Paper 077-2012

Using DDE and VBA Techniques to Import Data from Microsoft Word Tables in Programming Specification Files into SAS®

Zemin Zeng, Forest Research Institute, Inc., Jersey City, NJ
Mei Li, Novartis Pharmaceuticals Corporation, East Hanover, NJ

ABSTRACT

This paper will introduce a SAS application with Microsoft Word for selecting table data in Word files and importing data from Word tables into SAS. Dynamic Data Exchange (DDE) and Visual Basic for Application (VBA) techniques used in the application will be presented in great detail. The application significantly improves the efficiency and sustainability of automatically converting Word tables to SAS data. The application is widely used at work for programmatically retrieving the comments from the programming specification files during the creation of Data Definition documentation (define.xml/define.pdf) for New Drug Application (NDA) electronic submissions.

INTRODUCTION

In the pharmaceutical industry, Microsoft Word is standard word processing software and widely used to write formal documentations, including Clinical Study Protocol, Statistical Analysis Plan with table/listing/graph mock-ups and derived analysis dataset Programming Specification. These Word documents are guidelines for SAS programming and statistical analysis in clinical trials. Since SAS is the preferred statistical software in the pharmaceutical industry as recommended by FDA to perform statistical analysis on clinical trial data, it is highly desirable to find convenient and efficient ways to import information from Word files into SAS, especially import tables in Word files into SAS data. Regarding developments in this direction, there have been essentially two approaches to import Word tables to SAS [See Zhou (2009) for a summary]:

- One approach is to save Word tables as RTF files, and read the latter as plain text files into SAS, by using RTF tagsets to restore the tabulate data structure in SAS [See Hagendoorn, Squire, and Tai (2006)];
- The other approach is to preserve the tabulate structure by copying Word tables to Excel, and then use PROC IMPORT to read in data from Microsoft Excel [See Zhou (2009); Chen, Cui and Moseley (2011)].

In this paper, we shall introduce an alternative approach which was developed when we worked on Data Definition documentations (define.xml/define.pdf) for New Drug Application (NDA) electronic submissions. For the creation of define.xml/define.pdf, we need to copy the comment column from dataset programming specification tables in Word files. A key step to automate the comment filing process is to read Word tables in programming specification files into SAS datasets. In this paper, we shall provide all steps with details in our application to automatically import Word table data from Word files into SAS datasets by using Data Exchange (DDE) and Visual Basic for Application (VBA) techniques. The application is a result of dynamic interactions of SAS software with Microsoft Word. To select tables in dataset programming files and convert the Word tables into SAS datasets, users only need to click buttons, Specifically, click buttons to select locations of dataset specification files and output datasets, and click button to execute SAS program (See Figure 1 for the interface of our application on the next page).

The organization of the paper shall be the following: 1. Introduce the background of our application including the structure of the dataset programming files. 2. Briefly introduce the DDE techniques to read Word tables to SAS datasets, and how to overcome the key problems. 3. Briefly introduce the interaction between SAS and VBA, with more focus on the integration of SAS programs and VBA macros and how to run a SAS program by simply clicking the VBA button. 4. Finally, introduce the design of our application and the overall process flow.

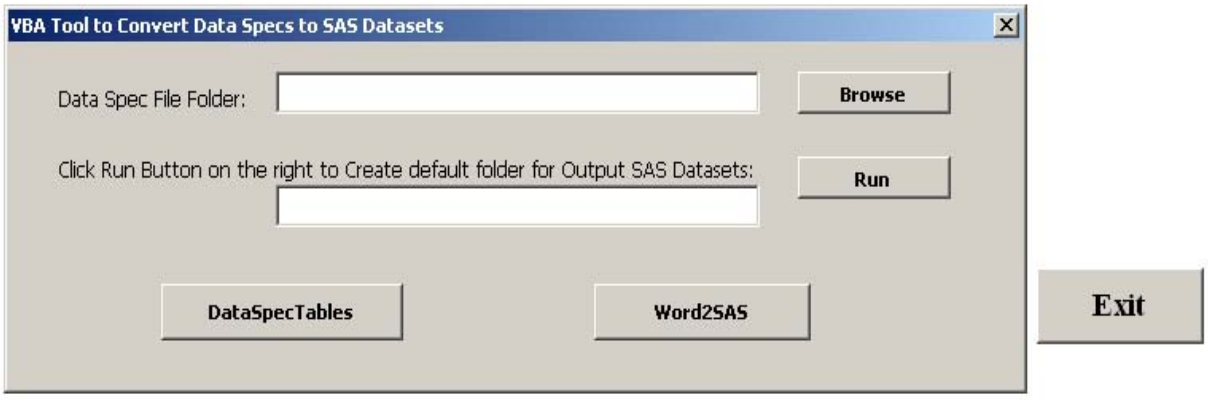
For the convenience of the interested readers, we include the complete SAS program code of our application in Appendix 1, and the key ingredients of the VBA macro code in Appendix 2, so that readers can easily modify them to fit individual needs. We hope our paper shows the power and beauty of the dynamic interactions between SAS software and Microsoft VBA, and encourages interested users (experienced SAS programmers or VBA experts) to develop user friendly new applications in Microsoft Word, through dynamic conversations between SAS programs and VBA macros.

BACKGROUND OF THE APPLICATION

We developed this application when we created Data Definition documentations (define.xml/define.pdf) for our clinical study NDA electronic submissions, In these clinical studies, we first created derived analysis datasets based on

programming specification files, then worked on define.xml/define.pdf. The comment column in define.xml/define.pdf file was previously needed to be manually copied from the tables in the programming specification files. Figure 2 is a sample of the programming specification file.

Figure 1: The Interface of Our Application for Converting Tables to SAS Datasets



There are two parts in the programming specification files. The first part is text which provides instructive information about the derived analysis datasets (such as purpose of the specifications, source raw SAS datasets structure of the target derived analysis dataset, and etc). The second part is a Word table which provides the variable attributes for the derived analysis dataset including comment column for the algorithm to derive.

Figure 2: Sample Derived Dataset Specification File

COMPANY NAME, INC. PROTOCOL XXX-MD-XX
STATISTICAL PROGRAMMING

1. **Purpose**
The derived data set DCOPD is created to store COPD efficacy parameters in such a structure to simplify the process of generating statistical analysis tables, data listings and figures.

2. **Source SAS Data Set**
The sources SAS data sets for DCOPD are the derived data set D_PROF and the raw SAS data sets COPDEXAC and HOSP.

3. **Derived Data Set Structure**
The structure of the derived data set DCOPD will be horizontal. DCOPD contains multiple records for each patient. All COPD records will be kept in this data set.

4. **Relevant SAS Variable Attributes for DCOPD (label: (D) XXXMDxx COPD Dataset)**
The variables in the table below are the additional variables that will be present in the data set:

(D): derived variable

NAME	LABEL	TYPE	FORMAT	LENGTH	COMMENTS
SOURCE	(D) Source Data	NUM	CPSOURCE. 1=COPD Exacerbation 2=Hospitalization		= 1 if from COPDEXAC raw data. = 2 from HOSP raw data.
COPSTDT	(D) COPD Start Date	NUM	YYMMDD10.	8	Derived from COPSTDTD, COPSTDTM and COPSTDTY
COPSTDTC	(D) COPD Start Date (C)	CHAR	ISO 8601	10	Character Decode of COPSTDT
COPENDT	(D) COPD Stop Date	NUM	YYMMDD10.	8	Derived from COPENDTD, COPENDTM and COPENDTY

The goal of our application is to convert the Word tables in programming specification files to SAS datasets (Once we have these datasets, the comment column in define.xml/define.pdf could be programmatically filled from the specification files. Also the variables attributes in programming specification files could be used to check or generate the corresponding ones in derived analysis datasets), and we shall introduce all the details on how to achieve this goal by simply clock buttons in next sections.

DDE TECHNIQUES FOR READING WORD TABLES TO SAS

It is well known that we could directly read Word table data into SAS via DDE, below SAS code could be one of the sample based on the information stated on SAS support website (which is a good reference place for systematic introductions on DDE for both Microsoft Excel and Microsoft Word as DDE servers with live examples in all scenarios): *http://support.sas.com/documentation/cdl/en/hostwin/63285/HTML/default/viewer.htm#ddeexamples.htm*

```
filename testit dde 'winword|" C:\stat\vba\DDT\DCOPD.doc "!CC' notab;
data testit;
  length var1 var2 var3 ... $1000;
  infile cmds dlm='09'x notab missover dsd LRECL=8000;
  input var1 $ var2 $ var3 $ ... ;
run;.
```

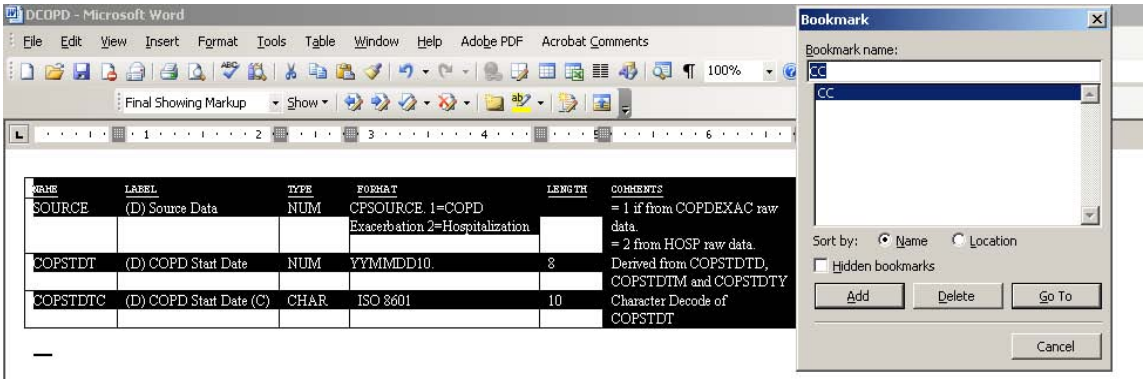
However, there are three critical things needed to take care of in order to run the above SAS code correctly to read Word table into SAS:

1. Add bookmark

We need to add bookmark when use DDE to read Word table into SAS. It is worth to point out that we should first highlight the World table then click INSERT->Bookmark (as illustrated in Figure 3) since the bookmark refers to the Word table. If forgot to highlight first, then one will get the error message below when run the above DDE SAS infile code.

```
ERROR: DDE session not ready.
FATAL: Unrecoverable I/O error detected in the execution of the DATA step program.
       Aborted during the EXECUTION phase.
```

Figure 3: The Correct Way to Add Bookmark



2. Word file need to be open when

Remember to keep the Word file open when run the above DDE SAS infile code. If not, the below ERROR message will occurs:

```
ERROR: Physical file does not exist, winword|"C:\stat\vba\DDT\DCOPD.doc"!CC.
```

3. Replace Special Characters Carriage return and Soft return (Shift+Enter)

The carriage return (Enter) and soft return (Shift+Enter) in Word table cell will cause unexpected effects of reading data. For example in Word table, when there is a carriage return in the table as in Figure 4, the generated SAS dataset will be as in Figure 5. The carriage return forces SAS to stop reading in second part of the cell.

Figure 4: Word Table with Return Special Character

Carriage return

NAME	LABEL	TYPE	FORMAT	LENGTH	COMMENTS
SOURCE	(D) Source Data	NUM	CPSOURCE. 1=COPD Exacerbation 2=Hospitalization		= 1 if from COPDEXAC raw data. = 2 from HOSP raw data.
COPSTDT	(D) COPD Start Date	NUM	YYMMDD10.	8	Derived from COPSTDTD, COPSTDTM and COPSTDTY
COPSTDTC	(D) COPD Start Date (C)	CHAR	ISO 8601	10	Character Decode of COPSTDT

In our application, we use VBA techniques to replace special characters with a single space for simplicity (One may replace these special characters with something standard characters, and then replace back in SAS data). In VBA language, special characters are represented by Chr Function and constants. For example Soft return (Shift+Enter) = Chr(11) and Carriage return = Chr(13).

Figure 5: Sample SAS Dataset from Word Table with Carriage Return

	NAME	LABEL	TYPE	FORMAT	LENGTH	COMMENTS
1	SOURCE	(D) Source Data	NUM	CPSOURCE. 1=COPD Exacerbation 2=Hospitalization		= 1 if from COPDEXAC raw data.
2	= 2 from HOSP raw data.					
3	COPSTDT	(D) COPD Start Date	NUM	YYMMDD10.	8	Derived from COPSTDTD, COPSTDTM and COPSTDTY
4	COPSTDTC	(D) COPD Start Date (C)	CHAR	ISO 8601	10	Character Decode of COPSTDT

Once we take care of these special characters, we will get the desired SAS dataset as in Figure 6 when we use DDE techniques to read Word table into SAS correctly.

Figure 6: Sample SAS Dataset from Word Table with Carriage Return Removed

	NAME	LABEL	TYPE	FORMAT	LENGTH	COMMENTS
1	SOURCE	(D) Source Data	NUM	CPSOURCE. 1=COPD Exacerbation 2=Hospitalization		= 1 if from COPDEXAC raw data. = 2 from HOSP raw data.
2	COPSTDT	(D) COPD Start Date	NUM	YYMMDD10.	8	Derived from COPSTDTD, COPSTDTM and COPSTDTY
3	COPSTDTC	(D) COPD Start Date (C)	CHAR	ISO 8601	10	Character Decode of COPSTDT

DYNAMIC INTERACTION BETWEEN SAS PROGRAM AND VBA MACRO

Visual Basic for Applications (VBA) is a common development environment among all Office applications, we have developed papers on VBA with Microsoft Office applications to interact with SAS software to perform post-process of SAS output in Microsoft Word [See Zeng & Li 2010 and 2011]. In this paper, we shall introduce new conversations between SAS programs and VBA macros. This interaction between SAS and VBA is extremely powerful, especially as in our application when SAS program requires actions in Microsoft Word as in using DDE techniques to read Word tables into SAS datasets.

The basic idea in our application of interaction between SAS program and VBA macro is that an executive file can be opened when one clicks a VBA button. In this executive file, we will call SAS system to run a SAS program. Below are key ingredients to execute a SAS program from VBA by just click a button:

Step 1. To use ShellExecute function to open a bat file, below is the sample code modified from the article by Michael Suodenjoki (*http://www.suodenjoki.dk/us/productions/articles/vbashellexecute.htm*):

```
ShellExecute(0, "Open", "\\frxwf002\vol2\Csdm-sysshare\SHAREALL\Utility\DataSpec2SAS\BATCH.bat",
      "",  0, 1)
```

Step 2. In the bat file, one need to call SAS system to run the SAS program, below is the sample code of the bat file (users may change the SAS program name, output list and log files as necessary). For more ways to execute SAS batch jobs, please refer to the note at *http://support.sas.com/techsup/technote/ts648/ts648.pdf*

```
sas C:\stat\vba\DDT\ddt_las.sas -NOSPLASH -NOICON -PRINT C:\stat\vba\DDT\ddt_las.lst
      -LOG  C:\stat\vba\DDT\ddt_las.log
```

THE DESIGN OF OUR APPLICATION AND OVERALL PROCESS FLOW

In the previous two sections, we introduced both SAS and VBA key techniques in our application for converting Word tables in programming specification files into SAS datasets by simply clicking buttons. This converting process is a perfect case for demonstrating the beauty of the interaction between SAS software and Microsoft Word via VBA macros. In this section, we shall introduce the interface of the application and what these buttons do for us.

The interface of our application is very simple as in Figure 7. There are essentially 5 buttons to click, specifically:

1. Click '**Browse'** to input the directory of programming specification files;

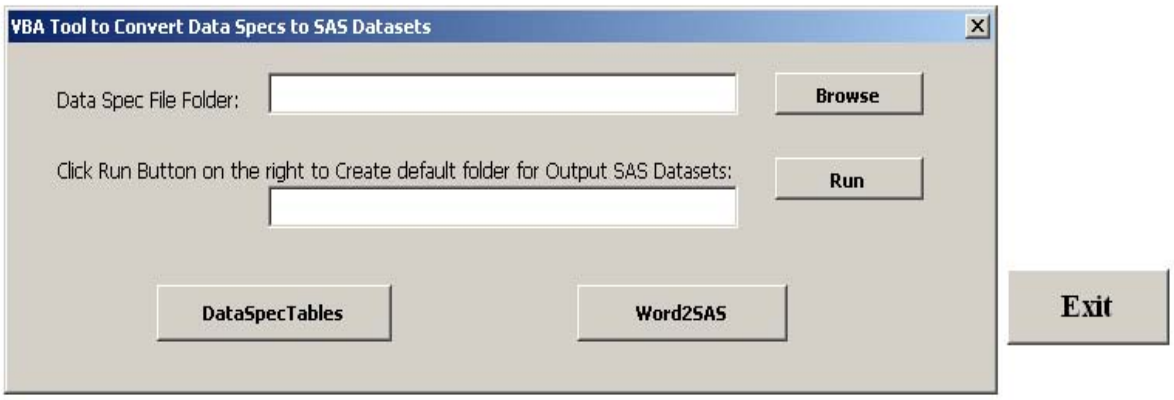
2. Click '**Run**' to select the output directory (at Forest, we have some default location for this output);

3. Click '**DataSpecTables**' to select the Word table portion in the programming specification files and save them as intermediate files to the output directory, furthermore, the click shall trigger the execution of the build-in VBA macro to add bookmarks, replace special characters carriage return and soft return (Shift+Enter) in the intermediate files;

4. Click '**Word2SAS**' to open the Batch file to trigger the execution of the SAS program. In this step, SAS datasets will be created and saved at the output directory, so does the SAS log file;

5. Click '**Exit**' to close and then delete all intermediate Word files and files generated by SAS.

Figure 7: The Interface of Our Application



CONCLUSION

The paper presents an alternative way to programmatically import Microsoft Word tables into SAS datasets by applying Visual Basic for Application (VBA) and Dynamic Data Exchange (DDE) techniques. In addition, the application with simply clicking buttons is very convenient and reliable, and demonstrates how VBA along with SAS enables us to achieve tremendous efficiencies in our day-to-day use of Microsoft Word.

REFERENCES

Michael Hagendoorn, Jonathan Squire, and Johnny Tai (2006). *Save Those Eyes: A Quality-Control Utility for Checking RTF Output Immediately and Accurately*. Proceedings of the Thirty-first SUGI Conference, paper 066-31

Jay Zhou (2009). *Importing Data from Microsoft Word into SAS®*. Paper CC18, the Proceedings of the PharmaSUG 2009

Zeng, Zemin and Li, Mei (2010). *Using SAS® Software and Visual Basic for Application to Dynamically Manipulate SAS List Files*, Coders' Corner Section, Proceedings of the SAS Global Forum 2010.

Zeng, Zemin and Li, Mei (2011). *Using Visual Basic for Application to Produce Table Of Contents from SAS Output List Files*, Applications Development Section, Proceedings of the PharmaSUG 2011

Min Chen, Xiangchen (Bob) Cui and Scott Moseley (2011). *Automating the Process of Preparing Data Definition Document for NDA Electronic Submission from Programming Specification in Word Format*, Paper CD02, Proceedings of the PharmaSUG 2011

Michael Suodenjoki (2002). *Starting Programs from VBAUsing ShellExecute to start any program or shortcut from inside*. http://www.suodenjoki.dk/us/productions/articles/vbashellexecute.htm

SAS Institute, technique note. *Examples of Batch Processing under Windows*.
http://support.sas.com/techsup/technote/ts648/ts648.pdf

SAS Institute, documentations. *DDE Examples*.
http://support.sas.com/documentation/cdl/en/hostwin/63285/HTML/default/viewer.htm#ddeexamples.htm

CONTACT INFORMATION

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

Zemin Zeng, Forest Research Institute, Inc., Jersey City, NJ
Email: *zengzemin@gmail.com / zemin.zeng@frx.com*

Mei Li, Novartis Pharmaceuticals Corporation, East Hanover, NJ
Email: *mei-4.li@novartis.com*

ACKNOWLEDGMENTS

APPENDIX 1: THE COMPLETE SAS PROGRAM FOR THE APPLICATION

```
/*************************************************************************
 * MACRO NAME:    DDE
 * AUTHOR:        Zemin ZENG
 * DATE:          03/09/2011
 * SAS VERSION:   Version 9.0 or 9.1.3
 * PURPOSE:       To read the data specs to SAS datasets
 * PARAMETERS:    filepath-- the path of the folder containing list files
 *                outpath -- the path of the folder to store SAS datasets
 * CALL SAMPLE:   % DDT(filepath=C:\VBA\LST, outpath=C:\VBA\ddt)
 *-----------------------------------------------------------------------------
 * CAUTIONS:      this macro will be called by VBA_buttons_DDT utility in folder:
                  R:\Utility and could be run any computer with PC SAS installed
*-----------------------------------------------------------------------------*/
%macro tosas(filepath=, outpath=);
OPTIONS noxwait noxsync mprint;
libname dslib "&outpath";

data newfile(keep=nlst nfile fexten);
  length nlst nfile $100;
  rc=filename('id',"&filepath");
  dirid=dopen('id');
  numsel=dnum(dirid);
  do i=1 to numsel;
    nlst=dread(dirid,i);
    x=reverse(strip(nlst));
    nfile=reverse(substr(x, 5));
    fexten=upcase(scan(x, 1, '.'));
    if fexten='COD' then  do; nfile=substr(strip(nfile), 6); output; end;
  end;
  rc=dclose(dirid);
run;

proc sql noprint;
  select nfile into :flist separated by ','
  from newfile;
quit;

%let memnum = 0;
%do %until(%length(%qscan(%quote(&flist), %eval(&memnum+1), %str(,)))=0);
   %let memnum = %eval(&memnum + 1);
   %let mem    = %qscan(%quote(&flist), &memnum,  %str(,));
data _null_;
  call execute("filename cmds dde 'winword|" || '"' || "&outpath\_DDT_&mem..doc" || '"' ||
"!CC' notab;" );
run;

data &mem(rename=(f1=name f2=label f5=code f6=fmtname f7=comment));
  length f1 f2 f3 f4 f5 f6 $1000;
  infile cmds dlm='09'x notab /**/ missover dsd LRECL=8000;
  input f1 $ f2 $ f3 $ f4 $  f5 $ f6 $;
  if INDEX(upcase(f3), 'TYPE') and INDEX(upcase(f2), 'LABEL')then delete;
        label f1="Variable Name"
              f2="Variable Label"
              f3="Type"
              f4="Format Name"
              f5="Length"
              f6="Comments";
run;

data dslib.&mem;
  set &mem;
  length memname $20;
  memname="&mem"; memname=upcase(memname);
run;

filename cmds clear;
%end;
%mend tosas;
*to call the macro;
%tosas(filepath=U:\DDT, outpath=U:\DDT);
```

APPENDIX 2: KEY VBA CODE IN THE APPLICATION

```
'* The VBA subroutine to open each Word file to replace special character
'* and add bookmarks
Private Sub CommandButton3_Click()
   Dim path            As String, path1 As String
    Dim filename        As String
    Dim tabl As Range
    Dim atable As Table
    WordBasic.DisableAutoMacros True
    path = TextBox1.Value
    path1 = TextBox2.Value
    '*** Loop through all List documents and replace special character***
    Dim fname As String, fname1 As String
    filename = Dir(path & "\*.doc", vbNormal)

Do Until filename = ""
        On Error Resume Next
        fname = StrReverse(Mid(StrReverse(filename), 5)) & ".doc"
        fname1 = "_DDT_" & StrReverse(Mid(StrReverse(filename), 5)) & ".doc"
Dim newdoc As Word.Document

Documents.Open path & "\" & fname
ActiveDocument.Tables(1).Select
Selection.Copy
ActiveDocument.Close SaveChanges:=False
Set newdoc = Application.Documents.Add
newdoc.Bookmarks("\EndOfDoc").Range.Paste
Selection.WholeStory
'*** replace soft return (Shift+Enter)
With Selection.Find
 .ClearFormatting
 .Text = Chr(11)
 .Replacement.ClearFormatting
 .Replacement.Text = " "
 .Execute Replace:=wdReplaceAll, Forward:=True, _
 Wrap:=wdFindContinue
End With
'*** replace carriage return
With Selection.Find
 .ClearFormatting
 .Text = Chr(13)
 .Replacement.ClearFormatting
 .Replacement.Text = " "
 .Execute Replace:=wdReplaceAll, Forward:=True, _
 Wrap:=wdFindContinue
End With
Selection.WholeStory
With ActiveDocument.Bookmarks
.Add Range:=Selection.Range, Name:="CC3"
.DefaultSorting = wdSortByName
.ShowHidden = False
End With
newdoc.SaveAs filename:=path1 & "\" & fname1, FileFormat:=wdFormatDocument
newdoc.Close

 On Error GoTo 0
 filename = Dir()
Canceled:
    WordBasic.DisableAutoMacros False
    Application.DisplayAlerts = True
    Application.ScreenUpdating = True
 Loop
End Sub
```

```
'* The VBA subroutine to run the SAS DDE program for each Word file to read
'* Word table data
Private Sub CommandButton4_Click()
    Dim path1 As String
    Dim filename        As String
    WordBasic.DisableAutoMacros True
    path1 = UserForm.TextBox2.Value
    '*** Loop through all List documents ***
   filename = Dir(path1 & "\*.doc", vbNormal)

 Do Until filename = ""
   On Error Resume Next
   Documents.Open path1 & "\" & filename
 On Error GoTo 0
 filename = Dir()
 Canceled:
    WordBasic.DisableAutoMacros False
    Application.DisplayAlerts = True
    Application.ScreenUpdating = True
 Loop
   Unload UserForm
   Dim exesas As Long
   exesas = ShellExecute(0, "Open", _
   "\\frxwf002\vol2\Csdm-sysshare\SHAREALL\Utility\DataSpec2SAS\BATCH.bat", "", 0,1)
End Sub
```