Paper 16

Distributed Applications with SAS/AF® Software on Client and Server

Pat Herbert, SAS Institute Inc., Cary, NC

ABSTRACT

With the advent of the Internet, the demand for distributed computing is on the increase. With the Nashville Vision, the user can use SAS software to create distributed SAS/AF objects to exploit the data and compute power of the SAS System. In this paper, we will discuss the use of SAS/AF applications on the client accessing remote models on the SAS server. Users familiar with SAS/AF software can write distributed applications without getting a higher degree.

INTRODUCTION

Distributed applications have become more mainstream. In particular, invoking methods on remote objects has become very popular because of technologies like COM/DCOM and Java. These distributed applications allow state-of-the-art user interfaces created with any of several application development tools to be bound to legacy systems with minimal effort. They also allow thin and/or smart clients to use large server based databases and to use the computing power of large servers. It is now possible to create SAS/AF objects using the SAS Component Language (SCL) that can be accessed remotely by various clients using numerous transport mechanisms. Those objects can be used locally within a SAS session or they can be used remotely from client applications that could be written in languages like C++, VB, Java and SCL. Likewise, SAS/AF applications can be built to create instances of and invoke methods on local objects or remote objects. These remote objects can be accessed via COM/DCOM or a special protocol implemented on top of TCP/IP. First, we will explore how to create SAS/AF objects that can be accessed from client applications. Then we will explore how to create SAS/AF applications that can create and invoke methods on remote objects. Finally, we will look at integration of the client application with the server objects.

DEFINITIONS

IOM (Interface Object Model) - IOM is the SAS internal mechanism upon which all inbound and outbound remote object access is built. It provides support for standard remote object protocols like COM/DCOM as well as the bridge protocol.

Bridge Protocol - The bridge protocol is a private TCP/IP based protocol that allows communications between disparate systems. This protocol is typically used when the standard remote object protocols cannot be used. An example of that might be when the client is implemented on a PC platform and the server component on a mainframe.

BRIEF HISTORY

The SAS system as been accessible as an Automation Server since version 6. This allowed users to script SAS from external applications but it did not provide direct access to user written objects so they were limited to the interfaces provided. There was also a mechanism in place to allow SAS/AF applications to access COM-based automation servers but the interfaces that were supported were limited with respect to supported data types and parameter use. Also, version 6 allowed SAS-to-SAS remote object use but the client and server both had to be written with SCL - there was no open access. We have now combined and extended many of these capabilities and provided a consistent set of interfaces so that SAS/AF objects can be created and used without concern for what kind of client will be using it. Likewise, SAS/AF applications can be written to access remote objects without concerns about the remote implementation language.

SAS/AF OBJECTS ON THE SERVER

In order for remote objects to be accessed, the client must know about the object, its location and its calling conventions. This set of information, typically referred to as the class metadata, consists of information needed by the client and the server to support access. Some of the items in the class metadata include the UUID that uniquely identifies the class, the list of methods defined by the class and the parameter ordering and types for each method. Some if this metadata is available as a function of defining SAS/AF classes but some of it has to be added in order to access the classes remotely. Beginning with version 7 of the SAS system, extensions were made to allow additional metadata to be added to SAS/AF classes. These enhancements allow the parameters, their ordering and their types to be defined with the class. There is some additional class metadata that is needed for remote objects so that client code can access them.

SAS/AF APPLICATIONS ON THE CLIENT

In order for SAS/AF applications to access remote objects, there must exist a SAS/AF proxy object that corresponds to the remote object. These proxy objects contain class metadata needed in order to marshal the parameters from the SAS/AF application to the remote object implementation. This class metadata is identical in form to that needed to define SAS/AF objects on the server. Once a proxy class is defined, it can be used to create and instance of and invoke methods on remote objects, including remote SAS/AF objects.

CONCLUSION

Because this technology is still being developed, the complete paper will appear at http://www.sas.com/rnd/appdev/webAF/doc/Papers/DistributedApps.htm before presentation at SUGI 24.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Pat Herbert
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Work Phone: 919-677-8000
Fax: 919-677-4444
Email: saspmh@wnt.sas.com