

Setting Up a Client Server Environment for End Users Using SAS/Connect

Howard Levine, American Express Financial Advisors

Introduction

One of the advantages of client server processing is that the strengths of available platforms can be utilized as effectively as possible to create the best possible systems. For example, it is possible to take advantage of GUIs on a PC and also utilize the data throughput capabilities of mainframes.

The purpose of this paper is to provide an example of how SAS/Connect was implemented at one site in order to provide background for someone who wishes to set up SAS/Connect applications at their own site. It provides some insights and experiences that are a useful supplement to the documentation provided by SAS Institute and your TCP/IP vendor(s).

The areas that will be covered are establishing communications between platforms using TCP/IP, determining where data will be located, determining where processing will take place, using SAS/Connect in production jobs, end user techniques utilizing SAS/Connect, and training and support issues encountered.

The Environment

At American Express Financial Advisors (AEFA), there are mainframe systems running the MVS and VM operating systems. There are also Tandem, Sun/UNIX, and RS6000/AIX distributed systems. The desktop environment is primarily OS/2, but a project to migrate

desktops to Win 95 and Win NT is underway. Novell Netware is used for file and print services on the PC network.

SAS is installed on one of our MVS systems and on OS/2 PCs in my unit. My unit uses SAS for capacity and performance management analysis. This includes using MICS and MXG, which are commercial products written in SAS, on MVS. One of the advantages of using SAS on OS/2 is the ability to easily create graphs and take advantage of the more sophisticated display manager capabilities available in a GUI compared to a text based TSO interface. The availability of performance data and SAS on the PC makes the work of analysts easier and faster. Performing analysis on the PC also reduces the demand for MVS processing power.

The particular system I will be describing is called the XYZ System. (Names are being changed to protect the innocent.) The XYZ System takes data that was entered into a Sybase database, compares it and matches it to data in an MVS SAS library, performs calculations to create new analysis datasets, and transfers the analysis datasets to the PC for use by an analyst.

In addition to accessing just the data in the XYZ System's datasets, the analyst must also access other data as well which might be located on either a PC or MVS. This means that the analyst needs SAS/Connect. The processes that create the PC files for her use SAS/Connect, but since the PC files created by the system are not always sufficient for analysis being done, ad hoc access to MVS

data must be provided to effectively support analysis.

An overview of the environment is shown in Figure 1.

The XYZ System will illustrate how a production environment can use SAS/Connect. Ad hoc use of SAS/Connect will also be described to support an end user environment.

Establishing Communications Between Platforms using TCP/IP

This is, without a doubt, the most difficult part of the entire process. It is essential that you identify all of the TCP/IP software on all of the platforms you will be using. This includes the name of the software, and the version, release, and maintenance level of the software. Take nothing as a given and check all of these details because one wrong detail will cost hours or days of time in wasted effort and delays.

Ensure that TCP/IP software is installed properly on all platforms. For server platforms such as UNIX and MVS systems, it will be necessary to work with a systems programmer or networking specialist. On the PC platform, the software should either be already installed or should be easy to install by yourself. You may need assistance from a PC or networking specialist.

Some important steps in MVS are making sure that the appropriate TCP/IP libraries are available to SAS for execution, setting the default access method to TCP (SAS option COMAMID=TCP), and appropriately setting the default for the TCPIPPRF option which identifies the location of the TCP/IP libraries SAS needs. These defaults should apply to both interactive and batch sessions.

It is also important to have a second TSO userid so that you can run a TSO session and use a SAS/Connect session at the same time. This is not important for SAS/Connect to work, but it will keep end users (including yourself) more productive. If people are forced to logoff TSO to use SAS/Connect in conjunction with TSO, they will not use SAS/Connect to access mainframe data.

On the PC, make sure the TCP/IP software is installed and configured correctly. Ideally, you should set up your PCs to be both clients and servers. The client part is fairly straight-forward, but the server capability means that your PC must be able to be a TELNET server - not just a TELNET client. You can use SAS/Connect effectively with a PC being used only as a client, but you are limiting yourself and are unable to fully take advantage of the distributed processing capabilities of the SAS System. This may be suitable for an end user, but will not be acceptable for PCs used for running or testing production systems.

The first tests that should be performed are simply logging into one computer from another using TELNET. Since SAS/Connect uses TELNET to establish the communications link, if TELNET does not work, the SAS/Connect TCP/IP connection will not work. You should be able to login from any machine in your production environment to any other machine in that environment. With OS/2 desktop systems, this means that an OS/2 machine must be able to login to other OS/2 machines and to MVS. Also, an MVS user must be able to login to OS/2 machines.

Once a simple TELNET session can be established, it is possible to try using SAS/Connect. SAS documentation will provide the appropriate information for the particular environments you are using. The

scripts provided by SAS Institute for logging onto various platforms are usually very close to what you will need in your environment. You will probably need to make some minor modifications, but they should be fairly simple. The best way to figure out what modifications are necessary is to login using TELNET and observe the prompts that are generated during the login process. Then, change the script to match what you see during your login processes.

An excerpt of a script is shown in Figure 2. A REXX command which executes the SASDMR command is shown in Figure 3. This command is executed by the script in Figure 2. The reason for using a REXX command is that the peculiar features of a particular PC can be handled in the REXX command by having each PC have its own version of the REXX command. Scripts will just have to look for the DOSASDMR command and not worry about how a particular PC is configured.

A problem encountered with using TELNET is that TELNET executes a LOGIN command for OS/2 when OS/2 is used as the remote host. Since PCs were already configured to use the LOGIN command to login to a Novell file server, this caused problems with the TELNET login process. The solution used was to rename the Novell LOGIN command to another name to permit the TELNET login to be properly executed.

Another problem was that the IP address alias set on the OS/2 machines was a number. This caused the OS/2 platforms to fail as TELNET servers. Adding a letter as the first character of the alias solved this problem. Apparently a number was translated into an (incorrect) IP address instead of simply being regarded as an alias.

Once you have established a SAS/Connect remote session, you should test it by doing remote submits, uploads, downloads, and use Remote Library Services (RLS) to access and edit a dataset on the server. Using RLS is the ultimate test that peer-to-peer communications have been established because all of the other SAS/Connect services do not require peer-to-peer communications.

Looking at the items in Figure 4 may be helpful for troubleshooting connection problems. SAS Technical Support is also very helpful.

When you successfully reach this point, you have overcome your technical infrastructure issues and are now ready to set up your applications.

Determining Where Data Will be Located

This can sometimes be a very simple issue. If someone else tells you where the data must be located, you may not have much to think about. However, if there are any intermediate processes requiring intermediate datasets, then you will have to determine the best place to put that data. As a general rule, you want to move data as little as possible. This means that if two datasets from different platforms must be used together, it is generally better to move the smaller dataset to the platform that has the larger dataset.

The output datasets of the process should be placed on the platform that will access the data. If necessary, multiple copies of the data may be put on various platforms to speed access by different users. For example, you could put a copy of certain datasets on each person's PC, on several

Novell file servers, or on several SAS/Connect servers.

Determining Where Processing Will Take Place

Processing should take place where the data is located. Unnecessary transfers of data create useless network overhead and can slow down overall processing.

If data is needed from two different platforms for a particular process, it is sometimes possible to either summarize or subset the data required from one of the datasets for the joining process. The summarization and subsetting can take place where the data is originally located with only the data absolutely needed for the join actually being moved to another platform.

It is generally advisable to move all data to one common platform before it is joined together in any processing. This insures that you can move only the data absolutely needed, that data can be moved efficiently using bulk file transfers, and that joining processes can be run more quickly.

The main idea is to do processing as close to the original data as possible in order to minimize file transfers.

Using SAS/Connect in Production Jobs

Using SAS/Connect in production jobs allows flexible use of SAS on the platforms that are best for the task. Advantages of using MVS are high data throughput, storage management that is sophisticated and reliable, mature job scheduling software, a mature production control system, and reliable data backups for error correction and

disaster recovery. Advantages of PCs are inexpensive storage, inexpensive CPUs, the ability to minimize network traffic during peak times, and a GUI environment.

For production batch jobs, SAS on MVS is the client and SAS on the PC is the server. An MVS batch job logs on to a PC and initiates a remote SAS session. The mainframe session controls what runs on the PC by remote submitting code. Generally, code should be located where it is executed. Code run remotely on another platform should be stored on that platform and executed by using a %INC or invoking an autocall macro. This keeps network traffic to a minimum and speeds execution.

All major data management functions such as large printouts, specialized output such as microfiche, backing up data, and maintaining archives are done using MVS. Generation data groups can be utilized effectively in batch processing to simplify the administration of a system. This technique has some advantages over aging SAS datasets using PROC DATASETS with regard to simplicity of data libraries and flexibility of storage management within the operating system. It is also possible to effectively use tape devices for mass storage very easily - a feat that would be difficult in a PC-only environment. To keep systems simple, a copy of the datasets that are transferred to PCs are also put into a SAS library on MVS. This permits quick "restores" of the PC files from MVS in the event that a file is lost on a PC. Also, access to data on MVS can be helpful when analysts are away from their workstations, but still need to examine the data.

Processing for the XYZ System

The XYZ system must join Sybase data to SAS datasets on MVS in order to allow further processing which allows files on a PC to be updated to support capacity planning analysis. The process is run from an MVS production batch job. This is done because the production control facilities in MVS are far superior to what is available for OS/2 (or most other PC and UNIX platforms). The first thing the batch job does is start a remote SAS session on a production PC. Code is remotely submitted to the PC to extract desired data from Sybase tables and transfer that data to MVS as SAS datasets.

The Sybase data is then joined with much larger SAS datasets already located on MVS. This joining occurs on MVS. The results of this joining process are refreshed files for the PC.

This refreshed PC data is written to a new generation of a generation data group (GDG) on MVS and then transferred to the PC using PROC UPLOAD (remember, the PC is the server and MVS is the client). The reasons for writing the data to a GDG before transferring it to the PC are that (1) damaged PC files can easily be restored by simply transferring the file again without worrying about PC file backups and (2) the data is available on MVS in case someone needs to access it there instead of on a PC.

Currently, the data is written to a Novell file server, but the data will also be written to a SAS/Connect server PC platform to facilitate more effective end user access. The data is then used by capacity planning analysts for modeling, graphing, and reporting utilizing SAS for OS/2.

Some explanation as to why processing is done this way is called for. For example,

why is Sybase data extracted using a PC and then transferred to MVS? Why not just transfer directly from Sybase on UNIX directly to MVS? The main reason is that SAS/Access for Sybase is less expensive to license on the PC than on MVS. Also, the ability to access Sybase data from a PC for analytical work is already necessary, so the SAS/Access for Sybase on OS/2 is already a "sunk" cost from that perspective. Finally, the volume of Sybase access we require for MVS SAS processing is very small at this time and can be handled very adequately with a PC.

Most of the processing - the joining of data - occurs on MVS because that is where most of the data is located.

The data to be transferred to the PC is first stored in a GDG. This makes it much easier to archive data than could be done with PC software. It is also more flexible from a storage management and maintenance standpoint than aging SAS datasets using PROC DATASETS. On MVS, the data is reliably backed up and the MVS data can be used to refresh damaged data on the PC. Any PC that is not a Novell file server is not backed up, so the data protection features are very important to us since we are writing data to machines that are not backed up. It is also possible to look at the data when only mainframe access is available such as when dialing in from a remote location.

Analysts prefer working on PCs rather than the mainframe when possible in order to take advantage of the user friendliness of PCs. Producing color graphs and attractive reports is much easier on a PC than in TSO. It is also usually much easier to print impressive output using PC printers than mainframe printers.

End User Techniques Utilizing SAS/Connect

From an end user perspective, use of SAS/Connect makes accessing data on other platforms much easier. Using file transfers, remote submits, and remote library services (RLS), it is possible to utilize mainframe resources while using the GUI interface on a PC. Often, an analyst will remotely submit code to subset or summarize a large mainframe dataset. Then, the result will be downloaded for reporting, graphs, and more detailed analysis.

A common use of RLS is to quickly and easily get variable lists from datasets and to examine a few records from a dataset just to understand what the data looks like. It is also possible to edit data using PROC FSEDIT when utilizing RLS.

Training and Support Issues Encountered

The best way to implement a system is to make it so simple and easy that nobody even realizes it is there, but they still benefit by it. The second best is to require small changes in end user behavior, but provide good benefits. Both of these approaches are being used.

With production jobs, the end users are not concerned about how the work is done as long as the work is completed successfully. If the files they use in their PC SAS sessions are created with batch jobs on MVS rather than manually submitting SAS statements, they simply do not care. As long as they have the functionality previously available, they are happy.

On user work stations, switching from EHLLAPI to TCP/IP was done by changing

users' config.sas and autoexec.sas files. The only substantial difference noticeable was that an Attachmate EXTRA! session was not used for the SAS/Connect remote session. File transfers were faster with TCP/IP also. Some users were willing to learn about RLS and took advantage of it for inspecting SAS dataset variable names and observation variable values.

Current production requirements call for SAS datasets to be placed in directories on Novell file servers so the data can be accessed by many users. The next step in fully utilizing SAS/Connect is to have a production PC with SAS/Connect set up as a SAS/Connect server. Rather than transferring datasets to a Novell file server, the files will be transferred to the SAS server which is an OS/2 PC. End users will then establish a remote SAS/Connect session on the SAS server to obtain access to the data rather than accessing data on a Novell file server. The benefit of doing this is that data can be processed on the server as well as the client. This reduces network traffic and processing time, especially when the server is a more powerful computer than the client. Even in a worst case scenario where an end user is using RLS for all data access on the SAS server rather than doing remote submits, the situation will be no worse than with a Novell file server because data will still be passed to the client for processing. When RLS is used appropriately for browsing and editing datasets, performance will usually be better than with a Novell file server if an end user knows how to use the WHERE statement and WHERE command. WHERE processing takes place on the server rather than on the client.

The reason for using a PC server rather than using MVS as a server for all situations is that small files can be processed by analysts less expensively on PCs. Also, having a

dedicated computing resource is really useful during MVS peak demand periods when immediate processing is desired.

Using a PC as a dedicated SAS/Connect server requires some training and adjustment by end users. Since MVS is still used as a remote server, end users must have a simple way of logging into MVS with SAS/Connect while still logged into the SAS/Connect PC server. Also, it is necessary to use RSUBMIT and ENDRSUBMIT statements and specify the remote session to submit statements to. Since there are multiple remote SAS sessions, there is not a remote session that SAS statements are always, consistently submitted to unless the user specifies the session on the RSUBMIT statement. This requires end user training and a willingness to change work habits in exchange for a more effective data processing environment.

Some changes can be handled for the end user. For example, end users have SAS libraries and filenames automatically defined for them in their autoexec.sas files. These file definitions can be changed from accessing Novell file server directories to accessing directories on the SAS/Connect server using RLS. Also, the autoexec.sas file on the SAS/Connect server can be configured to use LIBNAMEs and FILENAMEs for the remote session that are the same as what the end users previously used. This allows the same code used to access data on a Novell file server to be used to access data on the SAS/Server by using a submit and using RLS or, preferably, using remote submits to put the processing on the SAS/Connect server.

Conclusion

Using SAS/Connect provides considerable flexibility in determining where to locate both data and processing. This is useful in production environments for getting the most cost effective use of hardware and software. It also permits data to be positioned where it needs to be used. This can be done during off-peak times when computing and network capacity are both less expensive. Data can then be accessed quickly and efficiently using the best computer platforms to get the job done.

The best way to use SAS/Connect is to utilize TCP/IP to get the advantages of peer-to-peer communications and faster throughput. Allowing any machine to be either a client, server, or both greatly expands the capabilities of a computing environment. Analysts can work more quickly and effectively. By positioning data on platforms other than the standard production environments used to operate the business, it is possible to get effective computing power without adversely affecting computer systems used to directly service customers.

Acknowledgements

Special thanks to Fran from SAS Technical Support who was amazingly helpful when I was setting up SAS/CONNECT with TCP/IP. She pointed me in the right direction more times than I (want to) remember.

Thank you to the Capacity & Performances Services unit at AEFA for helping me to get the client/server environment implemented and used.

Author Contact Information

Howard Levine

e-mail	howard.levine@aexp.com
Phone	612-671-2799

References

SAS/CONNECT Software Usage and Reference Version 6 Second Edition, SAS Institute Inc.

Client/Server Computing with the SAS System, SAS Institute Inc.

Trademarks

SAS, SAS/Connect, and SAS/Access are registered trademarks of SAS Institute Inc.

Sybase is a registered trademark of Sybase Inc.

OS/2, TSO, VM, MVS, and IBM are registered trademarks of IBM Corporation.

Win 95 and Win NT are trademarks of Microsoft Corporation.

Attachmate and EXTRA! are registered trademarks of Attachmate Inc.

Novell and Netware are registered trademarks of Novell Corporation.

Figures

Figure 1

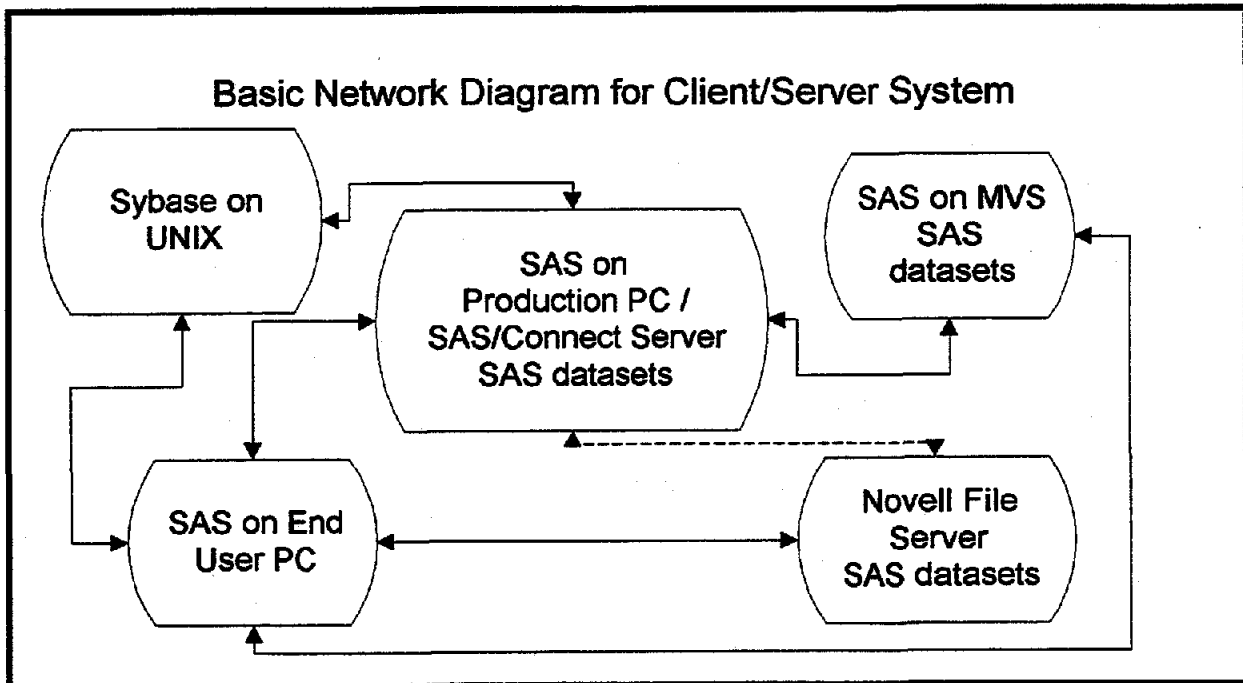


Figure 2

Excerpt from SAS/Connect Script used for logging onto OS/2

```

waitfor 'password', 20 seconds : nopass;
/* password cannot be input for batch - this should be used when prompting for the password is desired. */
/* input nodisplay 'password';
/* type CR; */
/* this is the password - do not use this line when the user is supposed to enter the correct password. */
type 'PASSWORD' CR;
/* wait for OS/2 prompt in TELNET session */
waitfor '.', 20 seconds : nouser;
log 'NOTE: Logged on to OS/2.... Starting remote SAS now.';
/* dosasdmr is a REXX command for bringing up a remote SAS session on OS/2 using the SASDMR command. */
type 'c:\dosasdmr' CR;
/* use a long wait time in case there are delays in getting a session started */
waitfor 'SESSION ESTABLISHED', 300 seconds : nosas;
log 'NOTE: SAS/CONNECT conversation established';

```

Figure 3

REXX Command File for Executing SASDMR

```

/* */

parse arg parms

'c:\sas\SASDMR.exe -DMR      -remote " " -NOXWAIT -NO$SYNTAXCHECK  -config
c:\sas\config.sas -nobatch' parms

```

Figure 4

SAS Options and Settings to Watch For

Option or Setting	Location
COMAMID=TCP	In SAS configuration files on all SAS platforms - it should be the default.
TCPIPPRF=prefix	MVS SAS configuration file. the correct prefix should be set as the default to identify the TCP/IP library.
-SET vqpname x	config.sas file on OS/2.
NO\$SYNTAXCHECK	Use whenever starting a remote SAS session at session invocation. The SAS/Connect script should cause this to be used.
NOXWAIT	Use at SAS invocation when starting a remote session on a PC or UNIX platform.