Automated Generation of Clinical Study Reports

Suhas R. Sanjee, Merck Sharp & Dohme Corp., Upper Gwynedd, PA
Rajavel Ganesan, Merck Sharp & Dohme Corp., Upper Gwynedd, PA
Jason Clark, Merck Sharp & Dohme Corp., Upper Gwynedd, PA

ABSTRACT

The results of clinical / pre-clinical trials are presented to regulatory agencies in a Clinical Study Report (CSR). A CSR typically consists of a number of tables, listings and graphs each of which are stored as individual files. The process of incorporating these tables, listings and graphs in a CSR is laborious and potentially error prone. An 'automated' technique is presented in this paper which uses SAS® and Windows Scripting Host (WSH) to generate a CSR document populated with specified tables, listings and graphs. This way, parts of CSR can be generated by a simple call to a SAS macro. SAS and WSH can be used to automate routine tasks performed in MS Office. WSH is a powerful and comprehensive utility which facilitates the controlling of Microsoft applications such as MS Word programmatically. The WSH utility allows VBScript programs to run on the Windows operating system as stand-alone applications. The proposed technique uses SAS to generate the appropriate VBScript with parameters, which in turn populates the CSR with the specified tables, listings and graphs. Parameters can include specific folder location, the order and location of these tables, listings and graphs in the CSR.

INTRODUCTION

VBScript (Visual Basic Scripting Edition) is an active scripting language developed by Microsoft. It is a limited but free version of Microsoft's Visual Basic programming language. VBScript comes in handy as an automation tool and has been widely used by Windows users and administrators.

The Safety section, compared to Efficacy section of the CSR, tends to be relatively stable. A technique to automate certain sections of the CSR writing (especially safety section) using VBScript and SAS is discussed here. List of files that need to be inserted into the CSR along with the location of insertion within the template CSR document are specified in an Excel sheet which is passed into the SAS macro. The SAS macro populates a VBScript which generates the CSR populated with tables, listings and graphs. It is assumed that the graphs are embedded in a word document either .doc or rtf format. Section number and bookmarks can be used to specify the location of insertion of tables, listings and graphs.

VBSCRIPT PROGRAMMING OVERVIEW

An overview of some of the VBScript methods used here is presented below.

CreateObject()

The CreateObject() method creates an automation object of a specified type and returns a reference to it. This method is used here to create a MS Word object so that the template CSR can be controlled by VBScript through this object.

Syntax: CreateObject (servername.typename [, location])

Here, *servername* and *typename* are required parameters whereas location is optional. Parameter *servername* specifies the name of application that provides the object, *typename* specifies the type/class of the object and location specifies where to create the object.

E.g. Set objword = CreateObject("Word.Application")

The above script creates a MS Word object and returns a reference to it.

Open()

This method acts on the document collection object. It opens the specified document and adds it to the documents collection.

E.g. Set objDoc = objWord.Documents.Open("C:\ PharmaSUG\CSR_template.doc")

The above script opens the document "CSR_template" and adds it to the document collection of the MS Word object "objWord"

GoTo()

This method acts on the document selection object and returns a Microsoft.Office.Interop.Word.Range object that represents the start position of the specified item, such as a page, bookmark, section or field.

Syntax:

return = objSelection.GoTo(wdGoToBookmark,BookmarkName) - for navigation to bookmarks

return = objSelection.GoTo(wdGoToSection,wdGoToFirst,SectionNumber,Name) - for navigation across sections

Here, objSelection is a selection object.

InsertFile()

This method inserts all or part of the specified file.

Syntax: Expression.InsertFile(FileName, Range, ConfirmConversions, Link, Attachment)

Expression and Filename are required parameters whereas others are optional. The *expression* should return a Range or Selection object. FileName is the full path of the file to be inserted. If you don't specify a path, Word assumes the file is in the current folder. *LINK* specifies whether the file should be inserted as link or not. LINK should have a value of "True" to insert the file as a link. The parameters range, ConfirmConversions and Attachment are out of the scope of this paper.

TypeText()

This method acts on the selection object and inserts the specified text at the current cursor position of the word document.

Syntax: selection1.TypeText(text)

Here, selection1 is the selection object and text is the text to be inserted at the current location in the document.

SAS PROGRAMMING OVERVIEW

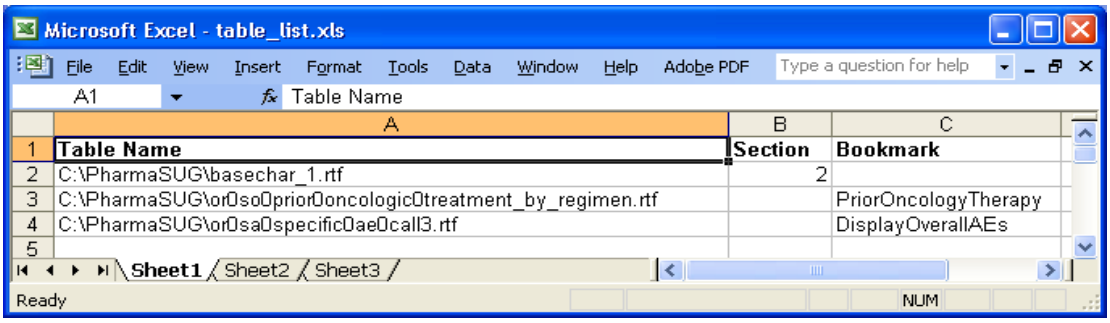
Te steps involved in the proposed technique from a SAS programming perspective are outlined here.

Step 1: Read Metadata Spreadsheet

Read the Excel file containing path of the files that are to be inserted and the location. Here location refers to section number or bookmark name which determines where in CSR the files are to be inserted.

Metadata excel sheet is shown in Figure 1. Either the section number or the name of the bookmark must be specified for each file. It is assumed that the bookmarks already exist in the template CSR. This excel sheet is read into SAS.

Figure 1: Snapshot of Metadata Excel Sheet



Step 2: Insertion of Files

The SAS code to generate the VBScript is shown in Figure 2. VBScript code is written to a file using the macro function *%STR*. A Word document is opened and the specified files are inserted by assigning appropriate values to object properties. For each record in the metadata dataset *path0lst*, GoTo() method is

invoked to navigate to the desired location of the Word document and the specified file is inserted using InsertFile method. Link = TRUE is used for InsertFile method to insert the file as a link so that the files can be refreshed to reflect the latest version. After inserting the last file in the metadata, the Word document is saved and closed.

Figure 2: Code Snippet to Generate VBScript for Inserting Files

```
data _NULL_;
  set path0lst end=eof;
  file "C:\PharmaSUG\&csr0name..vbs" lrecl=2048;
  *Initialization - Open the template CSR file;
  if _N_=1 then do;
    put "%str(Set objword = CreateObject%()" @ ;
    put '"Word.Application"' @; put "%str(%))";
    put "%str(objword.Visible = True)";
    put "%str(strComputer = %".%")";
    put "%str(Set objDoc = objWord.Documents.Open(%"C:\
          PharmaSUG\MockCSR.doc%"))";
    put "%str(Set objSelection = objWord.Selection)";
  end;

  *Insertion of files;
  *If section variable is not empty then insert the table in the
   specified section;
  if missing(section) ne 1 then
    put "%str(return=objSelection.GoTo%(wdGoToSection,wdGoToFirst,)"
        section "%str(,%"%"%"%))";
  else if missing(bookmark) ne 1 then
    put "%str(return=objSelection.GoTo%(-1,,,%")" bookmark "%str(%"%))";
  put "%str(returnsel=objSelection.InsertFile%(%")" table_name "%str(%")"
      "%str(,%"%"%", False, True, False%))";

  *Save the document and quit MS Word;
  if eof=1 then do;
    put "%bquote(objDoc.SaveAs("C:\PharmaSUG\&csr0name..doc"))";
    put "objword.Quit";
  end;
run;
```

Figure 3: VBScript Generated by the Code Snippet in Figure 2 with the Metadata in Figure 2

```
' Section 1: Instantiation of VBScript objects
Set objword = CreateObject("Word.Application")
objword.Visible = True
strComputer = "."
Set objDoc = objWord.Documents.Open("C:\PharmaSUG\MockCSR.doc")
Set objSelection = objWord.Selection

'Section 2: Insertion of files specified in the metadata
return=objSelection.GoTo(wdGoToSection,wdGoToFirst,2,"")
returnsel=objSelection.InsertFile("C:\PharmaSUG\basechar_1.rtf ","",False,
                                  True,False)

return=objSelection.GoTo(-1,,,"PriorOncologyTherapy ")
returnsel=objSelection.InsertFile("C:\PharmaSUG\or0so0prior0oncologic0treat
                                  ment_by_regimen.rtf","",False,True,False)

return=objSelection.GoTo(-1,,,"DisplayOverallAEs ")
returnsel=objSelection.InsertFile("C:\PharmaSUG\or0sa0specific0ae0call3.rtf
                                   ","", False, True, False)
'Section 3: Save the document and Quit MS Word
objDoc.SaveAs("C:\PharmaSUG\CSR_MK000_P000.doc")
objword.Quit
```

Figure 3 shows the VBScript generated by running the SAS code shown in Figure 2. Section 1 of the script instantiates VBScript objects and opens the template CSR – MockCSR.doc. The files specified in the metadata spreadsheet are inserted at specified sections and bookmarks using Goto() and InsertFile() methods (refer to Section 2 of Figure 3). "PriorOncologyTherapy" and "DisplayOverallAEs" are the bookmarks in the template CSR document. The document is saved and MS Word application is closed using SaveAs() and Quit() methods. Brief description of the VBScript methods can be found in VBScript Programming Overview section.

Step 3: Insertion of Boilerplate Text

Boilerplate text can be inserted to describe certain standard tables of the CSR. This is accomplished by using the TypeText method in VBScript. TypeText method inserts the text at the current cursor position of the document. Goto method is used to navigate to the desired location of the CSR before calling TypeText method.

E.g. put "%str(objSelection.TypeText %"Most frequently occurring Adverse Event is
Rash with the occurrence rate of 41 %")";

Here, the most frequently occurring adverse event (Rash) and its percentage of incidence (41%) are computed dynamically in the SAS program and passed into the TypeText method.

Step 4: Invoking VBScript from SAS

The generated VBScript can be invoked from SAS using the DOS command CSCRIPT and X statement as follows:

```
X "%str(cscript.exe %"C:\PharmaSUG\&csr0name..vbs %")";
```

CONCLUSION AND FUTURE DIRECTION

This technique has been successfully implemented on the Windows platform. The possibility of using this technique in UNIX has to be investigated. The same technique can be used to automate any routine task performed using MS Office and other Windows based applications.

It should be explored to include cross referencing and captioning features for the inserted tables. The files can only be inserted immediately after the section breaks or at bookmarks.

ACKNOWLEDGEMENTS

We would like to thank Gopal Rajagopal, Merck and Co. for his inputs. We would also want to thank our manager Mary Varughese for all the encouragement and support.

CONTACT INFORMATION

Author Name: Suhas R. Sanjee
Company : Merck Sharp & Dohme Corp.
Address : 351 Sumneytown Pike
City : North Wales State: PA Zip: 19454
Email: suhas_sanjee@merck.com

Author Name : Rajavel Ganesan
Company : Merck Sharp & Dohme Corp.
Address : 351 Sumneytown Pike
City : North Wales State: PA Zip: 19454
Email : rajavel_ganesan@merck.com

Author Name: Jason Clark, PhD
Company : Merck Sharp & Dohme Corp.
Address : 351 Sumneytown Pike
City : North Wales State: PA Zip: 19454
Email : jason_clark4@merck.com