

Automating Production of the blankcrf.pdf

Walter Hufford, Novartis, East Hanover, NJ

ABSTRACT

The annotated blank Case Report Form (blankcrf.pdf) is a critical component of the NDA submission. Per FDA guidance, source data domain, variable name and controlled terminology for each case report form (CRF) item included in the tabulation datasets submitted should be displayed on the blankcrf.pdf. Production of the blankcrf.pdf is a tedious, non-programming task that is increasingly becoming the responsibility of the statistical programmer. This paper describes an easy to use, automated method of annotating the CRF.

INTRODUCTION

Manual annotation of the blankcrf.pdf is performed using the Adobe Acrobat comment tool. Each new annotation is created individually while repeat annotations such as page header information (e.g., STUDYID, USUBJID and VISIT) may be copied and pasted as a group from one page to another. However, the typical blankcrf.pdf consists of over 60 unique pages and each page may contain many annotations. That adds up to a lot of annotations making for a monotonous and time-consuming task. In addition, manual annotation significantly increases the odds of both incorrect and inconsistent (both within and across blankcrf.pdf files) annotations. Automating the annotation process significantly reduces both the time and effort required to produce the blankcrf.pdf while eliminating incorrect and/or inconsistent annotations.

ADOBE ACROBAT FDF FILE

When a blankcrf.pdf page is manually annotated, Adobe Acrobat stores the annotation attributes (e.g., font, font size, text, color, box size) internally. These annotation attributes may be extracted from Adobe Acrobat and stored in an external file called a Forms Data Format (FDF) file. This is done using the Adobe Acrobat comment export tool. The FDF file is text file format which can be viewed and edited using any text editor. Once created, an FDF file may be used to annotate other similar CRF pages using the Adobe Acrobat comment import tool.

The FDF file structure is made up of 3 types of objects: header, body and trailer. The header object appears at the top of the FDF file and describes the number of annotations and the annotation source file name. The header object is followed by one or more body objects used to describe each annotation's attributes (e.g., placement on the CRF page, size of the box surrounding an annotation, font and size of the annotation text, background color of the annotation box and PDF page to display the annotation on). The trailer object is at the bottom of the FDF file signaling the end-of-file. Figure 1 displays an example of an FDF file generated using the Adobe Acrobat comment export tool. The example is based on a single annotated PDF page containing a single annotation for the variable named SEX.

```
%FDF-1.2
%âäÏ
1 0 obj
<</FDF<</Annots[2 0
R]/F(/C/PharmaSUG2014/PharmaSUG2014_example_CRF_DM.pdf)/ID[<B8EE73C02F4D874C93AAE44ACF8C2212><C2D8B7C
87E8FCE47B3CE92FB8082505F>]/UF(/C/PharmaSUG2014/PharmaSUG2014_example_CRF_DM.pdf)>>/Type/Catalog>>
endobj
2 0 obj
<</BS 3 0 R/C[0.0 1.0 1]/Contents(SEX )/DA(0 0 0 rg /Arial 10 )/DS(font: Arial,sans-serif 10.0pt; text-align:left; color:#000000
)/M(D:20140406095629-04'00')/NM(7c71d692-03dc-45c4-abd9-4bf182d12a4f)/Page 0/RC(<?xml version="1.0"?><body
xmlns="http://www.w3.org/1999/xhtml" xmlns:xfa="http://www.xfa.org/schema/xfa-data/1.0/" xfa:APIVersion="Acrobat:10.1.5"
xfa:spec="2.0.2" style="font-size:10.0pt;text-align:left;color:#000000;font-weight:normal;font-style:normal
!;font-family:Arial,sans-serif;font-stretch:normal"><p>SEX </p></body>)/Rect[407.868 324.866 431.868
338.866]/Subj(TextBox)/Subtype/FreeText/T(NCDS)/Type/Annot>>
endobj
trailer
<</Root 1 0 R>>
%%EOF
```

Figure 1 - FDF File Example Generated with Adobe Acrobat Comment Export Tool

AUTOMATION AND THE ANNOTATION REPOSITORY

The above example describes creating an FDF file for a single blankcrf.pdf page. However, the FDF file is not limited to a single page. One of the annotation attributes is PDF page number. Using this attribute we are able to create FDF individual objects for each annotation on each page of a PDF. By doing so we are now able to annotate several pages of the blankcrf.pdf with a single FDF file using the PDF comment import tool. This is a huge step forward from the manual annotation process. However, each blankcrf.pdf must contain identical CRF pages and the page ordering must be aligned (e.g., AE blankcrf.pdf page must always be on page 1). This works well when annotating a few CRFs based on similar protocols using the same CRF pages. However, it does not succeed in providing a truly automated method of annotating CRFs across an organization.

Rather than create a single re-usable FDF file a better approach is to create a repository of annotations and associated attributes which is used in the creation of study-specific FDF files. A repository containing a few required annotation attributes for each standardized CRF in an organization is relatively easy to set-up and maintain. Simple SAS® programs are created to (1) load individual annotation attributes to the repository, (2) select study-specific appropriate annotations from the repository, and (3) produce study-specific FDF files. As existing CRF pages are updated or new standard CRF pages are implemented the repository is easily updated and the SAS programs only require updating if the FDF file structure changes.

CREATING AND UPDATING A PDF / FDF LIBRARY

Prior to creating the annotation repository, a library of single page annotated CRFs is created along with associated FDF files. These individual PDF and associated FDF files are named according to their unique CRF identification. Initially, the library is populated using an existing blankcrf.pdf containing many pages. The Adobe Acrobat split file tool may be used to create the individual PDF files from each blankcrf.pdf page.

New single page annotated PDF files and associated FDF files are added to the library as they are approved within an organization. This may be done manually by annotating the new CRF page or by entering the annotations in the repository with coordinates that will align annotations along the left side of the CRF page. The annotations may then be easily dragged into position and exported to an FDF file to act as a source for the repository. The repository coordinates are then replaced with the coordinates from the new FDF file.

Sometimes manual intervention is needed to adjust an annotation's coordinates so that it aligns with other similar annotations (e.g., MHCAT and MHSCAT). Although not required, it is more pleasing to a reviewer if annotation boxes align when stacked. To do this the single FDF is updated as desired (e.g., adjust coordinates). The single page PDF is then opened and the existing annotations are removed (e.g., open comment tool, select all comments, delete). The comment import tool is then used to import the revised FDF file containing the new coordinates and both (PDF and FDF) are saved to the PDF / FDF library. This aligns the annotated single page PDF with its associated updated FDF file.

If an issue is spotted in an existing standard or a standard is updated the single page PDF affected is simply corrected/updated and its associated FDF file is reproduced. This library of FDF files serves as the source for the annotation repository.

CREATING AND UPDATING THE ANNOTATION REPOSITORY

A simple SAS program is created to read the contents of the PDF/FDF library and create the annotation repository. Figure 2 displays a snippet of the SAS program which assigns a SAS fileref to a UNIX pipe file containing the result of a UNIX ls command. This is used to obtain the names of all the FDF files within the PDF/FDF library.

```
%let filrf=mypipe;  
%let rc=%sysfunc(filename(filrf, %str(ls /vob/&IRPROJU./&IRPROTU./util), pipe));
```

Figure 2 – SAS code to read contents of the PDF/FDF library

The SAS program cycles thru each of the FDF files listed in the UNIX pipe file searching for specific content required for the annotation repository. Figures 3 and 4 display snippets of the SAS program used to search the contents of the FDF file for the two required annotation attributes: annotation text and coordinates.

```

rx=prxparse("#Contents\((.*?)\)\/#io");
start1=1;
stop=reclen;

call prxnext(rx,start1, stop, _infile_, position, length);
if rx gt 0 and length GT 0 then do;
  call prxposn(rx, 1, start1, length);
  element=substr(_infile_,start1,length);
end;

```

Figure 3 – SAS code to search the FDF file for annotation text

```

rx=prxparse("#/Rect\[ (.*?)\]\#io");
start=1;

call prxnext(rx, start, stop, _infile_, position, length);
if rx gt 0 and length gt 0 then do;;
  call prxposn(rx, 1, start, length);
  _coord=substr(_infile_,start,length);
end;

```

Figure 4 – SAS code to search the FDF file for annotation coordinates

The same SAS program is used to update the annotation repository when an existing FDF file is updated or a new PDF/FDF file is added to the library.

The annotation repository contains 10 variables: CRFID (unique CRF identifier), DOMAIN (SDTM domain), ELEMENT (annotation text), C1-C4 (Adobe Acrobat coordinates for annotation placement on the CRF), SUPP (supplemental qualifier variable [yes/no]), CT (name of associated controlled terminology) and ORDR (order of domains when more than one domain per page). CRFID is created on the fly based on the name of the FDF file. All other variable values are taken directly from the individual FDF files. Figure 5 displays an example of an annotation repository.

	A	B	C	D	E	F	G	H	I	J
1	CRFID	DOMAIN	ELEMENT	C1	C2	C3	C4	SUPP	CT	ORDR
2	DMG02	DM	DM = Demographics	58	581	162	595			1
3	DMG02	DM	STUDYID	174	523	230	537			1
4	DMG02	DM	USUBJID	334	520	390	534			1
5	DMG02	DM	[NOT SUBMITTED]	575	519	683	533			1
6	DMG02	DM	[NOT SUBMITTED]	386	450	494	464			1
7	DMG02	DM	BRTHDTC	386	430	442	444			1
8	DMG02	DM	[NOT SUBMITTED]	344	400	452	414			1
9	DMG02	DM	AGE	381	384	410	398			1
10	DMG02	DM	AGEU	423	384	455	398		UNIT	1
11	DMG02	DM	[NOT SUBMITTED]	384	362	492	376			1
12	DMG02	DM	SEX	410	320	434	334		SEX	1
13	DMG02	DM	RACEOR in SUPPDM	143	278	262	292	Y		1
14	DMG02	DM	RACESY in SUPPDM	108	259	227	273	Y		1
15	DMG02	DM	ETHNICOR in SUPPDM	548	239	683	253	Y		1
16	DMG02	DM	SRCRFSY in SUPPSC	309	175	437	189	Y		1
17	DMG02	SC	SC = Subject Characteristics	165	581	336	595			2
18	DMG02	SC	SCORRES when SCTESTCD = SBJINIT	361	342	556	357			2
19	DMG02	SC	SCTEST	156	341	204	355			2
20	DMG02	SC	SCORRES when SCTESTCD = CHDBER	540	299	735	313			2
21	DMG02	SC	SCTEST	75	298	123	312			2
22	DMG02	SC	SCTEST	107	220	155	234			2
23	DMG02	SC	SCORRES when SCTESTCD = SRCRF	536	218	731	232			2

Figure 5 - Annotation Repository Example

The SUPP variable is used to flag annotations which refer to a supplemental qualifier variable. This flag is used in the DATA _NULL_ step to change the color of the annotation text from black to red. The ORDR variable is used to identify the order of domains on a blankcrf.pdf page when more than one domain contributes to the same page. This variable is also used in the DATA _NULL_ step to change the color of the annotation text box background color to highlight which annotations come from which domain.

The individual FDF files contain many more annotation attributes than are necessary to store in the repository. Annotation box coordinates are one example. The FDF file contains a 3 decimal place precision for each coordinate. The human eye cannot differentiate between coordinate 125.725 and 125 so the annotation repository only requires the whole number with no decimal precision. Figure 6 displays an example of an FDF file generated by a DATA _NULL_ step and contains only the attributes required for a blankcrf.pdf. Although this example generates the exact

same annotation as that displayed in Figure 1, note how much less information is stored in the DATA _NULL_ generated FDF file.

```
%FDF-1.2
1 0 obj
<< /FDF << /Annots [2 0 R ] >> >>
endobj
2 0 obj
<< /BS 21 0 R/C[0.0 1.0 1.0 ]/Contents(SEX )/DS(font: Arial 10.0pt; text-align:left; color:#000000 )/Page 0 /Rect[407 324 431 338
]/Subj(TextBox)/Subtype/FreeText/T(NCDS)/Type/Annot>>
endobj
trailer
<< /Root 1 0 R >>
%%EOF
```

Figure 6 - FDF File Example Generated with a DATA _NULL_ Step

CREATING AND USING A STUDY-SPECIFIC DRIVER

A study-specific driver is created for each new blankcrf.pdf to be produced. The driver identifies which unique CRF pages are used in a study and the blankcrf.pdf page on which they appear. Note that Adobe Acrobat starts page numbering at page 0 so the FDF file page attribute is 0 for annotations appearing on the first page of the blankcrf.pdf. To avoid confusion when entering page numbers in the driver, it is a good idea to allow the end-user to enter page 1 in the study-specific driver and to convert to 0 in the SAS code used to produce the FDF file.

The study-specific driver contains 3 variables: CRFID (unique CRF identifier), DOMAIN (SDTM domain), and PAGE (blankcrf.pdf page of CRFID). It is created from a template driver produced at the same time as the annotation repository. The template driver contains a row for each unique CRFID in the repository and values for CRFID and DOMAIN are pre-populated. Thus, the end-user only needs to identify which page the CRFID occurs on in their blankcrf.pdf. CRFIDs not used are simply left with a blank PAGE value or removed from the study-specific driver. Figure 7 displays an example of a study-specific driver.

	A	B	C
1	crfid	domain	page
2	AEI16	AE	1
3	AEI16R	AE	2
4	AEI01	AE	35
5	AEI02	AE	79
6	CM2G02	CM	37
7	CM2G02R	CM	36
8	CMG02Rs	CM	12
9	CMG02s	CM	11
10	ZQG07s4	EX	15
11	DMG02	DM	17
12	DMG02	SC	17

Figure 7 – Study-specific Driver Example

CREATING A STUDY-SPECIFIC FDF FILE

A simple SAS program is created to read the study-specific driver, select the required data from the annotation repository and create a study-specific FDF file which will be imported into the blankcrf.pdf. The program contains basic SAS code which includes a simple DATA _NULL_ step to produce the FDF file.

Recall that the header object appearing at the top of the FDF file describes the number of annotations to follow in the FDF body. Care must be used here to ensure the header and body align on the number of objects. If the FDF header contains more objects than the body, Adobe Acrobat will not allow the FDF file to be imported. If the FDF header contains fewer objects than the body, Adobe Acrobat will successfully import the FDF file with no warning but will only display body annotations up to the number specified in the header resulting in missing annotations at end of blankcrf.pdf.

CONCLUSION

Production of the blankcrf.pdf is a tedious, non-programming task that is increasingly becoming the responsibility of the statistical programmer. However, the standardization of CRFs in conjunction with the simplicity of the FDF file structure brings automation of this task within reach for all organizations. As demonstrated in this paper, SAS can easily create the FDF file as long as we can tell it which attributes to assign to each annotation (e.g., annotation text,

location on CRF page). Doing so allows for quick, consistent and accurate blankcrf.pdf production and allows programmers to focus on programming.

ACKNOWLEDGMENTS

I would like to thank Christina Scienski and Geo Joy for their review and suggestions.

CONTACT INFORMATION

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

Walter Hufford
Novartis
One Health Plaza
East Hanover, NJ 07936-1080
Walter.Hufford@Novartis.com
www.Novartis.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.