PharmaSUG 2019 – Paper AD-94

A SAS® and VBScript Cyborg to Send E-mails

Nikita Sathish, Seattle Genetics, Inc., Bothell, WA

ABSTRACT

Clinical trial monitoring requires that SAS® programmers periodically generate statistical analysis reports and distribute the reports to cross-functional teams for review. Most reports are distributed via email or an enterprise content management system. Manual generation of email, however, can introduce errors. For example, recipients can be omitted, formatting errors can occur, and reports may not be sent in a timely manner. SAS® programmers can avoid these limitations by sending reports directly from SAS® to large, customized, email distribution lists.

SAS® programmers can further improve report distribution by combining SAS® with VBScript. VBScript allows programmers to overcome security concerns associated with SAS®, by not having to store login credentials or tweak the configuration file. We recommend that programmers send emails from SMTP server through SAS® using VBScript. This approach allows programmers to send emails with attachments, pass through SAS® macro variables to VBScript, and build dynamic email contents. In this paper, we detail this simple approach and present ways to send emails using PC SAS® and SAS® server.

INTRODUCTION

For any technology, automation applied to an efficient operation magnifies the efficiency. Automation ensures productivity and consistency in a business process. We have come up with this simple stand-alone approach to automate the process of sending e-mail using SAS® for Microsoft Windows.

SAS® supports three interface methods to send mail:

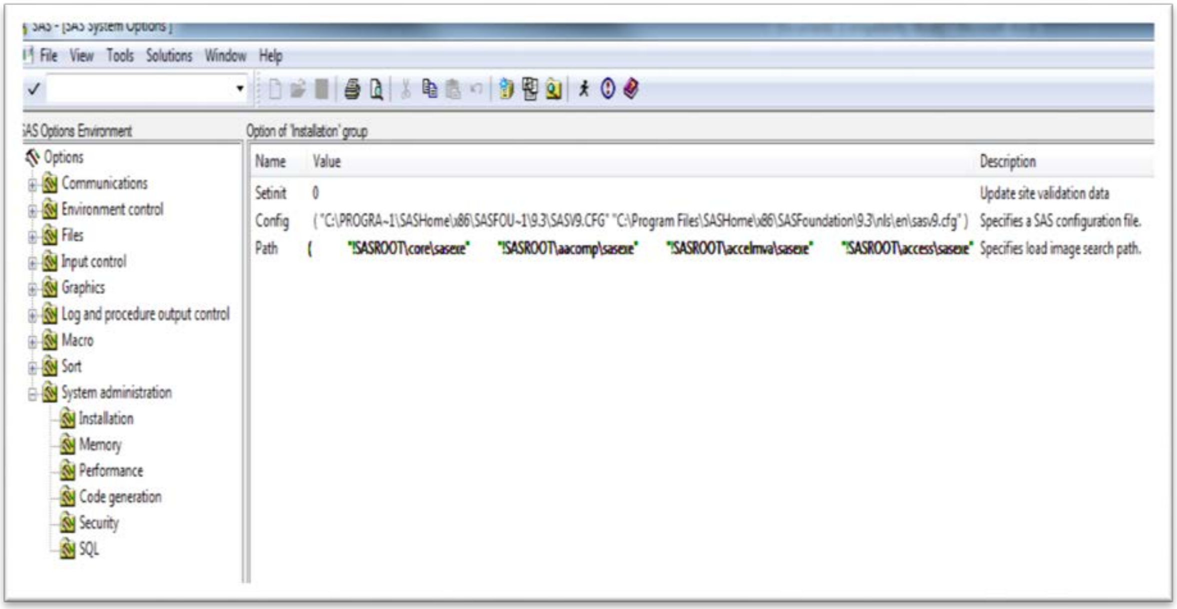
• MAPI: "Messaging Application Program Interface". This is the default and makes use of the mail client on your computer, e.g. Microsoft Exchange or Outlook or Mozilla Thunderbird.

• SMTP: "Simple Mail Transfer Protocol". With SMTP you bypass the mail client and you have to specify outgoing mail account settings, comparable to specifying the outgoing mail settings in Outlook or Thunderbird.

• VIM: "Vendor Independent Mail".

Since options are used in the initialization of the SAS® session, you have to specify them to define the mail interface in the SAS® configuration file (for SAS® 9: SASV9.CFG). This file is normally located in C:\Program Files\SAS\SAS® 9.1\nls\en.

Check the System Options in the Display Manager (Tools / Options / System Options / System Administration / Installation) to verify the location in your installation as shown below in Display 1.



Option of 'Installation' group

Name	Value	Description
Setinit	0	Update site validation data
Config	("C:\PROGRA~1\SASHome\x86\SASFOU~1\9.3\SASV9.CFG" "C:\Program Files\SASHome\x86\SASFoundation\9.3\nls\en\sasv9.cfg")	Specifies a SAS configuration file.
Path	("!SASROOT\core\sasexe" "!SASROOT\aacomp\sasexe" "!SASROOT\accelmva\sasexe" "!SASROOT\access\sasexe"	Specifies load image search path.

Display 1. To verify the location of SAS® configuration file

Sending mail through any of the three interfaces discussed above, requires the sender to input their username and password in the SAS® config file which compromises password security. This paper presents you with independent code to send emails through SAS® using VBScript without compromising password security.

THE CODE

Assign the required macro variables (&name &date &day &month &year) that should be passed to VBScript as parameters and create a format to represent the months of the year as below:

```
proc format;
  value $ month
    01="January"
    02="February"
    03="March"
    04="April"
    05="May"
    06="June"
    07="July"
    08="August"
    09="September"
    10="October"
    11="November"
    12="December";
run;

data test;
  infile dirlist length=reclen;
  input line $varying256. reclen;
  if index(line,"<filename>") and index(line,"<file format>");
  date=substr(strip(line),18,8);
  day=substr(strip(date),7,2);
  month=substr(strip(date),5,2);
  year=substr(strip(date),1,4);
  mon=put(month,month.);
  call symputx("name", strip(line), "g");
  call symputx("date", strip(date), "g");
  call symputx("day", strip(day), "g");
  call symputx("month", strip(mon), "g");
  call symputx("year", strip(year), "g");
run;

%put &name &date &day &month &year;
```

We have embedded a VBScript file in the SAS® code to generate email. The code in this VBScript file is provided in the next section.

```
%let allOK=1;
data _null_;
script = "<location of VBScript file>";
        fvar = catx('\',pathname('WORK'),'closeVBscript.txt');
        file script filevar=fvar;
      if &allOK = 1 then do;
             error='none';
      end;
      else do;
            error='error';
      end;

       runvb = 'cscript //nologo '|| quote(script)
                  || ' ' || quote(error)
                  || ' ' || quote(‘<List of email recipients’)
                  || ' ' || quote("<output file name>")
                  || ' ' || quote("&day &month &year")
                  || ' ' || quote("&name");
      /*Call VBScript and write Std Error to log*/
       infile retval pipe filevar=runvb lrecl=32000 end=eof;
       input;
       line=_infile_;
run;
```

VBSCRIPT CODE

This script allows you to take any HTML file and send it as the content of an email and include another file as an attachment. This script uses Collaboration Data Objects (CDO) to send mail and can be configured for authentication. CDO is an application programming interface included with Microsoft Windows and Microsoft Exchange Server products. The library allows developers to access the Global Address List and other server objects, in addition to the contents of mailboxes and public folders. CDO is a technology for building messaging or collaboration applications. CDO is not a part of the Outlook Object Model and it doesn't provide any event-based functionality, nor can Outlook objects be manipulated using CDO.

The input to the VBScript code contains the following:

String - Optional flag to alter the subject to inform recipients of an error

String - Full path to attachment

String - Semicolon-separated list of email addresses

STEP-BY-STEP APPROACH

Step 1: First we need to specify the VB arguments and build an Email object to pull the username from the network.

```
Set MyEmail=CreateObject("CDO.Message")
Set objNetwork = CreateObject("Wscript.Network")
```

Step 2: Get current date stamp for log file

```
y = year(now)
m = month(now)
d = day(now)
if m < 10 Then m = "0" & m end if
if d < 10 Then d = "0" & d end if
isoDate = y & m & d
```

Step 3: Confirm your script is called as expected

```
If args.length > 0 Then
      If args(0) = "error" Then subj_txt = "ERROR in calling program" End If
      If Not isEmpty(args(1)) Then
            toline = args(1)
      End If
      If Not isEMpty(args(2)) Then 'add attachment
                  MyEmail.AddAttachment args(2)
      End if
      If Not isEMpty(args(3)) Then
              subj_txt = " Periodic AE monitoring Using Excel - " & args(3)
        End if
        If Not isEmpty(args(4)) Then
                MyEmail.TextBody = "<your text>," & vbNewLine & vbNewLine &
"<your attachment>(" & args(4) & ")
        End if
End If
```

Step 4: Enter the email subject, attach username to the email domain and specify the To, Cc and Bcc parameters

```
MyEmail.Subject=subj_txt
MyEmail.From= objNetwork.UserName + "@mycompany.com"
MyEmail.To= toline
MyEmail.Cc = ""
MyEmail.Bcc = ""
```

Step 5: Create Body from SAS® DataSet to HTML

```
MyEmail.CreateMHTMLBody "<your file path>.html"
```

Step 6: The mechanism to send messages

```
cdoSendUsingPickup (1)
cdoSendUsingPort (2)
cdoSendUsingExchange (3)
```

In this approach, we use the Port:

```
MyEmail.Configuration.Fields.Item
("http://schemas.microsoft.com/cdo/configuration/sendusing")=2
```

Step 7: Once the messages have been picked up they are sent to an SMTP server, which delivers emails to the recipients

```
MyEmail.Configuration.Fields.Item
("http://schemas.microsoft.com/cdo/configuration/smtpserver")="sendsmtp.<doma
in>"
```

Step 8: We need to now specify the SMTP port number and send the email. SMTP acts just like a postman here to pick up and deliver the messages as shown in Figure 1 below

```
MyEmail.Configuration.Fields.Item
("http://schemas.microsoft.com/cdo/configuration/smtpserverport")=25

MyEmail.Configuration.Fields.Update

MyEmail.Send

set MyEmail=nothing
```

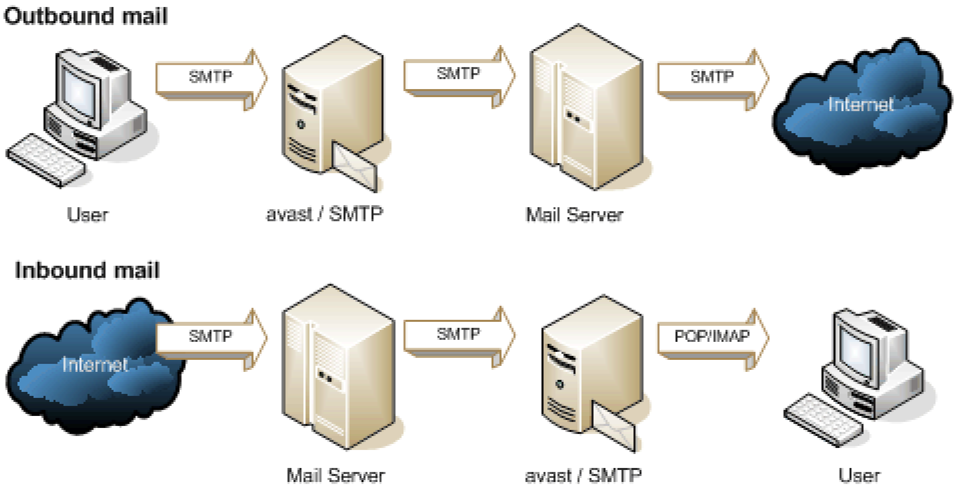


Figure 1. Role of the SMTP Server

SAS® SERVER CODE

Below is the code snippet to send email through SAS® server:

```
filename mailbox email;
data _null_;
   file mailbox to="&_metauser"; /* metauser will be you */
   subject="<your EMAIL subject>";
   put “<your EMAIL body>”;
run;
```

CONCLUSION

Utilizing the features of SAS® and VBScript saves time on redundant tasks, eliminates manual errors and enhances scope for innovation. This stand-alone approach of sending emails through SAS® is very simple, efficient and reliable.

REFERENCES

http://www2.SAS.com/proceedings/forum2008/038-2008.pdf

https://en.wikipedia.org/wiki/Collaboration_Data_Objects

http://www.serversmtp.com/en/what-is-smtp-server

ACKNOWLEDGMENTS

I would like to thank Shawn Hopkins and Gordon Fancher for their technical inputs to the paper. I am grateful for the insightful comments given by Yang Wang in reviewing the paper.

CONTACT INFORMATION

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

Name: Nikita Sathish
Enterprise: Seattle Genetics, Inc.
Address: 21823 - 30th Drive S.E. Bothell, WA 98021
Work Phone: 425-527-2790
E-mail: nsathish@seagen.com