Paper 217-27

A SAS/AF® Mainframe/PC File Transfer Application

Rob Nelson, Centers for Disease Control and Prevention, Atlanta, GA
Janet Royalty, Centers for Disease Control and Prevention, Atlanta, GA

ABSTRACT

SAS® systems at the Centers for Disease Control and Prevention operate on a variety of computing platforms. Most users prefer to use the SAS System for PCs for data analysis. However, many important data sources are located on CDC's MVS mainframe. We have developed a SAS/AF file transfer application as an alternative to other means of accessing SAS data across platforms. Other options could include remotely submitting mainframe jobs from the PC, assigning remote library names from the SAS System for PCs, creating transport files from the PC, or running SAS jobs from the mainframe. This application, which mimics the appearance of other file transfer applications, allows users to easily move SAS data between the mainframe and PC. It utilizes user supplied parameters, including mainframe account data and directory information for both PC and mainframe, to dynamically populate and display lists of SAS data sets and catalogs. Files from either platform can be selected and transferred to the other.

INTRODUCTION

Frequently, SAS users at CDC need to transfer data across platforms and operating systems. We have developed an application using SAS/AF and SAS/Connect® that allows users to upload or download SAS data sets between a MVS mainframe and local workstations operating on Windows NT.

This application will be presented in 5 loosely divided sections, one each for the four AF Frames and corresponding SCL and a section for the script file used by SAS/Connect. It is our intention to present this application in its entirety and in a manner that will allow readers to recreate the application.

CREATING AF FRAMES AND SCL

When executed, this application displays four frames. LOGON collects user ID and password for submission to the mainframe. MFACCT allows the user to input the mainframe account he or she wishes to explore. MFLIB displays the MVS sequential data sets that serve as SAS libraries in the selected account. MAIN displays the SAS data sets and catalogs in the selected library. The frames and their corresponding SCL are explored in the order in which they execute.

The layout of the application's first frame is illustrated below, in Figure 1. All frame objects are labeled with red numbers. Table 1 lists object name and type. During execution, user id and password are entered into this frame and then passed to the SAS/Connect script file (MF.SCR). The script file establishes a remote mainframe session. If a session is successfully started, the application passes control to the next frame, MFACCT.FRAME. Please refer to the SCL for the labeled sections corresponding to the objects btnOK, btnCancel, entUserid, and entPassword.

Figure 1 – LOGON

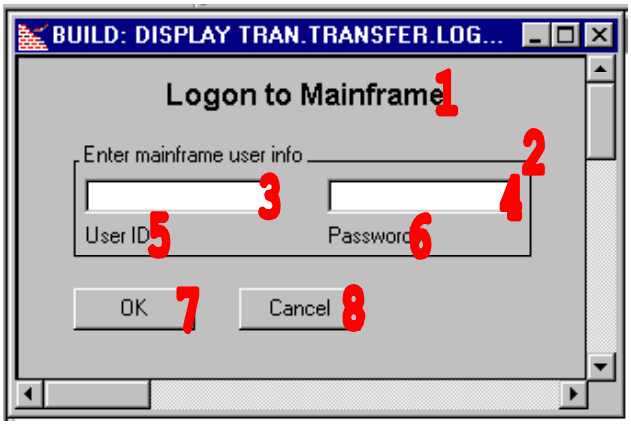

Table 1 – LOGON.FRAME Object List

No.	Name	Object type
1	LblTitle	Text label control
2	Containerbox1	Container box
3	EntUserid	Text entry control
4	entPassword	Text entry control
5	LblUserid	Text label control
6	lblPassword	Text label control
7	BtnOK	Push button control
8	BtnCancel	Push button control

LOGON.SCL

```
INIT:
DECLARE char userid password;
declare num onmf;
entUserid._cursor();
RETURN;

MAIN:
onmf=rlink('MFID');
if onmf =1 then goto acct;
RETURN;

BTNOK:
userid=trim(entuserid.text);
password=trim(entpassword.text);
if userid = '' then do;
  userlist=makelist();
  userlist=insertc(userlist,'Provide User ID');
  rc=messagebox(userlist);
  userlist=dellist(userlist);
  entUserid._cursor();
end;
else do;
  call symput ('userid',userid);
end;
```

```
if password = '' then do;
  passlist=makelist();
  passlist=insertc(passlist,'Enter Password');
  rc=messagebox(passlist);
  passlist=dellist(passlist);
  entPassword._cursor();
end;
else do;
  call symput ('password',password);
end;
SUBMIT CONTINUE;
GOPTIONS GACCESS=GSASFILE;
%LET MFID=999.999.999.99;
OPTIONS PS=55 LS=80 NOFMTERR COMAMID=TCP
REMOTE=MFID;
FILENAME RLINK 'c:\tran\mf.scr';
signon mfid;
ENDSUBMIT;
if rlink('MFID')=0 then do;
  errlist=makelist();
  errlist=insertc(errlist,
  'SAS log and try again.  ');
  errlist=insertc(errlist,
  'Please check userid, password, and');
  errlist=insertc(errlist,
  'Mainframe logon unsuccessful. ');
  rc=messagebox(errlist);
  errlist=dellist(errlist);
  btnOK._cursor();
end;
RETURN;

BTNCANCEL:
cancellist=makelist();
cancellist=insertc(cancellist,
'User ID and password must be provided');
rc=messagebox(cancellist);
cancellist=dellist(cancellist);
entUserid._cursor();
call execcmd('End');
RETURN;

ENTUSERID:
entpassword._cursor();
RETURN;

ENTPASSWORD:
cursor btnOK;
RETURN;

ACCT:
call display('mfacct.frame');
call execcmd('End');
RETURN;

TERM:
return;
```

Figure 2 shows the layout of the second frame, which is used to identify mainframe dataset search criteria. Object name and type are listed in Table 2. This frame accepts user input of a mainframe account name, which at CDC identifies the high-level node of the data set name. If a non-existent account name is entered, an error is returned. If a valid account name is entered, a member listing of libraries within the account is created and control is passed to the frame MFLIB.FRAME. The SCL for this frame contains a labeled section corresponding to the object btnOK. Please note that the commandOnClick property of the object Exit should be set to 'End.' .

Figure 2 - MFACCT

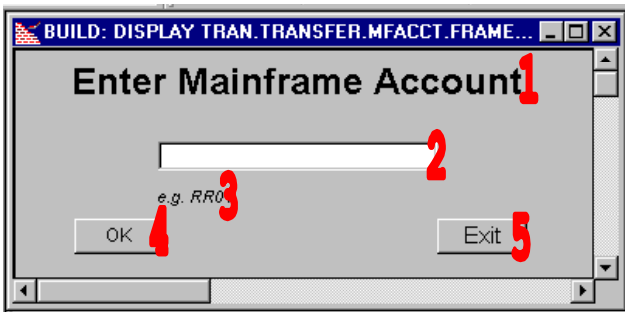

Table 2 – MFACCT.FRAME Object List

No.	Name	Object type
1	TxtTitle	Text label control
2	EntAccount	Text entry control
3	TxtExample	Text label control
4	LblDirlabel	Text entry control
5	LblDirdisplay	Text entry control
6	containerbox1	Container box
7	EntSearch	Text entry control
8	BtnSearch	Push button control

MFACCT.SCL

```
INIT:
RETURN;
BTNOK:
acct=trim(entacct.text);
acct2=trim(entacct.text);
call symput ('acct2',acct2);
if acct = '' then do;
  errlist=makelist();
  errlist=insertc(errlist,
  'A Mainframe Account must be provided');
  rc=messagebox(errlist);
  errlist=dellist(errlist);
  entacct._cursor();
  RETURN;
end;
else do;
  acct='"' || "'" || acct || "'" || '"';
  call symput ('acct',acct);
  password="'" || symget('password') || "'";
  userid="'"||symget('userid')||"'";
  rc=libname('perm',
  'c:\tran');
  rc=delete('perm.mffiles');
  SUBMIT CONTINUE;
  libname perm 'c:\tran';
  filename dir ftp '' list host='999.999.999.99'
  pass=&password user=&userid cd=&acct;
  data perm.mffiles(keep=name type line);
    infile dir length=linelen;
```

```
    input line $varying500. linelen;
    namelen = linelen - 40;
    name=substr(line,57,namelen);
    if name=' Dsname' then delete;
    if substr(line,1,3)='Mig' then
     type='MIGRATED';
    else if substr(line,35,2)='FS' and
     substr(line,53,2)='PS' then type='SAS';
    else if substr(line,35,2)='FB' and
     substr(line,53,2)='PS' then type='PDS';
    else if substr(line,35,1)=' ' and
     substr(line,53,1)=' ' and  index(upcase(
     line),'TAPE') then type='TAPE';
  run;
  proc sort data=perm.mffiles;
    by descending name;
  run;
  ENDSUBMIT;
  dsid=open('perm.mffiles','i');
  if fetchobs(dsid,1) NE 0 then do;
    errlist=makelist();
    errlist=insertc(errlist,
     'was provided.  Please try again. ');
    errlist=insertc(errlist,
     'An invalid mainframe account ');
    rc=messagebox(errlist);
    errlist=dellist(errlist);
    entacct._cursor();
    * conditional return;
    rc=close(dsid);
    RETURN;
  end;
  rc=close(dsid);
  call display('mflib.frame');
  call execcmd('End');
end;
RETURN;
```

The frame layout and object list for the third frame are given below in Figure 3 and Table 3. This frame lists the SAS Libraries in the account chosen in the previous frame. When a library is chosen in this frame, a list of its contents (data sets and catalogs) is created and control is passed to the next frame, MAIN.FRAME. The objects btnOK and btnMfacct have labeled sections within the SCL. As with the previous frame, the commandOnClick property of the object Exit should be set to 'End.'

Figure 3 - MFLIB.FRAME

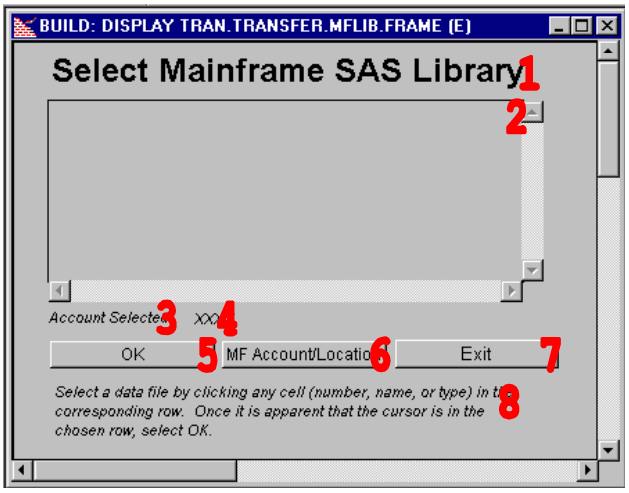

Table 3 – Object List

No.	Name	Object type
1	LblTitle	Text entry control
2	TblData	Data table
3	LblSelected	Text label control
4	LblAcct	Text label control
5	BtnOK	Push button control
6	BtnMfacct	Push button control
7	BtnExit	Push button control
8	LblComment	Text label control

MFLIB.SCL

```
INIT:        declare list mffiles;
declare num nlevels;
declare char acctexis;
declare char acct;
declare object datable;
length datname $80;
_frame_=_frame_;
_frame_._get_widget_('TBLDATA',datable);
txtAcct.label=" ";
if symget('acct') = '' then do;
  call display('mfacct.frame');
  call execcmd('End');
end;
else do;
  link poplbox;
end;
RETURN;

MAIN:
RETURN;

BTNOK:
  datable._getcolumntext('name',datname);
  IF datname='' then do;
    errlist=makelist();
    errlist=insertc(errlist,
    'A mainframe SAS library must be selected');
    rc=messagebox(errlist);
    errlist=dellist(errlist);
    btnmfacct._cursor();
```

```
    RETURN;
  end;
  else do;
    acct=symget('acct');
    if indexc(acct,"'") NE 0 then do;
      pos1=(indexc(acct,"'")) + 1;
      acct=substr(acct,pos1,50);
      pos2=indexc(acct,"'") - 1;
      acct=substr(acct,1,pos2);
    end;
    call symput('lib',' ');
    lib= "'" || left(trim(acct)) || "." ||
    left(trim(datname)) || "'";
    call symput('lib',lib);
    call execcmd('End');
  end;
RETURN;

BTNMFACCT:
rc=delete('perm.mffiles');
datable._setDataset(' ');
call display('mfacct.frame');
link poplbox;
RETURN;

POPLBOX:
acct=symget('acct');
if indexc(acct,"'") NE 0 then do;
  pos1=(indexc(acct,"'")) + 1;
  acct=substr(acct,pos1,50);
  pos2=indexc(acct,"'") - 1;
  acct=substr(acct,1,pos2);
end;
txtAcct.label=left(acct);
rc=open('perm.mffiles','U');
dat1=dsid('perm.mffiles','U');
datable._setDataset('perm.mffiles');
datable._displayColumnLabel('name','type');
datable._setDisplayedColumns('name','type');
rc = CLOSE(dat1);
datable._setColumnAttribute('name',
'COLUMN_WIDTH',100);
datable._setColumnAttribute('type',
'COLUMN_WIDTH',50);
RETURN;
```

The fourth and final frame is shown below in Figure 4, and object name and type are given in Table 4. SAS data sets and catalogs can be transferred from this frame by selecting files individually in the origin library, and then pressing the “Upload” or “Download” button, as appropriate. Labeled sections exist in the frame’s SCL for the objects btnDownload, btnUpload, btnLoclib, and btnMflib. Again, the commandOnClick property of the object Exit should be set to ‘End.’

Figure 4 - MAIN.FRAME

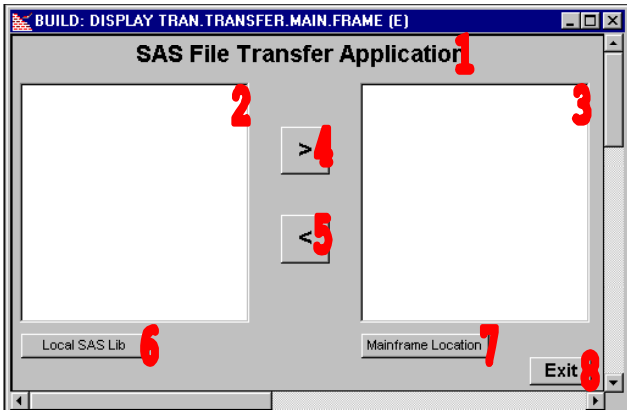

Table 4 – Object List

No.	Name	Object type
1	lblTitle	Text label control
2	lbxLocsas	List box control
3	lbxMfsas	List box control
4	btnUpload	Push button control
5	btnDownload	Push button control
6	btnLoclib	Push button control
7	btnMflib	Push button control
8	btnExit	Push button control

MAIN.SCL

```
INIT:
dcl list sasfiles, num rc posa posb posc posd
pos1 pos2, char textvar dnsel dnlib dnfil
dnlibstr uplib up down upsel upfil uplibstr
locengine mfengine;
* Define & clear macro variables;
call symput('acct',' ');
call symput('lib',' ');
call symput('password',' ');
call symput('userid',' ');
lbxlocsas.title=" ";
lbxmfsas.title=" ";
call display('logon.frame');
LINK LIBRARY;
RETURN;

MAIN:
RETURN;

BTNLOCLIB:
selection=liblist();
lbxlocsas.title=selection;
sasfilelist1.library=selection;
sasfilelist1.typeFilter='Catalog Data';
lbxlocsas.model='sasfilelist1';
logclass=loadclass('sashelp.fsp.logical.class');
logobj=instance(logclass);
locmembers=makelist();
call send(logobj,'_get_members_',locmembers,
"type='library'","name engine");
do i=1 to listlen(locmembers);
  a=getiteml(locmembers,i);
  locindex=searchc(a,selection);
```

```
  if locindex NE 0 then locengine=getitemc(a,2);
end;
locmembers=dellist(locmembers);
if sasfilelist1.library=' ' then do;
  lbxlocsas.title=' ';
  dnliblist=makelist();
  dnliblist=insertc(dnliblist,
  'No local SAS library selected');
  rc=messagebox(dnliblist);
  dnliblist=dellist(dnliblist);
  btnLoclib._cursor();
  RETURN;
  end;
RETURN;

BTNMFLIB:
call display('mflib.frame');
LINK LIBRARY;
if sasfilelist2.library=' ' or
libref('MFREMOTE') > 0 then do;
  upliblist=makelist();
  upliblist=insertc(upliblist,
  'No mainframe SAS library selected');
  rc=messagebox(upliblist);
  upliblist=dellist(upliblist);
  btnMflib._cursor();
  RETURN;
  end;
lbxmfsas._refresh();
RETURN;

BTNUPLOAD:
if sasfilelist2.library=' ' or
libref('MFREMOTE') > 0 then do;
  upliblist=makelist();
  upliblist=insertc(upliblist,
  'Select a mainframe library');
  rc=messagebox(upliblist);
  upliblist=dellist(upliblist);
  btnMflib._cursor();
  RETURN;
end;
if lbxlocsas.selecteditem=' ' then do;
  upfillist=makelist();
  upfillist=insertc(upfillist,
  'Select a SAS file to upload');
  rc=messagebox(upfillist);
  upfillist=dellist(upfillist);
  btnloclib._cursor();
  RETURN;
end;
dnsel = upcase(lbxlocsas.selecteditem);
dnlib = upcase(left(sasfilelist1.library));
dnlibstr=upcase(left(trim(sasfilelist1.library))
) || ".";
uplib = upcase(left(sasfilelist2.library));
posa=length(trim(dnlibstr)) + 1;
dnfil=upcase(substr(dnsel,posa,60));
if index(dnfil,".DATA") NE 0 then do;
  posb=index(dnfil,".DATA") - 1;
  dnfil=substr(dnfil,1,posb);
   down = dnlib || "." || left(trim(dnfil));
   up   = uplib || "." || left(trim(dnfil));
   SUBMIT CONTINUE;
   data &up;
   set &down;
   run;
   ENDSUBMIT;
end;
else if index(dnfil,".CATALOG") NE 0 then do;
  posb=index(dnfil,".CATALOG") - 1;
  dnfil=substr(dnfil,1,posb);
  catlib=left(trim(symget('lib')));
  down = dnlib || "." || left(trim(dnfil));
  up   = left(trim("upcat")) || "."
  || left(trim(dnfil));
  SUBMIT CONTINUE;
  RSUBMIT;
  libname upcat &catlib;
  proc upload incat=&down outcat=&up;
  run;
  ENDRSUBMIT;
  ENDSUBMIT;
end;
sasfilelist2.library=uplib;
lbxmfsas._refresh();
RETURN;

BTNDOWNLOAD:
if sasfilelist1.library=' ' then do;
  dnliblist=makelist();
  dnliblist=insertc(dnliblist,
  'Select a local SAS library');
  rc=messagebox(dnliblist);
  dnliblist=dellist(dnliblist);
  btnLoclib._cursor();
  RETURN;
end;
if lbxmfsas.selecteditem=' ' then do;
  dnfillist=makelist();
  dnfillist=insertc(dnfillist,
  'Select a SAS file to download');
  rc=messagebox(dnfillist);
  dnfillist=dellist(dnfillist);
  btnMflib._cursor();
  RETURN;
end;
upsel = upcase(lbxmfsas.selecteditem);
uplib =
upcase(left(trim(sasfilelist2.library)));
uplibstr=upcase(left(trim(sasfilelist2.library))
) || ".";
dnlib =
upcase(left(trim(sasfilelist1.library)));
posc=length(trim(uplibstr)) + 1;
upfil=upcase(substr(upsel,posc,60));
if index(upfil,".DATA") NE 0 then do;
  posd=index(upfil,".DATA") - 1;
  upfil=substr(upfil,1,posd);
  up   = uplib || "." || left(trim(upfil));
```

```
  down = dnlib || "." || left(trim(upfil));
  SUBMIT CONTINUE;
  data &down;
  set &up;
  run;
  ENDSUBMIT;
end;
else if index(upfil,".CATALOG") NE 0 then do;
   if locengine='V6' and mfengine='REMOTE6'
   then do;
     englist=makelist();
     englist=insertc(englist,
     'local library or contact SDMB  ');
     englist=insertc(englist,'local library.
Please select a version 8');
     englist=insertc(englist,'mainframe catalog
to version 6 or earlier');
     englist=insertc(englist,'Unable to transfer
version 6 or earlier  ');
     rc=messagebox(englist);
     englist=dellist(englist);
     btnLoclib._cursor();
     RETURN;
   end;
posd=index(upfil,".CATALOG") - 1;
upfil=substr(upfil,1,posd);
catlib=left(trim(symget('lib')));
up   = left(trim("upcat")) || "." ||
left(trim(upfil));
down = dnlib || "." || left(trim(upfil));
SUBMIT CONTINUE;
RSUBMIT;
libname upcat &catlib;
proc download incat=&up outcat=&down;
run;
ENDRSUBMIT;
ENDSUBMIT;
end;
sasfilelist1.library=dnlib;
lbxlocsas._refresh();
RETURN;

TERM:
submit continue;
signoff mfid;
ENDSUBMIT;
RETURN;

LIBRARY:
lib=symget('lib');
rc=libname('MFREMOTE');
SUBMIT CONTINUE;
libname MFREMOTE &lib server=mfid;
ENDSUBMIT;
if libref('MFREMOTE') > 0 then do;
  liblist=makelist();
  liblist=insertc(liblist,
  'file type.  Please check SAS log. ');
  liblist=insertc(liblist,
  'library or inconsistent mainframe ');
  liblist=insertc(liblist,
  'Error caused by migrated mainframe');
  rc=messagebox(liblist);
  liblist=dellist(liblist);
  sasfilelist2.library=' ';
  lbxmfsas.title='ERROR';
  lbxmfsas._refresh();
  RETURN;
end;
sasfilelist2.library='MFREMOTE';
sasfilelist2.typeFilter='Catalog Data';
lbxmfsas.model='sasfilelist2';
lbxmfsas.title=lib;
lbxmfsas._refresh();
logclass=loadclass('sashelp.fsp.logical.class');
logobj=instance(logclass);
mfmembers=makelist();
call
send(logobj,'_get_members_',mfmembers,"type='lib
rary'","name engine");
do i=1 to listlen(mfmembers);
  a=getiteml(mfmembers,i);
  mfindex=searchc(a,'MFREMOTE');
  if mfindex NE 0 then mfengine=getitemc(a,2);
end;
mfmembers=dellist(mfmembers);
RETURN;
```

RLINK PROGRAM

The SAS/Connect script file follows. More than the AF code of this application, the structure of the SAS/Connect script file is site and operating system specific. In this example, the script file is stored in the directory C:\TRAN. Please refer to the SAS Online Documentation for information on writing SAS/Connect script for your system.

MF.SCR

```
if tcp goto onoff ;
  /* if EHLLAPI goto onoff ;
if IRMA goto onoff ;
if CXI goto onoff ;  */
log 'ERROR: This RLINK file is for
OPTIONS=EHLLAPI, IRMA, or CXI.';
abort;

ONOFF:
if signoff goto signoff;
log 'scanning for Enter Userid . . .' ;
waitfor 'IKJ56700A ENTER USERID -': cont, 20
seconds;
STOP;

CONT:
type "&userid" LF ;
snapshot;
scanfor 'ENTER CURRENT PASSWORD ': pswd1a,
10 second;
STOP;

PSWD1A:
type "&password" LF ;
```

```
log 'scanning for the Ready prompt . . .' ;
scanfor 'READY' : startsas, 30 second ;
goto menu ;

MENU:
log 'scanning for the Netpass Application Menu .
. .' ;
scanfor 'MAINFRAME APPLICATION MENU' : pickTSO,
1 second ;
goto tsomenu ;

PICKTSO:
type 'TSO' enter;
waitfor 'Main Application Menu' : optionT, 30
seconds ;
STOP;

TSOMENU:
log 'scanning for the TSO Main Menu . . .' ;
scanfor 'Main Application Menu' : optionT, 1
seconds ;
goto cmdline ;

OPTIONT:
log 'waiting for the TSO command prompt . . .' ;
type 'T' enter ;
waitfor 'Enter TSO command' : rlink, 5 seconds ;
STOP;

CMDLINE:
log 'scanning for the TSO command prompt. . .' ;
scanfor 'Enter TSO command' : rlink, 1 seconds ;
STOP;

RLINK:
log 'executing the SAS version 6.09 PCLINK
command . . .' ;
type "free fi(sysproc)" LF ;
type "alloc fi(sysproc)" ;
type "  da('sys1.uads', 'tso.prod.clistlib',
'tso.prod.helplib'," ;
type "     'tso.user.clistlib') shr"  ;
type LF ;

waitfor 'READY' : startsas, 10 seconds ;
STOP;

STARTSAS:
type '%SAS ';
type " O('DMR,COMAMID=tcp,tcpipprf=tcpip, ";
type "noterminal,no$syntaxcheck')" LF;
waitfor 'SESSION ESTABLISHED', 120 seconds :
end;

log 'Note: Your micro-to-host link has been
started normally.' ;
STOP;

END:
STOP;

SIGNOFF:
scanfor 'Enter TSO command', 15 seconds ;
type 'end' enter;
scanfor 'Main Application', 15 seconds;
type 'X' enter;
scanfor 10 seconds;
STOP;
```

CONCLUSION

This application illustrates a method of providing SAS users with an easy tool to transfer SAS data across operating systems. Additional features for this application might include providing transfer of non-SAS files, the ability to create new SAS libraries, and additional error trapping.

REFERENCES

SAS Institute Inc. (1999) , *SAS OnlineDoc® Version 8*., SAS Institute Inc., Cary, NC: SAS Institute Inc.

CONTACT INFORMATION

Contact the author at:

Rob Nelson
Centers for Disease Control & Prevention
Mailstop E-48
1600 Clifton Road
Atlanta, GA 30333
E-mail: RNELSON1@CDC.GOV