Distributed Processing: Using Remote Library Services to Bring Storage to the Workstation

Daryl Hoffman
Resource Administration Group
Center for Academic Computing
Penn State University

Overview
This poster paper will attempt to describe how SAS/SHARE®, SAS/CONNECT® and remote library services can be used to create a remote or client/server system for delivering information across TCP/IP from Windows 3.1 to VM/ESA 1.1.

This poster paper will touch on the problems encountered and the techniques discovered to create a client/server system at the Center for Academic Computing of Penn State University.

Introduction
Client/server has become the new 'buzzword' of the early '90s. With SAS® moving in the direction of providing a distributed information system, the mainframe is becoming more of a large data handler or server and the workstation is becoming the way to access the information stored there. A workstation may be a 486 machine or it could be a small Unix or VMS workstation.

The Center for Academic Computing at the Pennsylvania State University is beginning to see this change become a reality. Currently the academic accounting system and data reside on an IBM mainframe running SAS® 6.08 on CMS. With the move toward distributed computing and with workstations competing with the mainframe for better means of displaying and reporting, the move to client/server is becoming an important part of the evolution of the accounting system. The Accounts office has begun to investigate the feasibility of moving the information system software, written in SAS/AF®, to the Windows 3.1 platform, using SAS® and SAS/CONNECT®. The data will continue to reside on the mainframe because of the size of the databases, but the information delivery will be done on the PC running version 6.08 TS 405 of the SAS® system.

The feasibility study looked at all of the problems encountered in attempting to port the code to the Windows platform. Other problems were also encountered with SAS/CONNECT and TCP/IP. Currently the port of the information system is continuing and testing is still in progress. This paper will leave the conclusions and decisions up to the reviewers of the paper.

What is Remote Library Services?
According to the preliminary documentation, remote library service is a SAS System tool that allows for the development of intelligent client/server applications. Remote library service allows the programmer to define a library reference, libref, on a remote platform and access the data transparently from the local host. Remote library service is made up of a remote engine and a server.

The Move to Distributed Computing
We have been hearing of the move to downsize from all sides. Businesses are attempting to cut costs by downsizing manpower. Universities are attempting to cut costs as well by removing the dependence on large mainframe or mini-computers. The workstation market is beginning to fill the gap of processing power for a fraction of the cost and allows the user to have the entire machine in their office or even on their desktop.

Our academic department has began to move in this direction and has just recently removed the MVS operating system from the academic mainframe. It appears that sometime within the next 5 years that the other operating system currently running on the IBM ES/9000, unning VM/ESA, will also be phased out. As VM/ESA is phased out, another operating system will need to replace it, and current operating systems being looked into include Unix, AIX, OS/2 and Windows. It is for this reason that we have
started to move the applications and processing of information off of the mainframe and onto a workstation. The current platforms being investigated by our group are Windows for the local host and VM/ESA as the database server. Others may choose Unix, AIX, or some other distributed platform service, but that is not important for the discussion at this time. With SAS Institute's continued support for Multi-Vendor Architecture™, the information system developed on one platform can easily be moved to another.

With release 6 of the SAS system, it became easier to create distributed processing across different platforms and still maintain the same basic look and feel. Moving an application from the mainframe to a workstation could be accomplished in a very short time, so long as the planning was there from the beginning. This planning would include operating system checks, library definitions and utilizing host system strengths. In the case of our information system moving the code was not in the planning.

First Steps Towards A Distributed Application

The AF application was ported first along with a subset of the databases. After all of the host dependent code was removed or altered, then testing was done to insure that the application was doing the same task on both the mainframe and the Windows platform.

After this task was completed, remote library services was announced in beta for release 6.08 TS 405. Before the announcement of remote library services, we had been attempting to use EHLLAPI as our method of data transport and job submission. After the announced release, we began to take a more in depth look at TCP/IP and WINSOCK as the chosen method of data access and remote job submission. After deciding to go with SAS/CONNECT and remote library services, we had to work on getting the TCP/IP products working on both the mainframe and the Windows platform.

SAS Products Needed

In order to run a client/server application in interactive mode rather than batch mode a release of SAS 6.08 TS 405 or later is needed. Remote library services was not available on the CMS or Windows platforms until this TS level release. TS 407 was the production release of the remote library services for SAS/CONNECT and SAS/SHARE.

The code necessary for making the connection using TCP as the communications method is the same whether remote library services is chosen or not.

```sas
%let mynode=ip-addr-of-host;
filename rlink 'script file path';
options comamid=TCP remote=mynode;
signon;
```

The code changes in the assignment of the SAS data libraries on the remote host. You need to specify the RLS engine to inform SAS that you are pointing at the remote library rather than trying to assign a local one.

```sas
libname libref SASE9 server=mynode;
```

Connection Method

With SAS/CONNECT there are several connection methods supported for remote library services; TCP/IP, APPC, or NETBIOS. TCP/IP was chosen because it was currently available and support existed in-house to help with any problems encountered.

When using the TCP/IP method of data transport and access you need to have some form of a WINSOCK available to Windows. Penn State uses FTP software's TCP/IP suite of applications. Other supported TCP/IP vendors are listed in Tech Report P-224 dealing with the changes made to release 6.08 of the SAS system. This list does not cover all of the available TCP/IP packages. SAS also has its own WINSOCK which is now being distributed along with the current release of SAS.

On the remote host you will also need to have TCP/IP running as well. For the IBM mainframe running CMS the requirement is IBM"s VM TCP/IP 2.0 or later. If you are using a different remote platform, consult the references listed at the end of this paper for other system requirements and recommendations.

In addition to the TCP/IP software running on both platforms, you will also need to have a linemode telnet port available. The normal default for CMS is to have a port for 3270 telnet at 23 and a port for linemode at some other location. Some sites also run with the same port being allocated for both linemode and full-screen. The Center chose to run three different machines on the same port to provide access to linemode telnet. This created some problems and will be discussed later in this paper. The reason for the three machines at Penn State is because of the volume of users that need to access the CMS system.
Penn State has over 25,000 users who could access the system on a daily basis and the three machines help divide the load. Normally there are approximately 1,000 users logged on a daily basis during the semester.

Problems encountered

While testing the remote library engine, SASE9, several problems, both large and small were encountered. For the most part, the maintenance that needed to be applied for 6.08 TS405 went smoothly on the mainframe. However, on the Windows platform, we had to remove the SAS/CONNECT directories and some of the Base and other product information to have the install process go properly. The only reason we realized that we had done a wrong install on CMS was because of the fact that our LOADLIBs were not showing the correct dates. The SAS log was also informing us that we did not have remote library services or maintenance properly installed. The error we received was an invalid free stor request at a particular address on CMS. We had to reapply maintenance on CMS for the past several releases of TS to get to the level of support that we needed to run remote library services. Remote library services was really only in 'test mode' under TS 405 and was then given production status under TS 407 and 6.10 of the CMS and Windows releases due out very soon.

Also, as mentioned in the previous section, we had three different machines running linemode telnet sessions. The problem we had was that we would go in as one port and exit out the other. This meant that the data streams were not 'talking to each other'. To solve this problem we copied the system file TCPIP DATA to a user disk and made the SAS program on the local host, always go to the same port. This meant that if the systems people ever changed the IP address, we would have to recode them into our AUTOEXEC or SAS program which set up the IP address of the remote server. For sites with only one TCPIP server on the mainframe, this step will not necessary.

Reasons for using Remote Library Services

With the current shift here on our academic computing platform to phase out the CMS operating system, this was our first shot at accessing data remotely. The academic system only has SAS on the CMS and Windows platforms and thus, we could not easily port to a Unix platform at this time. Windows was also chosen as the remote platform because of the unsure direction of OS/2 on the IBM workstation platform.

Another reason for using the Windows platform for remote library services was because of previous usage of SAS 6.04 and the Micro to Host Link remote submission of SAS jobs for reporting on the DOS platform. Other users at our campus had successfully used the Micro to Host Link, so the support was there as well.

Remote library services then presented a way to allow the data to reside on a remote platform and the processing to take place on the workstation. In the past, one had to remote submit a job to analyze the data or FTP the data from the remote machine to the local machine for processing and reporting on the data.

Should you use Remote Library Services?

If you have large data sets and need to manipulate the data on a regular basis, you may want to consider moving the data to the local machine on a daily basis and then running a batch update of the data overnight. Remote library services is not a good choice for users who need to have concurrent access to the same data many times throughout the day. There is still some overhead involved in receiving the data from the remote host and then editing or browsing the data on the local platform. When choosing to use remote library services, the number of users needing access to the data, the number of observations being retrieved, how often the data is accessed and updated, and the hosts involved, each play an important part in the decision making process. If there is a large number of observations that need to be retrieved using remote library services or if many users need to access and update the data at the same time, you may want to investigate some form of a batch merge overnight process. Data integrity is always a concern when one switches from a more interactive access method to a batch oriented process, so you need to look at the whole picture before reaching a decision.

What about data and access views

Another concern is the ability to access SAS/ACCESS views from remote library services. Currently and in the next release of SAS (6.10) there are no plans to add support for the editing or updating of SAS/ACCESS views from the local host. In order to update a view, the programmer must remote submit an SQL update request using PROC SQL or use some form of a data step to apply the updates to the existing remote SAS data view. This problem is also encountered when processing any of the new VIEWs that can be created with release 6.08 and above of the SAS system.
Is there another way to do TCP/IP

A more efficient way for using remote library services and more importantly TCP/IP connection methods would be to allow for direct socket to socket connections. As noted in the section on TCP/IP, one needs to have a linemode session available on the host. If SAS would allow RXSOCKET to be used with WINSOCK and simply allow the programmer to inform the SAS program where the socket is located to connect to, then the linemode telnet port would not be necessary. One could simply invoke SAS/CONNECT on the remote host and connect to the socket. There are no current plans to implement this and this suggestion will only impact users who run on 3270 display devices, such as MVS or CMS connected terminal sessions.

Using Visual Basic to Create the Connection

After attempting to connect with SAS/CONNECT® and TCP/IP for nearly a year, it was discovered that the port number and information necessary to initiate the SAS® connection from the Windows platform to the CMS host was displayed and cold be captured. The problem was getting the port number from the host to the local workstation.

The attempt was made to use an REXEC client on the Windows platform and then capture the SAS® output back to a log file on the PC and use the log file as input to a SAS® data step to read the port number. The problem was that after SAS® initialized on the mainframe it did not display the port number on the console. Next CMS Pipelines was used to try and capture the information. This was also unsuccessful. Finally with encouragement from another user in the University community Visual Basic with HI.LAPI was chosen as the method to initiate the logon to CMS and then the capture of tel port number and the subsequent passing of that information to the Window program.

The VB program took about a week to develop and another week or two to test. The program prompted the user for their login userid and password and initiated a logon to the CMS host. After logging on the the mainframe, SAS® was started with the following command:

SAS (DMR COMAMID=TCP

This then produced the port number being printed to the CMS console. The Visual Basic program then captured the port number from the console and stored it in a variable. wsa then written to a SAS® program that was to be used to start SAS® on the PC platform. The signon program for SAS® for Windows contained a macro variable to define the host system name and the OPTIONS COMAMID=TCP and server=mynode (macro variable). SAS® was then started by the Visual basic Application and control was turned over to SAS® and Windows. At this point the connection between CMS and Windows with SAS/CONNECT® using TCP/IP had been established and we could run remote submits to the host or use the Remote Library Services to access any necessary SAS® datasets.

The main reason that this method was chosen was because of the line-mode telnet port not being fully supported or encouraged at our site and to automate the process so that any user could initiate a SAS® session and have the same parameters. The program could be tailored for other hosts including CMS, MVS or possibly Unix.

In the future the application will have a profile which will allow the user to provide a userid and password (if desired), as well as the paths and file information for any SAS® autoexec or script files. Currently the VB program searches the path for the sas.exe files and requires the user to specify the script (.scr) file before starting the login sequence. The user also has the option of providing an autoexec file which is appended to the signon.sas program created by the VB application.

It is hoped that SAS® may support something similar to this so that the SAS® applications developer does not have the need to learn yet another programming language. This project was a learning experience on the author's part and the hope is that this will help some of the other applications developers out there that have the need to access their data from an IBM mainframe running CMS or MVS and not have to worry about a linemode telnet port being available.

References

SAS/CONNECT Software, Usage and Reference, First Edition
Technical Report P-224, Technical Report P-255,
SAS Communications 'New Distributed Computing Capability'

1397
Acknowledgements

*SAS, SAS/CONNECT, SAS/SHARE, and SAS/ACCESS are all registered trademarks of SAS Institute, Cary, NC.*

*DOS and Windows are trademarks of Microsoft Corporation*

*Visual Basic is a trademark of Microsoft Corporation*

The author wishes to thank Mimi Ridenour for help in the editing of this paper.