Paper 34-26

CONNECTing with SAS® Version 8: Virtually Transparent to the User

Lawrence Altmayer
U.S. Census Bureau

ABSTRACT

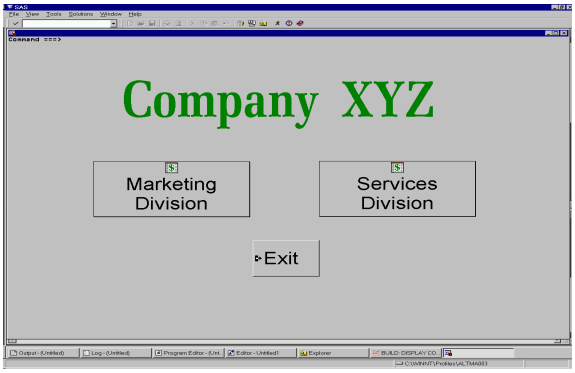
Whether used to connect from home to office or personal computer (pc) platform to mainframe network, it is straightforward with SAS version 8. We know it can be more efficient in many ways to store large data sets in a mainframe environment, and access them from the pc. This paper looks at using pc SAS/CONNECT® to access SAS course data sets from a UNIX® mainframe, a concept readily extended to larger data sets in practice. We will also touch on the basic configuration for accessing a SAS/AF® application when clicking on a SAS icon on a Windows® desktop. Next we look at ways to verify the data are being read from the UNIX mainframe. Finally, we make a small change to the properties of the main menu 'Exit' push button to immediately return to the desktop, upon exiting the application.

We will also see that it is easy to convert current version 6 applications to version 8, using Destiny course materials as a starting point. Although version 8 contains many new object-oriented features, this paper will focus on converting what we already have in version 6, directly to version 8 using PROC COPY, with the version 6 legacy object concept.

INTRODUCTION

Now that SAS version 8 is upon us, it is time to take advantage of its new features, and translate what we have from version 6 into the new environment. It is not as hard as it may seem. This paper gives a "walk-through" of applying a traditional client/server SAS/CONNECT application to a SAS/AF example, the "Company XYZ" application developed in the SAS Institute course, "SAS/AF Software: Changes and Enhancements in Version 8". The paper also shows how a version 6 SAS/AF example, "Bettabuys Supermarkets" from a previous Destiny course in FRAME development, is readily brought into version 8, using PROC COPY.

THE "COMPANY XYZ" EXAMPLE



In practice, although we may be using the pc version of SAS, it is usually more practical to put the large data sets we deal with in a

mainframe client/server environment, where there is more space. In many cases, a subset of the data will be accessed at any given time, rather than the entire data set. That is not the case in this paper; here, we access an entire data set, although the methods described here can be extended to the situation where a subset is called.

The example given here, the "Company XYZ" application, is from the SAS Institute course mentioned above. (The above screen has been modified for easier viewing.) The application consists of several FRAMES, and accompanying SAS component language (SCL, new with version 8, with similarities to version 6's screen control language), and data sets. The exercise described in this paper consists of getting the application to work on my pc in the economic directorate area of the Bureau of the Census. Next, the data sets are copied to one of the the area's UNIX mainframe machines, using transport files. Here, this was done by the Census SAS support staff; it is easily accomplished with SAS/CONNECT, if the remote machine's internet protocol (ip) address is known. Finally, I make changes to the SAS autoexec and configuration files to enable a direct SAS/CONNECT session with the UNIX machine involved. I also make changes to the properties of the SAS Windows desktop icon to use these autoexec and configuration files when clicking on the icon. Details follow.

Autoexec File

The autoexec file can contain any SAS language statement. It is usually used for such things as specifying SAS options, and initializing libraries with libname statements at SAS invocation, so this doesn't have to be done at the beginning of each session. It can also be used to control the application work space (aws). Here, in addition to these things, we incorporate SAS/CONNECT statements into the autoexec file, so we can start a client/server session with the UNIX mainframe machine, immediately upon going into SAS. The autoexec file used is shown in Appendix 1.

The autoexec file statements needed to control our aws are an 'options' and a 'dm' statement. The option noawsmenu removes the menu from the bar above the FRAME, to maintain control over the user's control of the screen. The 'awsmaximize' option causes the screen to be maximized with the aws.

The main statements needed for the SAS/CONNECT session are an 'options' and 'filename' statement, as well as two 'libname' statements to indicate the librefs associated with the two remote directories containing data sets used. The options statement remote= indicates the remote machine we will connect to. This can use an ip address, although many sites have an alias set up which can be used here. A comamid= portion of the options statement indicates the type of link we use to establish to the remote host. Here it is TCP.

A filename statement is used to specify a script needed for this type of link, which must be used for signing on. The rlink reserved argument in the filename statement indicates such a script will be used here. Minimal change in the default 'tcpunix.scr' was required on my part to allow connecting to the UNIX mainframe machine I used for this example. Finally, the libname statements for the 'mktgdata' and 'servdata' librefs show that the server used for the remote connection in our steps6 machine. This is an alias on our system, and can be a remote session id or

server name.

Configuration File

The system default configuration file is also edited to allow for direct invocation of the SAS/AF application, upon initializing our SAS session. This requires the addition of two statements to the configuration file: an -initcmd and a -nodmsexp statement. The second statement is needed to enable the first, and is placed at the bottom of the file, below the box warning that the INSTALL Application edited below that part.

The -initcmd statement is the first statement in the config file, but can be placed anywhere. In this case, it is used along with several additional commands, to control the screen area accessible to the user. The 'zoom' command maximizes the given window, 'toolclose' closes the toolbar, and 'command close' removes the command bar. The first few lines of the configuration file, showing the -initcmd statement, are given in Appendix 2.

Verifying Data Access

The easiest way to verify the data are being accessed from the UNIX mainframe (especially if we have similar librefs on both machines) is to remove libname statements for the local directories from our autoexec file for our local session. We can also remove the 'noawsmenu' option from the autoexec file, to allow us to view the 'explorer' window during our session. Clicking on 'General attributes' within 'Properties' for the desired libname shows the UNIX directory we are reading from. Be sure to replace the 'noawsmenu' option to the autoexec file for the final application.

Main Menu 'Exit' Push Button

If we leave the application as it is at this point, when we click on the 'Exit' push button on the main menu, we'll leave the application, but will remain in the SAS System. Then, we would have to click on 'Exit...' on File in the upper left corner symbol of the screen to exit the SAS session completely and return to the desktop.

To avoid this extra click, all we have to do is modify the properties of the 'Exit' push button in the mainmenu.frame. We do this by going to the Properties Window for the push button, and clicking on Attributes, Behavior for the push button. Looking at the Attribute name, commandOnClick, we see the default value is 'end.' If we change this value to 'bye,' we will immediately leave the SAS System upon clicking on the push button.

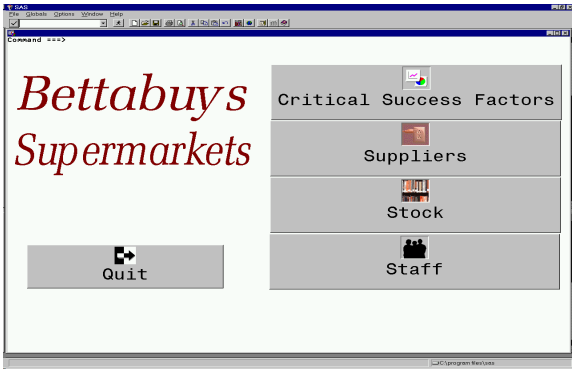
Desktop Icon Properties

In order to enable the features shown above in the autoexec and configuration files, we have to 'point' to them when invoking SAS. We do this by changing the properties of the icon on our Windows desktop, in the shortcut, target entry field. Whenever one invokes a SAS session from the command prompt of the machine they are using, he/she can use any of several system options as part of the execution statement, following the SAS executable file. This principle also applies to the shortcut, target entry field in the icon properties. After specifying the SAS executable file (sas.exe, including its complete pathname), we add the -autoexec and -config options to this command, to indicate the path for the new autoexec and configuration files we want to use for our session. This may differ from system to system; if you cannot find it, check

with your system administrator as to where they are. It is also a good idea to create a new autoexec and configuration file, leaving the originals intact, just in case a back-up file is needed at some point.

The statement used in this entry field is given in Appendix 3.

THE "BETTABUYS SUPERMARKETS" EXAMPLE



Transforming version 6 applications to version 8 is easy using PROC COPY. This does not take advantage of new object-oriented features of SAS/AF, such as SAS Component Language and other application development features, but it allows immediate use of version 8 with the present application. (Again, this screen has also been modified for easier viewing.) In moving this example to version 8, I found I had to "play" with the FRAMEs a bit to get them to look right after copying them. Upon copying to version 8, screen objects are treated as version 6 "Legacy" objects, with attributes as they had in version 6. These attributes can be displayed in familiar attributes Windows, or using screen control language code to define attributes, as done with version 6. Upon copying, catalogs and data sets have new .sas7bcat and .sas7bdat extensions, respectively, in their new libraries.

The PROC COPY code I used to copy the catalog and data sets from version 6 to 8 is shown in Appendix 4. It begins with two libname statements, one for the version 6 directory, with a 'v6' engine option on the libname statement. It should be noted that if the data set and catalog names will not change, the user must specify different directories for source and target, since a file cannot be copied into itself. The general format of the copy procedure is similar in some respects to that of another procedure, PROC DATASETS. It differs in that both a source and target (in= and out=) libref are used in the PROC COPY statement. It is similar in that the member type (e.g., catalog, data set) is specified in that statement. Selected members are included in a separate SELECT statement in the procedure.

CONCLUSION

The process of accessing data sets in a client/server environment using SAS/CONNECT is much the same in version 8 as it is in version 6. This is also the case with configurations needed (e.g., autoexec and configuration files) for directly accessing a SAS/AF application.

Copying both SAS/AF applications and data sets is also easy, using PROC COPY from the version 8 session, to bring over the necessary catalogs and data sets.

REFERENCES

1. “Company XYZ” example from SAS Institute course, “SAS/AF Software: Changes and Enhancements in Version 8,” ©1999 SAS Institute Inc., Cary, NC 27513, USA. All rights reserved.

2. “Bettabuys Supermarkets” example from Destiny Corp. course, “Using the FRAME Entry,” ©1996 Destiny Corp. All rights reserved.

3. SAS/CONNECT Software: Usage and Reference Version 6, Second Edition, ©1994 SAS Institute, Cary, NC, USA. All rights reserved.

AUTHOR

Lawrence Altmayer
U.S. Census Bureau
ESMPD, Room 1223-4
Washington, DC 20233-0001
(301)457-2581
Lawrence.W.Altmayer@census.gov

Appendix 1

```
libname co_xyz   'c:\temp\nfra\appl';
dm 'awsmaximize on';
options remote=steps6 comamid=tcp;
filename rlink 'c:\winnt\system32\tcpunix.scr';
signon;
options ls=132;
libname mktgdata '/export/home/altma003/temp/nfra/mktgdata' server=steps6;
libname servdata '/export/home/altma003/temp/nfra/servdata' server=steps6;
```

Appendix 2

```
-initcmd "af c=co_xyz.appdev.mainmen.frame; zoom; toolclose; command close;"

/* set default locations                                              */
-fontsloc "!sasroot\core\resource"
-TRAINLOC ""
-EMAILSYS VIM
-EMAILID ""
.
.
.
.

-nodmsexp
```

Appendix 3

"C:\Program Files\SAS Institute\SAS\V8\SAS.EXE"
-AUTOEXEC "C:\WINNT\SYSTEM32\AUTOEXV8.SAS"
-CONFIG "C:\Program Files\SAS Institute\SAS\V8\SASV8new.CFG"

Appendix 4

```
libname c v6 'c:\sas';
libname d    'c:\My Documents\My SAS Files\V8';
proc copy in=c out=d
memtype=catalog;
select bettab1;
run;
proc copy in=c out=d
memtype=data;
select staff stock;
run;
```