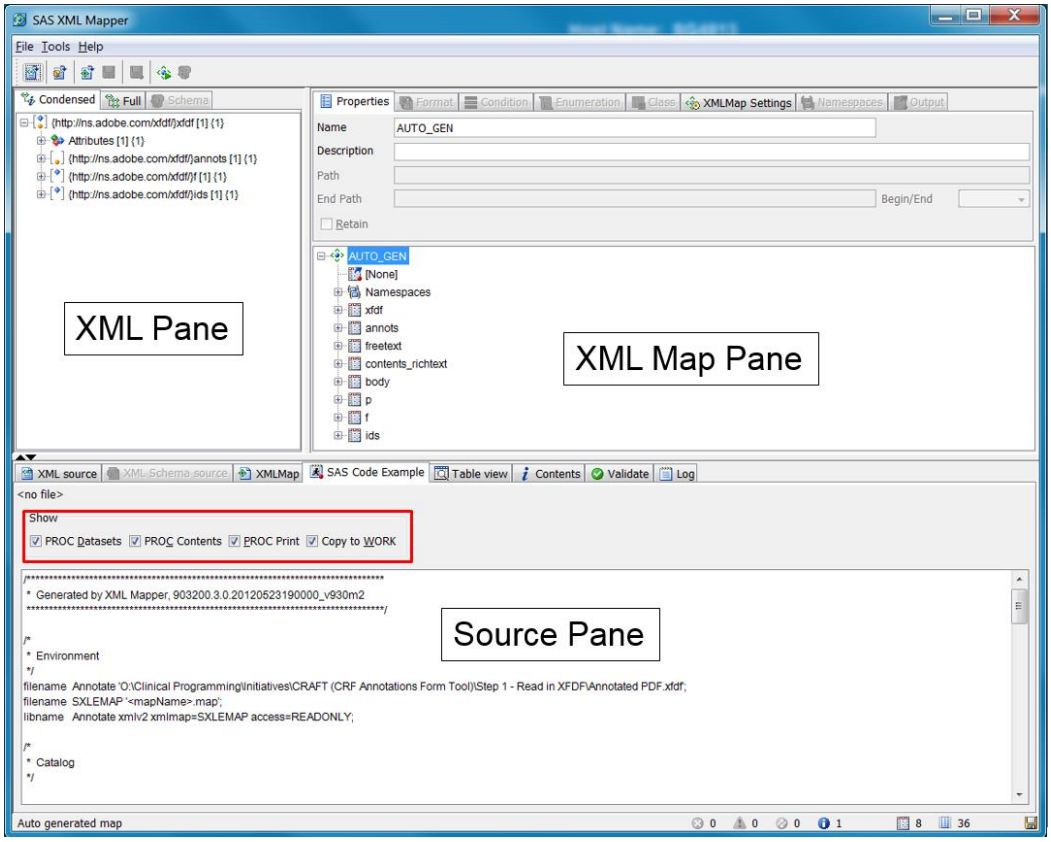

Figure 4. SAS XML Mapper Automatically Analyzes the Structure of XFDF File and Generates Basic Metadata

To import the XFDF file, click on “File” and select “Open XML…” from the drop down menu. Then by clicking on “Tools” and “AutoMap use XML”, it automatically converts the XFDF file to SAS datasets (displayed in XML Map Pane) that contain basic metadata for annotations. It also automatically creates the SAS code (displayed in Source Pane) which can be copied and saved in SAS. The first FILENAME statement defines the reference “Annotate” to the physical location of the XFDF file. The second FILENAME statement defines the reference “SXLEMAP” to the physical location of the XML MAP file. The third LIBNAME statement uses the reference “Annotate” to reference the XFDF file. It also specifies the XML engine and uses the reference “SXLEMAP” to reference the XML MAP file. Finally, close the SAS XML Mapper and save it in the MAP file.

MERGE MULTIPLE SAS DATA SETS INTO A SINGLE ONE AND KEEP NECESSARY VARIABLES

Open SAS and run the SAS code generated from SAS XML Mapper; it will read in all the SAS datasets that contain annotation contents as well as attributes. The variables that are necessary for this annotation migration process are background color of annotation box (freetext_color), CRF page number (freetext_page), annotation box position (freetext_rect), attribute of annotation text including font style/size/alignment/color (defaultstyle) from “freetext” dataset and annotation contents (p) from “p” dataset. The rest of the datasets can be used as a bridge to connect “freetext” and “p”. All datasets have some ordinal variables that can act as key variables to merge all these datasets into one dataset. Table 1 is an example of the combined single dataset that contains all the selected variables.

freetext_page	defaultstyle	p	freetext_rect	freetext_color
4	font: Arial 12.0pt; text-align:left; color:#000000	Domain=DM	151.474000,702.753000,229.474000,721.537000	#93C9FF
4	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.RDPENDTC	218.919000,668.363000,333.519000,686.545000	#93C9FF
5	font: Arial 12.0pt; text-align:left; color:#000000	Domain=DM	228.056000,704.062000,306.056000,722.847000	#93C9FF
5	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.SUBJID	1.345980,666.562000,75.890700,688.346000	#93C9FF
5	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.SUBJID	154.146000,610.925000,290.946000,632.710000	#93C9FF
6	font: Arial 12.0pt; text-align:left; color:#000000	DOMAIN=DM	179.244000,704.318000,257.994000,719.318000	#93C9FF
6	font: Arial 12.0pt; text-align:left; color:#000000	DOMAIN=DS	177.465000,727.751000,258.465000,743.012000	#C4ECFF
6	font: ArialMT 12.0pt; text-align:left; color:#000000	DOMAIN=MI	175.201000,750.409000,256.801000,771.409000	#DFDFFF

Table 1. Single Dataset that Contains all the Necessary Variables for Annotations

In order to later fit these variables into the FDF code, the position indexes within variable freetext_rect need to be separated by blank spaces instead of commas. In addition, the color hex code within variable freetext_color needs to be converted to the RGB code. This website provides a handy reference for the color conversion from hex code to RGB code: https://convertingcolors.com/hex-color-FFFF00.html. The updated results are displayed in Table 2.

freetext_page	defaultstyle	text	box	color
4	font: Arial 12.0pt; text-align:left; color:#000000	Domain=DM	151.474000 702.753000 229.474000 721.537000	0.58 0.79 1
4	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.RDPENDTC	218.919000 668.363000 333.519000 686.545000	0.58 0.79 1
5	font: Arial 12.0pt; text-align:left; color:#000000	Domain=DM	228.056000 704.062000 306.056000 722.847000	0.58 0.79 1
5	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.SUBJID	1.345980 666.562000 75.890700 688.346000	0.58 0.79 1
5	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.SUBJID	154.146000 610.925000 290.946000 632.710000	0.58 0.79 1
6	font: Arial 12.0pt; text-align:left; color:#000000	DOMAIN=DM	179.244000 704.318000 257.994000 719.318000	0.58 0.79 1
6	font: Arial 12.0pt; text-align:left; color:#000000	DOMAIN=DS	177.465000 727.751000 258.465000 743.012000	0.77 0.93 1
6	font: ArialMT 12.0pt; text-align:left; color:#000000	DOMAIN=MI	175.201000 750.409000 256.801000 771.409000	0.87 0.87 1

Table 2. Updated Single Dataset with Modified Annotation Box Position and Color Code

IMPORT AN UPDATED NEW BLANK CRF AND UPDATE THE SINGLE DATASET WITH THE NEW BLANK CRF PAGE NAME AND PAGE NUMBER

Since the purpose is to export the existing annotations to an updated new blank CRF, the page number of annotations may need updating, as new CRF pages may have been added. In our sample new CRF, a new form “IRR Concomitant Medication Log [CMIRR]” is added, which occupies two pages. This can result in page number changes for the rest of existing forms, specifically for those forms after the new form. To adjust the page numbers, firstly, the table of contents page from the old CRF will be imported. Figure 5 shows an example of what the table of contents page looks like in a CRF.

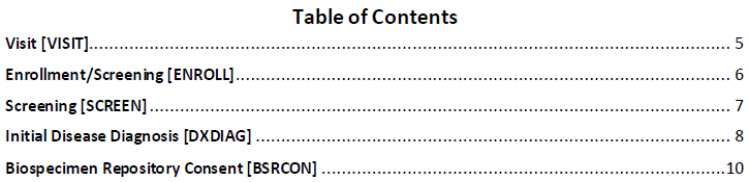

Figure 5. Example of Table of Contents Page in CRF

Before importing to SAS, the CRF PDF file needs to be converted to a TXT file so that SAS can use the code below to import the CRF TXT file (Elamathivadivambigai, 2017):

```
proc import datafile="&dev.annotated_crf.txt" out=table3 dbms=dlm replace;
    delimiter='[';
    getnames=no;
    guessingrows=2147483647;
run;
```

After some modifications in DATA steps, Table 3 is generated by displaying the CRF page name and the corresponding page number. Please note that the page numbers read into SAS by SAS XML Mapper are one less than the original page numbers in CRF. However, this will not affect the result as all annotations will finally be exported to the right corresponding CRF pages using the FDF code macro at the end.

crform	freetext_page
VISIT	4
ENROLL	5
SCREEN	6
DXDIAG	7
BSRCON	9

Table 3. Sample Dataset with CRF Page Name and Page Number

Then the dataset in Table 2 and the dataset in Table 3 can be merged together by variable freetext_page. Since some forms are more than one page long and only the starting page number will be displayed in table of contents (e.g. in Table 3, page 8 is not displayed because it's the second page of form DXDIAG), the RETAIN statement is used so that each annotation in Table 2 can get its CRF origin:

```
data table4(drop=_crform);
    merge table2(in=a) table3(rename=(crform=_crform));
    by freetext_page;
    if a;
    retain crform;
    if _crform ne '' then crform = _crform;
run;
```

The resulting dataset is shown in Table 4.

freetext_page	defaultstyle	text	box	color	crform
4	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.RDPENDTC	218.919000 668.363000 333.519000 686.545000	0.58 0.79 1	VISIT
4	font: Arial 12.0pt; text-align:left; color:#000000	Domain=DM	151.474000 702.753000 229.474000 721.537000	0.58 0.79 1	VISIT
5	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.SUBJID	1.345980 666.562000 75.890700 688.346000	0.58 0.79 1	ENROLL
5	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.SUBJID	154.146000 610.925000 290.946000 632.710000	0.58 0.79 1	ENROLL
5	font: Arial 12.0pt; text-align:left; color:#000000	Domain=DM	228.056000 704.062000 306.056000 722.847000	0.58 0.79 1	ENROLL
6	font: Arial 12.0pt; text-align:left; color:#000000	DM.RFICDTC	198.074000 573.872000 298.074000 591.872000	0.58 0.79 1	SCREEN
6	font: ArialMT 12.0pt; text-align:left; color:#000000	DM.RFPENDC	394.438000 575.181000 494.438000 593.181000	0.58 0.79 1	SCREEN
6	font: Arial 12.0pt; text-align:left; color:#000000	DOMAIN=DM	179.244000 704.318000 257.994000 719.318000	0.58 0.79 1	SCREEN

Table 4. Updated Single Dataset by Attaching CRF Origin for Each Annotation

Similarly for the new blank CRF, the table of contents will be imported to SAS and modified to generate the dataset with new CRF page name and number. The resulting dataset is displayed in Table 5 with the new page number variable name freetext_page2.

crform	freetext_page2
AE	41
AEYN	40
BSRCON	9
CARHHDT	93
CM	37
CMIRR	60

Table 5. Sample Dataset of New Blank CRF Page Name and Page Number

The dataset in Table 4 and the dataset in Table 5 can then be merged together by variable crform. In the combined dataset, again the RETAIN statement is used so that each annotation in Table 4 can get its updated CRF page number:

```
data final5(drop=freetext_page freetext_page2);
    set final4;
    by crform freetext_page;
    retain page;
    if first.crform and first.freetext_page then page = freetext_page2;
          else if first.freetext_page then page = page + 1;
run;
```

EXPORT ANNOTATIONS IN AN FDF FILE AND IMPORT THE ANNOTATIONS BACK TO THE NEW BLANK CRF

The next step is to put annotation text, page number, box position index, font attribute, and background color into macro variables (Black, 2015):

```
proc sql noprint;
    select count(*) into: num
    from final5;
    %let num = &num;
    select text, page, box, defaultstyle, color into
          : text1 -: text&num,
          : page1 -: page&num,
          : box1 -: box&num,
          : style1 -: style&num,
          : color1 -: color&num
    from final5;
quit;
```

Finally, these macro variables can be put into the FDF code template (Black, 2015) and the macro will loop through the number of annotations to be created:

```
%macro anno();
    data anno;
          length anno $1000;
          anno = '%FDF-1.2'; output;
          anno = "1 0 obj<</FDF<</Annots["; output;
          %do i = 1 %to &num;
                anno = "%eval(&i + 1) 0 R"; output;
          %end;
          anno = "]>>/Type/Catalog>>endobj"; output;
          %do i = 1 %to &num;
                anno = "%eval(&i + 1) 0 obj<<"; output;
                anno = "/C[%str(&&color&i)]"; output;
                anno = "/Contents(%str(&&text&i))"; output;
                anno = "/DS(%str(&&style&i))"; output;
                anno = "/F 4/Page %trim(%eval(&&page&i))"; output;
                anno = "/Rect[%trim(&&box&i)]"; output;
                anno = "/Subj(Text
                Box)/Subtype/FreeText/Type/Annot>>endobj"; output;
          %end;
          anno = "trailer"; output;
          anno = "<</Root 1 0 R>>"; output;
          anno = '%%EOF'; output;
    run;
%mend;
```

Within the macro, an opening wrapper code is created at the beginning. Then a place holder for each annotation is created so that the annotation can be created after. A close wrapper code is created at the end. By running this macro, the annotations will be converted from SAS datasets to FDF format and then exported and saved in an FDF file. Finally, the annotations in the FDF file can be imported back to the new blank CRF, which is similar to the steps of exporting annotations from aCRF PDF file.

CONCLUSION

Since it is very likely that new CRF pages are added to a study and the amended CRF needs to get annotated again, it would save a tremendous amount of time if the existing annotations could be automatically migrated from an old aCRF to the new one with the proposed method. Thus, the only thing left is to annotate the newly added CRF page rather than annotating the entire new CRF version by hand from scratch.

REFERENCES

Elamathivadivambigai, Aishhwaryapriya. 2017. “A novel method to track mapping of all CRF variables into SDTM datasets.” *Proceedings of the PharmaSUG 2017 Conference*, Baltimore, MD: PharmaSUG.

Black, Steven. 2015. “Have SAS Annotate your Blank CRF for you! Plus dynamically add color and style to your annotations.” *Proceedings of the PharmaSUG 2015 Conference*, Orlando, FL: PharmaSUG.

ACKNOWLEDGMENTS

I would like to take this opportunity to thank my manager Leo Wang, the Statistical Programming Associate Director and Qi Jiang, the Biometrics Vice President, for reviewing my paper and providing suggestions. I also would like to thank my colleague Boxun Zhang who inspired me to write this paper.

CONTACT INFORMATION

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

Yating Gu
Seattle Genetics, Inc.
(425) 527-2510
ygu@seagen.com