Mainframe Files On Your Windows Desktop with SAS/CONNECT®

Michael G. Sadof, MGS Associates, Inc.

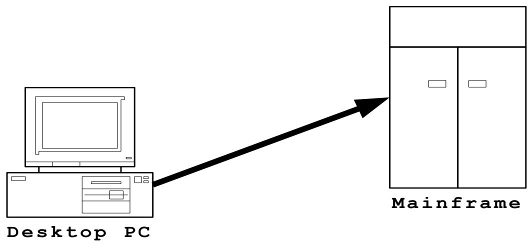
ABSTRACT

Since it is common to have data residing across several platforms, it is important to utilize the strengths of each computer system. SAS/CONNECT® Software enables you to view and update your data on multiple platforms from any one connected computer. This paper provides some quick and easy code to get your SAS/CONNECT applications up and running smoothly.

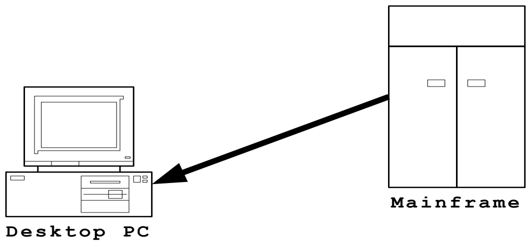
This paper will focus on mainframe to Windows connectivity. With only a few simple statements you can manipulate your mainframe based data from your familiar SAS windows environment. The SAS/CONNECT software must be installed on each platform. This paper will describe some tweaks for the standard SAS connection script used to connect to mainframe and describe a simple template for your SAS/Connect sessions. SAS/Connect features Remote Library Services, Remote Transfer Services, and Remote Compute Services enabling us to harness the power of the mainframe without resorting to the archaic JCL syntax.

DEFINITIONS

First, let us start with some definitions. The Local Computer or Local Host Computer is the computer to which you are initially signed on. For instance if you utilize PC SAS and you open up a SAS session then your PC SAS session is the Local Host. If you use your PC to log into a UNIX SAS session through terminal emulator software then your Local Host is the UNIX session. When you use either of these local SAS sessions (interactive or batch) to sign on to another SAS session on a different platform that is the Remote Host or Remote Server. When we send data from the Local Computer to the Remote Host we are uploading. When we are copying data from a Remote Computer or Internet server to our local machine we are downloading. The PROC DOWNLOAD procedure will move stored data from the Remote Host to your Local Session. The PROC UPLOAD procedure will move locally stored data to the remote host. In this paper we will be starting SAS on the Windows Platform and Connecting to a remote mainframe server.



PROC UPLOAD



PROC DOWNLOAD

FIVE EASY STEPS

There are essentially five steps in every SAS/CONNECT session which I will summarize here but leave the reader to explore elsewhere so I can focus on the automated mainframe connect.

Step 1 - Establish the communication access method. You need to know the IP address or the DNS name of the server you are connecting to. If this code does not work try it without the quotation marks surrounding the macro variable &node.

```
%LET node=123.2233.444.333;   ** or maybe %let node=myzos.myco.com;
OPTIONS COMAMID=TCP REMOTE="&node";
```

Step 2 - Identify the RLINK script file. I will explain this in more detail later.

An example of the RLINK file identification is as follows where the '!sasroot' has been identified in the configuration file when SAS was installed:

```
filename rlink "!sasroot\connect\saslink\mytcptso.scr";
```

The script for linking to a mainframe TSO session is 'tcptso.scr'. Check your "!sasroot\connect\saslink" directory for further options. The script file is a text file containing comments and commands to log on to the remote computer. I have provided sample modifications to the rlink script and named it 'mytcptso.scr'. To adjust this file you must first set 'echo on;' and 'trace on;'. You will find these as the first two lines of the script and you may uncomment those lines. These commands within the file will produce traces and echoes to the SAS log so you can determine where the problem lies. In my experience the most problems occur with the timing so you may want to bump up some of the 'wait' times. Another area of concern is the command that actually starts SAS on the remote machine. You will note that the 'tcptso.scr' script file (for zOS) a command similar to the following will actually start SAS:

```
type "sas o('dmr,comamid=TCP,noterminal,no$syntaxcheck')" LF;
```

You may have to adjust this command to point to your actual SAS executable file on the mainframe as described in attached logon script. There are two implementations of the SAS Connect software on the mainframe. One is through the regular TSO logon, the other is through a background process running known as the SAS-Spawner which sits and waits for a SAS command. This is more popular on Unix based systems and is not usually seen on zOS platforms. The script file is actually an automated way of typing the user responses to the logon sequence. Once you study the script file you will get the hang of it.

Step 3 - Signon

The actual signon is accomplished with the signon statement and is the statement in your SAS program that initiates the script in the RLINK file. It is handy to code it with the node name as follows:

```
   SIGNON "&node";
```

Step 4 - Remote Submit your programs and procedures.

```
   RSUBMIT;
      ****;
      * SAS statements;
      ***;
   ENDRSUBMIT:
```

All statements within the RSUBMIT block will be executed on the remote machine.

Step 5 – Sign off

To end to remote session the signoff procedure is as simple as coding the SIGNOFF command as follows or it may be typed into the command line and executed that way:

```
   SIGNOFF;
```

Full Program:

```
   %LET node=mvs1;             ** or maybe %let node=myzOS.myco.com or ip address;
   OPTIONS COMAMID=TCP REMOTE=&node; ** need quotes if you use IP address *;

   FILENAME rlink "!sasroot\connect\saslink\tcptso.scr";

   SIGNON &node;

   RSUBMIT "&node";
      LIBNAME  myremote "XYZ.YOUR.ZOS.SAS.FILE";
   ENDRSUBMIT;

   * Now on the local Session *;
   LIBNAME newremote
     SLIBREF=myremote
     SERVER="&node";

   PROC CONTENTS DATA=newremote._all_;

   SIGNOFF;
```

An explanation of the above code is as follows:
The OPTIONS statement will assign the TCP communications access method and name the remote session to be `'mvs1'`. This must be a valid alias for the particular server's IP address. The FILENAME statement will select the logon script file to be used. The SIGNON statement will begin execution of the RLINK script file. After execution of the SIGNON statement you will see the logon messages from the remote computer and you will in fact be logged on to the remote SAS session as well as your local session. The RSUBMIT command will indicate that all of the statements until the ENDRSUBMIT command will be executed on the remote host. The LIBNAME statement will allocate a remote library to the remote session. It will not be available to your local session until you ENDRSUBMIT and go back to

the local session and then execute the last LIBNAME statement. This final LIBNAME statement will assign a library name of `'newremote'` to you local session but it will point to the `'myremote'` library on the remote host. The SLIBREF= clause must point to the remote name and the remote server. At this point you may use PROC COPY or any other SAS procedure to copy or manipulate the data on your local machine. You must remember, however, that the data is residing on the remote machine and network traffic might delay a procedure. Using this method you will be able to process remote data on your local machine as opposed to submitting commands to process the data on the remote machine. Your local SAS session will retrieve data from the remote computer as necessary to process SAS statements. This processing is done outside of the RSUBMIT block as shown in the above example. . One added benefit of SAS/CONNECT is that is can read and write both version 6.12 and version 8 SAS files. You can connect from a Version 8 system directly to a version 6.12 system and the translation will be transparent to the user.

SCRIPTS AND SAMPLE PROGRAMS

I have included a sample script and a sample program as an appendix which will attempt to logon to your mainframe from your PC. It will prompt for the username and password. It will then assign a local library name to a remote SAS library residing on the mainframe. You will then be able to use the libraries as if they were on the PC with the standard V8 or V9 windows interface. You can store code in catalogs, data in SAS data files, Formats in format libraries etc. The script is labelled "mytcptso.scr" and is initially provided by SAS but may need to be modified. The SAS program "zos_connect.sas" will be more familiar to base SAS users and is described above.

In the code example and accompanying SAS log you will see SAS messages for logon to first the PC and then the zOS system.

CONCLUSION

SAS/CONNECT Software can assist in distributing the processing load on your computer network. You may use these services to connect to less heavily loaded computers on your network and obtain faster results. It enables you to view remote data easily from within your local SAS session or copy files reliably. SAS/CONNECT gives you the flexibility to control where you data is stored and processed. It enables you to work with a more familiar interface whether it is in the PC, UNIX or Mainframe batch environment. It is particularly handy when SAS data is spread across multiple platforms and you would like to work with it in one program. The Remote Library services, Remote Compute services, and the File Transfer services bring it all together in one nice package known as SAS/CONNECT.

REFERENCES

SAS Institute Inc., SAS/CONNECT® Software: Usage and Reference, Version 6, Second Edition, Cary, NC: SAS Institute Inc., 1994

TRADEMARKS

APPENDICES

mytcptso.scr	- Contains Logon Script
zOS_connect.sas	- SAS Code
zOS_connect.log	- Log of Sample Connection

AUTHOR CONTACT INFORMATION

The author is happy to assist in your implementations and welcomes, comments, questions, and suggestions. He may be reached at:

Michael G. Sadof
MGS Associates, Inc.
Bedford, NH
mgs@mgsnet.net