PharmaSUG 2014 - Paper CC40

Inserting MS Word Document into RTF Output and Creating Customized Table of Contents Using SAS® and VBA Macro

Haining Li, Neurological Clinical Research Institute, Mass General Hospital, Boston, MA
Merit E. Cudkowicz, Neurological Clinical Research Institute, Mass General Hospital, Boston, MA
Hong Yu, Neurological Clinical Research Institute, Mass General Hospital, Boston, MA

ABSTRACT

In clinical trials, to effectively monitor study progress and subject safety, the investigators are frequently requested to submit a set of tables and listings at regular basis throughout the study. With the ODS (Output Delivery System) RTF destination, the programmers could build tables and listings that are opened directly in MS Word and other word-processing packages. However, to expedite the review process, with each submission, a Summary Report to highlight any study update during the report cycle is highly recommended. In practice, the key study team members often provide the programmers with a MS Word document with multiple sections of study updates to be inserted into the RTF outputs. Having all information consolidated in one document will keep the integrity of the report and make it easy for reference. Also, with the various length of the Summary Report at each submission, it would be helpful if the SAS program could automatically create a Table of Contents (TOC) which includes the hyperlinks to each section of the Summary Report in the final RTF outputs. Unfortunately, SAS does not provide a stable function to insert the MS Word document into the RTF outputs and create the customized TOC. This paper will offer a method to insert any MS Word document and prepare the inserted file to create customized TOC using SAS and VBA Macro. With this approach, the customized TOC will maintain the hyperlinks to configurable sections of the inserted documents. This method is very flexible, robust and can find broad application in SAS reporting.

INTRODUCTION

In clinical trials, the SAS programmer is responsible for creating the SAS code to generate a set of tables and listings for reviewers to monitor study progress and subject safety. As part of the submission, it is often requested by reviewer to insert the study updates provided in separate Word documents into final SAS output for each report. Because of complexity and proprietary format of MS Word, it is always a challenge to insert the MS Word documents and create the customized TOC for the inserted documents for easy navigation. To address this issue, this paper introduces an innovative method to insert MS word document into RTF output and create customized table of contents.

PROBLEM FORMULATION, PROPOSED METHOD AND DETAILED DESCRIPTION

To insert a MS Word into SAS, we divide the process into three steps. Step 1 records VBA macro to split MS Word into separate pages, step 2 keeps track of the split pages, and step 3 inserts the split pages into the proper place and creates the TOC. These three steps are presented below in details.

Step 1: Record the VBA macro below to MS Word in order to split the documents which need to be inserted into separate pages.

Place all the Word documents which are needed to be inserted in a folder, e.g. called `insert_files.` Inside this folder, create a subfolder, e.g. called SPLITPAGE and assign a SAS libname to it. It is always a good practice to delete all the possible files from folder SPLITPAGE before each SAS submission to prevent any possible problem and/or conflict. The SAS code to accomplish the above is listed below:

```
X "MKDIR ""&SAS_Dir.\insert_files\SPLITPAGE"" ";
libname split "&SAS_Dir.\insert_files\SPLITPAGE";
X "del /Q   &SAS_Dir.\insert_files\SPLITPAGE";
```

There is a simple way to launch MS Word from SAS with the following SAS code:

```
x ' "C:\Program Files\Microsoft Office\OFFICE11\WINWORD.exe" ';
%LET RC=%SYSFUNC(SYSTEM(START WINWORD));
```

System options NOXWAIT and NOXSYNC are required before launching MS Word within SAS. The NOXWAIT option specifies that the DOS command prompt window should disappear without requiring user to type EXIT when the process is finished, while the NOXSYNC specifies that the process should execute asynchronously. That means

control is returned immediately to the SAS System and the command continues executing without interfering with the current SAS session (Lori S. Parsons, PharmaSUG 2009, cc18).

The SAS macro splitpage can split the files into separate pages and the newly generated pages are saved in the folder: &SAS_Dir.\insert_files\SPLITPAGE as created before. Each MS Word page will be created as a single Word file named according to the file number and page number, separated by symbol '_'. For instance, if a file named Section2 has seven pages, then seven files named Section2_1, Section2_2 …Section2_7 will be generated and each of them contains only one page. If a file named Section4 has only one page, then just one file with name Section4_1 will be generated. Here is the SAS code for macro splitpage.

```
%macro splitpage (document_list= );
  %local i  fl;

     %do i = 1 %to
     %eval(%sysfunc(count(%sysfunc(compbl( %sysfunc(trim(%sysfunc(left(&document_lis
     t))))))),%str( ))) + 1);
      %let var
     =%sysfunc(compbl( %sysfunc(trim(%sysfunc(left(%qscan(&document_list,&i,%str( ))
     ))))));
     %let file_name = %scan(&document_list,&i,%str( ));
     %put &file_name;
     %let infile =  &SAS_Dir.\05_ancillary_data\insert_files\&file_name..doc ;
     %put &infile;

     %LET RC=%SYSFUNC(SYSTEM(START WINWORD));

     DATA _NULL_;
       X=SLEEP(8);
     RUN;

     FILENAME WORD DDE 'WINWORD|SYSTEM';

     DATA _NULL_;
       FILE WORD;
       PUT '[FILEOPEN .NAME = "' "&infile" '"]';
       PUT '[ExtractAll( )]';
       PUT '[FILECLOSEALL]';
       PUT '[FILEEXIT]';
     RUN;
     DATA _NULL_;
       X=SLEEP(5);
     RUN;
  %end;
%mend splitpage;
%splitpage (document_list= Section1  Section2   Section3  Section4  );
```

In the above SAS DATA step, a VBA function called "ExtractAll()" extracts all the pages from a MS Word file and saves them into the folder as individual files. In this macro, a system level setting switch "Application.ScreenUpdating" should be turned off to avoid unnecessary screen update and speed up the processing. Otherwise, more popup message will slow down the document processing. Meanwhile, one important portion of this module is to find the correct page break location. This VBA function uses 1-of-N approach to process the MS Word document, which means it continues to eliminate unwanted pages until the only needed page is left. The last page and previous pages are handled differently to make the split process go smoothly inside the VBA function. Here is the VBA code for the above mentioned function:

```
Sub ExtractAll()
    Application.ScreenUpdating = False
    OriginDocName = ActiveDocument.FullName
    Dim i As Integer
    Dim iNumPages As Integer
    Dim sCurrentDoc As String
    Dim rgePages As Range
    Dim PgFlds As String, j As Long
```

```
    Dim FldHi As Range, FldLo As Range, Fld As Range
    Dim PgHi As Long, PgLo As Long
    MyName = Replace(ActiveDocument.Name, ".doc", "_")
       'Get the current document's name
    iNumPages =
Documents(ActiveDocument.FullName).ComputeStatistics(wdStatisticPages)
Documents(ActiveDocument.FullName).Close SaveChanges:=wdDoNotSaveChanges
    'Get the number of pages
    For j = 1 To iNumPages
        Set doc = Documents.Open(OriginDocName)
        With doc
          .SaveAs .Path & "/SPLITPAGE/" & MyName & j, .SaveFormat
           Set FldHi = .Range.Characters.Last
           Set FldLo = .Range.Characters.Last
           If j <> iNumPages Then
               Set FldHi = FldHi.GoTo(What:=wdGoToPage, Name:=iNumPages)
               Set FldHi = FldHi.GoTo(What:=wdGoToBookmark, Name:="\page")
               Set FldLo = FldLo.GoTo(What:=wdGoToPage, Name:=j)
               Set FldLo = FldLo.GoTo(What:=wdGoToBookmark, Name:="\page")
               Set Fld = .Range(FldLo.End, FldHi.End)
               Fld.Select
               Fld.Cut
               If Fld.Characters.First.Previous.Text = Chr(12) Then
                   Fld.Text = vbNullString
               Else
                   Fld.Text = Chr(12)
               End If
           End If
           If j <> 1 Then
            Set FldHi = FldHi.GoTo(What:=wdGoToPage, Name:=j - 1)
            Set FldHi = FldHi.GoTo(What:=wdGoToBookmark, Name:="\page")
            Set Fld = .Range(0, FldHi.End)
            Fld.Text = vbNullString
           End If
           While .Characters.Last.Previous.Text Like "[" & vbCr & Chr(12) & "]"
             .Characters.Last.Previous.Text = vbNullString
           Wend
           .Save
           .Close
         End With
            'Activate the document where we started from
         Next
              'Do work
        Application.ScreenUpdating = True
        Documents.Open (OriginDocName)
End Sub
```

Step 2: Use the following SAS code to keep track of each file and each split page.

After splitting pages, the following codes could be used to find all the RTF and Word files in the specified folder, and put the file names into the lists.

```
%let path=&SAS_Dir.\05_ancillary_data\insert_files\SPLITPAGE;
FILENAME DIRLIST PIPE "DIR ""&PATH"" /B ";

DATA dirlist1;
INFILE dirlist LENGTH=RECLEN;
INPUT FILENAME $varying1024. RECLEN;
IF SCAN(filename,2,'.') IN ('doc' 'rtf');
path="&path";
filename1=STRIP(filename);
filename=STRIP(path)||'\'||STRIP(filename);
RUN; (Ajay Gupta, PharmaSUG 2013, cc24)
```

After that, each file name list should be analyzed to obtain the section number and page number, which is needed by step 3. the section number and page name could be separated by '_'. So, if a different separator is used, changes should be made accordingly in the code. Here is the SAS code for analyzing the file names.

```
data dirlist2;
set dirlist1;
filename1t=tranwrd(filename1,'.doc'  ,'');
filename2=tranwrd(filename1t,'.rtf'  ,'');
file_orig=scan(filename2,1,'_');
file_page=scan(filename2,2,'_');
  filename_rtf1 =tranwrd(FILENAME,'\','\\\\');
  filename_rtf2 ='\*\fldinst {\\INCLUDETEXT'||' "'||filename_rtf1||'"}';
  filename_rtf3='{\field{'||filename_rtf2||'} }';
  filename_rtf ="'"||filename_rtf3||"'";
  drop filename_rtf1 filename_rtf2 filename_rtf3;
run;

proc sort data= dirlist2; by file_orig file_page ;run;

data _null_;
  set dirlist2;
  by file_orig file_page ;
  retain j k (0,0);
  j=j+1;
  if first.file_orig then do;
    j=1;
    k=k+1;
    call symput('filenum',k);
    *macro var. &filenum: Number of files;
    call symput(compress('file'||k),file_orig);
    *macro var. &&file&k: file name of the k-th file ;
  end;
  if last.file_orig then do;
    call symput(compress('pagenum'||k),j);
    *macro var. &&pagenum&k: Number of pages in the file_k;
  end;

  call symput(compress('page'||k||'_'||j),filename_rtf );
  *macro var. &&page&k._&j: Name of the j-th page in the k-th file;
 run;
```

Step 3: Insert the split pages and generate table of contents

By assigning the parameters for section number and page number, programmer can control the arrangement of the inserted sections and create the customized table of contents. The sample output from the code below is given in Figure 1. In this example, all four sections from inserted Word documents are shown in the table of contents.

The SAS code to insert the split pages and generate table of contents are given below.

```
filename rpt   "temp.rtf";
ods rtf file=rpt   contents notoc_data startpage=yes ;
  ods escapechar='^';

%macro rpt_insert;
%macro Sections ;
   ODS RTF  startpage=now  TEXT=  &&&page&i._&j   ;
%mend Sections;

%do i=1 %to &filenum;
    %do j =1 %to &&pagenum&i;
      %if &i=1 and &j=1 %then %do;
        ods rtf prepage ="^S={outputwidth=100% just=l} {\tc\f3\fs0\cf8   Section 1
Study Summary}";
      %end;
```

```
     %if &i=2 and &j=1 %then %do;
        ods rtf prepage ="^S={outputwidth=100% just=l} {\tc\f3\fs0\cf8   Section 2
Study Update}"; * Reference: Jay Zou, Proceedings of the Sas Global Forum 2008.
March 16-19, 2008;
     %end;
     %if &i=3 and &j=1 %then %do;
        ods rtf prepage ="^S={outputwidth=100% just=l} {\tc\f3\fs0\cf8   Section 3
Study Inclusion Criteria }";
     %end;
     %if &i=4 and &j=1 %then %do;
        ods rtf prepage ="^S={outputwidth=100% just=l} {\tc\f3\fs0\cf8   Section 4
Study Exclusion Criteria }";
     %end;
     ODS RTF ;
     %unquote(%nrstr( %Sections ; )  ); quit;
    %end;
  %end;
%mend ;
%rpt_insert;
```

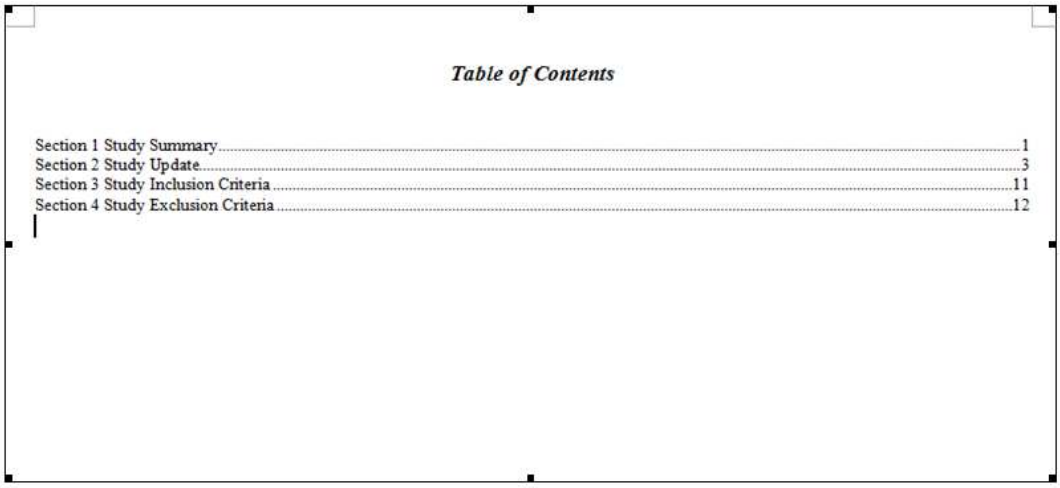

Figure 1. Sample TOC output from the proposed SAS codes

CONCLUSION

In this paper, we proposed a generic solution for inserting multiple MS Word documents into the SAS RTF report. The proposed method can be used to split a word document into individual pages cleanly using SAS and VBA Macro. By organizing the split pages, the documents can be inserted in a user controllable order and therefore a customized table of contents can be established accordingly.

REFERENCES

Gupta, Ajay. “Combining First Page of Multiple RTF Outputs in SAS using Bookmark and VBA Macro.” *Proceedings of the PharmaSUG 2013 Conference*. May 12-15, 2013, Chicago. Available at http://www.lexjansen.com/cgi-bin/xsl_transform.php?x=pharmasug2013&s=pharmasug&c=pharmasug.

Zhou, Jay. “ Creating a Custom Table of Contents in RTF Documents.” *Proceedings of the Sas Global Forum 2008*. March 16-19, 2008, San Antonio, Texas. Available at http://www2.sas.com/proceedings/forum2008/238-2008.pdf.

Parsons, Lori S. “ Importing Data from Microsoft Word into SAS.” *Proceedings of the PharmaSUG 2009 Conference*. May 31-June 3, 2009, Portland, Oregon. Available at http://www.lexjansen.com/pharmasug/2009/cc/cc18.pdf.

RECOMMENDED READING

- Advanced Programming for SAS®

CONTACT INFORMATION

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

Haining Li
Mass General Hospital
50 Staniford St
Boston, MA
617-643-5123
Hli15@partners.org