

USING THE SAS® SYSTEM FOR REMOTE DATA ACCESS AND DISTRIBUTED COMPUTING

**John McIntyre
SAS Institute Inc.**

Introduction

This paper will discuss SAS Institute's view of distributed processing and review the use of the SAS System tools, SAS/CONNECT™ software and SAS/ACCESS® software in distributed applications.

In particular, I want to cover some of the ways that client/server applications can be characterized, and how use of the SAS System tools can provide customer applications with outstanding value.

What is client/server computing?

In the many trade journals that we all read, there are numerous discussions about client/server computing. At SAS Institute, we view client/server computing, in its simplest form as one process or task making a request for services from another process or task. In a SAS System environment, this could be accomplished by a SAS application making a request to a DBMS, like DB2, and receiving data from the database server. Client/server computing, therefore, is basically a style of computing which calls for two processes to interact towards the filling of a request.

In most cases, though, the popular definition of client/server includes other nuances, such as the presence of workstations as clients, the use of SQL as a query tool to data and the access to data being implicit. This last item is important, because it means that the method for data transfer from the server to the requesting client does not result in a copy of the result set on the client disk. In addition, it is often the case that client/server suggests a transaction-oriented data access, moving small amounts of data.

Data services only?

Another common perception of client/server applications is that they are usually clients making requests for data services from the server. That is, most consumers view

client/server applications as the way for a workstation to offload some processing from the server, often a corporate mainframe, without having to duplicate the corporate data on each client machine.

However, there is no reason to limit the kinds of services that a client/server application can provide to data access. Clients often need the use of compute services on remote platforms. This may be because of specialized facilities like vector processors. Another reason for remote compute services is to minimize the cost of the transfer of large amounts of data.

Often, client/server is contrasted with peer-to-peer computing. I will not examine peer-to-peer definitions in this paper. Suffice it to say that peer-to-peer computing accomplishes all of the same tasks of client/server transactions, without the inherent hierarchical arrangement of clients and servers.

How can applications be converted to client/server computing?

Applications of all kinds have three basic segments. All applications need to access and format data. They need to apply some logic to the data in an effort to translate it into useful information. All applications must offer these results to the user in a friendly interface, whether that be an interactive screen display or simply hard copy graphs and tables.

The most often stated motivation for exploring client/server technology is to realize savings by using the processor cycles of workstations to offload processing cycles on corporate mainframes. This configuration has a lot of attraction to information professionals, working under constrained budgets. Successfully implemented, it should allow large processors to be used for the tasks for which they are best suited, such as management of corporate data and network communications, and allow those environments with the most pleasing human interface to handle the actual interactions with end-users.

Downsizing or rightsizing?

The decision process around client/server applications often goes in large swings. Many consumers interested in making use of new workstation technology aspire at first to move applications and their supporting data files on to the workstation or network. In most cases, this is not the preferred solution — for many reasons — not the least of which is the sheer volume of data. Applications written to run on large mainframes do not confront the problems of storage and management of corporate data. As a consequence, the applications access the necessary data in its raw form, reducing it as part of the work of the application. When applications like this are planned for movement to workstation platforms, the issue of the volume of data becomes relevant. Thus, efforts to 'downsize' applications by simply transferring them turn into considerations of 'rightsizing'.

In a 'rightsizing' frame of mind, application developers begin to consider the three segments of the application for movement. The first candidate is the presentation layer. This is almost a given, since part of the reason to consider this move is to take advantage of the graphical user interface available on workstations. The next segment is the application logic segment. If it can be moved to the workstation, then a savings can begin to be realized, since processing which would have been going on in the mainframe will now be accomplished on the workstation.

Here, the MultiVendor Architecture™ (MVA) design of the SAS System is of particular value. Programs written for the mainframe can be moved to workstation platforms with little or no modification. Having the programs executing on the workstation would allow the data to remain on the mainframe, and make use of the 'cheap' workstation cycles. Of course, as mentioned before, those programs may be written expecting the full data file to be available. If that is the case, then an application programmer is faced with the 'myth of client/server' processing — that separation of data and processing means that data will not have to move across the network.

Distributing the processing

To accomplish this division of labor, the software is going to have to allow for client/server data access as well as offer client/server access to computing services. In our example, the application programmer with access to both types of server 'services' can choose to divide the application logic along the lines of the data access and reduction portion and the final analytic and report writing portions.

If we can assume that the application can be logically, as

well as physically, segmented, then a number of scenarios can be explored. In the beginning stage, the application segments all reside on the central processor. Data are accessed in their raw form, logic applied and results generated, all on a single platform (see Figure 1).

Alternative #1 - Remote data access

The first alternative would be to leave the data in the central processor, and move the application logic and presentation to the workstation (see Figure 2). This moves some of the chargeable processing off the central processor and leaves the data where it is stored.

However, it does not address the challenge of voluminous data. A client/server application which stores data remotely still must move that data to the processor where the application will execute. This fact is often obscured because too often the sample cases for client/server applications are more of a transaction orientation than decision support.

One way to address this problem is to effect some amount of data reduction in the central processor before it is moved to the workstation. View technology offers some help here. Using SAS/ACCESS software views to data, applications can pass data reduction instructions to database systems at the time the data are being requested. So, end-user applications at the workstation receive data which has been 'rolled-up' or subset inside the database system. The workstation application then works on only the level of detail necessary without having transferred too much data.

Of course, this strategy still comes with a cost. View technology works as a window into live data. That means that the application will be accessing the production data each time it is invoked. If the view has been constructed to provide data reduction, it is using valuable database system processing cycles, possibly impacting transaction applications running against the same data.

Alternative #2 - Split logic

A second alternative splits some of the logic of the application across the two platforms. Using SAS System tools, this application could still use SAS/ACCESS software to materialize data from the data base system, but make use of some of the data reduction procedures within the SAS System (see Figure 3). Use of PROC MEANS, PROC SUMMARY or PROC FREQ could easily allow for the aggregation of detailed data into meaningful subtotals for analysis and presentation. The output from these procedures could then be passed to the workstation for

further logic processing and preparation of the presentation format.

The advantage of this strategy over number 1 is that the impact on the production database can be reduced by having SAS application logic handle the data reduction tasks outside of the database processor.

It still, however, requires access to the production data at the time the application executes, and has the additional cost of returning some of the logic processing cost to the central processor. This cost may still be acceptable when compared to the costs in time and productivity of transferring large files across busy network systems.

Alternative #3 - Information database

A third strategy adds an information database to the algorithm. An information database would be the repository of data, copied from the live production data system, with appropriate data formatting logic applied. All of the SAS System tools referenced above continue to be used, with SAS/ACCESS software providing a path to the production data, and SAS procedures applied to accomplish data reduction. But, instead of immediate transfer to the workstation applications, a database of SAS data files exists as the source for access by workstation applications.

The advantages to this strategy are many. The most obvious is that production database systems are insulated from daily access by decision support applications. The process of building this information database can be accomplished in non-production hours.

The SAS System offers enhancements which better qualify its use as an information database. In Version 6 releases, capabilities exist for adding indexes, compressing the size of data files, and password protection of files. In addition, with the return of SAS/SHARE® software to the mainframe platforms, concurrent update access to data files in the information database is available.

Into the future

The SAS System continues to develop its capabilities to provide services for the development of distributed applications. Using currently available tools, data can be accessed from a wide variety of proprietary sources. In Release 6.07, that list has expanded to include IMS, CA-DATACOM/DB, and ADABAS for MVS, and INGRES for VMS. In addition, the ability to pass native SQL requests to relational DBMSs is being added, further improving the efficiency of view technology. Looking

ahead, SAS Institute is working on interfaces to popular DBMS products on our supported UNIX platforms. Interfaces are also under development for SQL Server in OS/2 and PC file formats like Lotus WK formats.

In the client/server data area, the connectivity capabilities will be enhanced to allow for access to remote data sources using LIBNAME references. This will make the access to these sources appear more transparent, and will allow applications to access remote data without making a local disk copy.

Finally, in the client/server access to compute services, SAS/CONNECT software will be enhanced to allow for asynchronous processing of remote compute tasks. This 'background' processing will allow for local sessions to route requests for compute services to remote machines, and continue doing local processing.

Summary

Each of these strategies makes use of both platforms to its best use. The issue of user presentation is clearly a workstation issue. The division of the labor for application logic and data access will be made, taking into account not only the physical location of data and the availability of hardware resources, but other business goals. With the ability to use each platform as both a data server and a compute server, SAS users are not locked into strategy for the design of their distributed applications.



SAS Institute Inc.
Sales & Marketing Division
SAS Campus Drive
Cary, NC 27513
Phone 919-677-8000
Fax 919-677-8123

SAS, SAS/ACCESS, and SAS/SHARE are registered trademarks, and SAS/CONNECT and MultiVendor Architecture trademarks, of SAS Institute Inc., Cary, NC, USA.
Other products mentioned are trademarks or registered trademarks of their respective companies.
Copyright © 1992 by SAS Institute Inc.

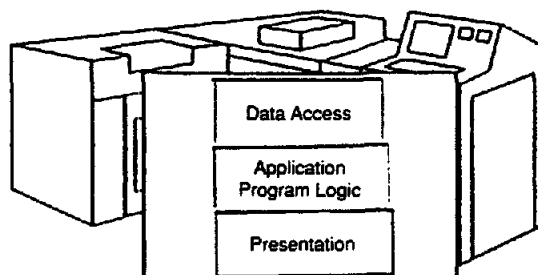


Figure 1

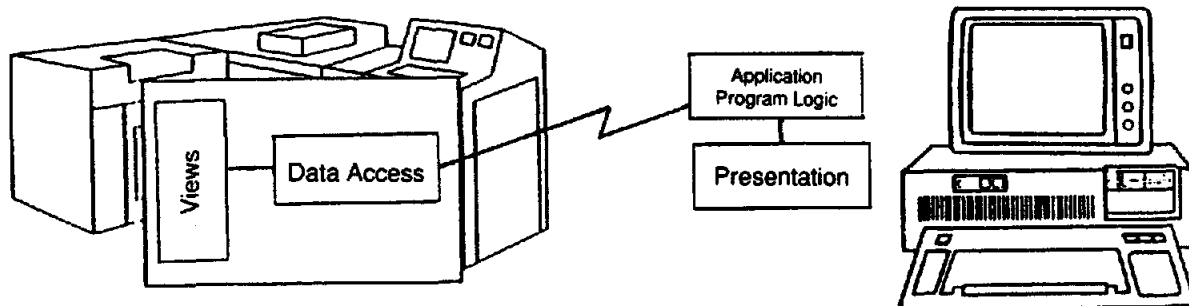


Figure 2

