Current and Future Developments for SAS® Software under MVS
Brian P. Bowman, SAS Institute Inc., Cary, NC

ABSTRACT
This paper discusses MVS specific developments in Release 6.08 of the SAS System and current research activities. The primary focus is on connectivity, interoperability, and the integration of SAS Software with other environments.

INTRODUCTION
With the introduction of Version 6, SAS Institute delivered the first installment of MultiVendor Architecture™ or MVA™. The MVA Host layer is what defines “SAS Software under MVS” by providing the interface to MVS facilities that support the SAS environment. The Host layer is subdivided into functional subsystems that provide specific services such as memory management, external file I/O, interprocess communication, machine code generation, and so forth. The organization of the Host layer provides the foundation for integrating MVS specific capabilities into SAS Software.

In Release 6.08, features implemented in the Host layer enable access to a variety of computing environments. SAS/SASSESSION software provides an interactive SAS environment for CIGS terminal users. New host functionality dynamically translates IEEE and VAX data types to and from corresponding IBM® 370 formats. This capability is used by inter-host communication facilities currently under development. An interface for developing ISPF dialog management applications is now part of Base SAS software.

Future developments in SAS software for MVS are supported by ongoing research activities. Increased client-server and distributed computing capabilities are part of the MVS operating system’s current evolution. Institute developers are researching ways to exploit these capabilities in SAS Software. The integration of MVS facilities with networked workstations provides new opportunities for interoperability in the SAS System.

SAS/SASSESSION SOFTWARE
For a number of years, SAS Institute has received requests to support SAS software running from CIGS. CIGS applications utilize services provided by the CICS environment and access resources under its control. For a transaction processing application with specific functionality, this methodology works well. But an interactive SAS session provides a general purpose computing environment where resource usage is unpredictable. Even a modest SAS application can consume resources to upset an otherwise well behaving CICS system. Substantial modifications to the Host layer would also be required to interface SAS software directly to CICS services.

Because CICS is transaction based, its resources and services are most often utilized in relatively short time intervals corresponding to transaction length. In contrast, the computing activities in a typical interactive SAS session can obtain and hold resources for long periods of time. A principal factor in implementing the SAS Host layer is the degree to which the underlying platform supports multitasking, large memory allocations, asynchronous I/O, dynamic file allocation, and a variety of other operating system functions. As a transaction processing system, CICS is not designed to provide this level of functionality.

After understanding the limitations imposed by the CICS environment, SAS/SASSESSION software was developed in an APPC/MVS framework. APPC/MVS provides the inter-host communication and a SAS/ACCESS® engine provides access to the DB2 data. This method allows the EIS application on OS/2 to select and transfer only the DB2 data that it requires. But how can OS/2 process DB2 data in IBM 370 format? This is where the host implementation of dynamic data translation comes in. The following diagram illustrates:

![Figure 1](image-url)

DYNAMIC DATA TRANSLATION
A forthcoming cooperative processing feature will provide transparent access between SAS sessions running on heterogeneous platforms. For example, a SAS/EIS® application running on OS/2® can have record level access to data stored in a DB2® database on MVS. APPC provides the inter-host communication and a SAS/ACCESS® engine provides access to the DB2 data. This method allows the EIS application on OS/2 to select and transfer only the DB2 data that it requires. But how can OS/2 process DB2 data in IBM 370 format? This is where the host implementation of dynamic data translation comes in. The following diagram illustrates:

![Figure 2](image-url)

Access to data through SAS engines is defined by the SAS data set model. The model currently recognizes character and floating
point numeric data types. IBM mainframes use the EBCDIC character set while all other SAS platforms use ASCII. Translating character data between the two is a simple process using a byte mapping table and a single IBM 370 machine instruction. Depending on the platform, floating point numeric data are represented in IBM 370, VAX, or IEEE format (IEEE is used by the PC, OS/2, and UNIX platforms). Complex bit manipulation algorithms are required to transform floating point numerics between representations.

The problem of translating heterogeneous data types is further complicated by the common usage of SAS variables. In the SAS data model, the VAR statement (or other features) may be used to select a subset of variables from a data set. For a given access the number and type of variables is unknown until application (that is, DATA step, PROC, SCL, SQL) programming statements are parsed. IBM 370 floating point numerics are internally represented in 8-byte double precision format. The LENGTH statement can be used to specify a stored length for numeric variables when fewer than 8 bytes of precision are needed. This reduces data set size but complicates transformation.

Performance was a primary consideration in the design of dynamic data translation. All selected variables must be translated for each record accessed through the remote engine. In the previous example, the DB2 Engine can perform remote WHERE processing on behalf of SAS/EIS software. This may reduce the number of records passed to OS/2 but it cannot change the number of variables translated for each record. Only the VAR statement and corresponding capabilities in SCL and SQL can reduce the number of translated variables. Data translation must be optimized because remote access response time is critical.

To achieve flexibility and performance, dynamic data translation is implemented in the Host layer's code generation subsystem. On the host, machine instruction sequences named code streams are generated for the DATA step and other SAS language environments. A code stream tailored to the type, length, and number of selected variables is constructed dynamically when remote data set access is initialized. Remote I/O operations invoke the code stream to perform translation in the direction of data transfer. For example, a READ request from OS/2 translates the data from EBCDIC-IBM 370 representation to ASCII-IEEE. A WRITE request translates in the opposite direction. The code stream is removed when remote data set access is terminated.

ISPF APPLICATION INTERFACE

In Release 6.08, an interface to IBM's Interactive System Productivity Facility (ISPF) is now included in the base SAS System. This replaces the SAS/DMF® (SAS/Dialog Manager Interface) product available in Version 5 of the SAS System. ISPF provides dialog management services for developing applications (dialogs) in an interactive (or batch TMP) TSO environment. The SAS interface enables the DATA step to be used as a dialog development language. The document “Developing ISPF Applications with the SAS System, Release 6.08” provides details on the SAS interface and a list of IBM documentation pertaining to ISPF. Here is a brief summary of dialog development using the SAS interface.

ISPF applications are created by combining dialog elements. Some of the elements include panels, dialog variables, dialog functions, and messages. Panels represent display screen layout and define fields associated with individual dialog variables. Dialog variables are mapped to DATA step variables using the ISPF VDEFINE service. The DATA step (dialog function) can process data entered on the panel, perform computations and query data sets, update dialog variables with the results, and redisplay the panel.

Processing conditions may be displayed on the panel using ISPF messages.

All dialog management services are available in the SAS interface. Services are invoked using DATA step CALL statements in the following format:

```
CALL ISPLINK('servicename', additional-parameters)
```

or

```
CALL ISPEXEC('servicename additional-parameters')
```

ISPLINK and ISPEXEC correspond to the two forms of dialog management service invocation defined by IBM. The servicename and additional-parameters must follow the exact conventions defined in ISPF documentation. The return code from the service is provided in ISP_RC, a SAS System defined numeric variable. A special servicename of 'SAS' is available to change the status of some of the SAS options that relate to the ISPF interface.

A new command, HOSTEDIT, allows ISPF/PDF EDIT or BROWSE to access the contents of a SAS Display Manager window. When HOSTEDIT is entered from a window command line, an ISPF/PDF EDIT or BROWSE session appears with the contents of the window. A Display Manager window that may be modified (for example, PROGRAM or NOTEPAD) will invoke EDIT. Upon exit, the contents of the EDIT session are placed in a Display Manager window. A Display Manager window that is read-only will invoke BROWSE.

FUTURE ISSUES

From its inception, the Version 6 SAS System has provided connectivity between workstations and mainframe systems. Source statements can be submitted from a workstation to a SAS session on MVS. When processing completes the results are delivered to the workstation. This method requires a SAS session on MVS to perform computing tasks on behalf of the workstation. Bulk file transfer between different platforms is also available. These features provide inter-host connectivity but require users to know where data resides and which platform will process it.

Continuing research on interoperability suggests that the SAS System can provide a more integrated view of networked heterogeneous platforms. Client-server computing provides a model for distributed processing where cooperating SAS software components may reside on different platforms according to function. A software component is matched with the platform that best fits its resource utilization.

New MVS interfaces provide opportunities for enhancing connectivity and interoperability in the SAS System. With TCP/IP, NFS, and AF5 available on MVS, workstations can access mainframe data and computing resources through common network facilities. OpenEdition™ MVS (POSIX) brings UNIX workstation capabilities to the mainframe. Research on how SAS software might exploit these capabilities is currently underway.

CONCLUSION

Release 6.08 provides expanded access for SAS Software under MVS. Connectivity is enhanced through the APPC/MVS implementation of SAS/SESSION software. The host dynamic data translation facility is an important component of interoperability in the SAS system. A new interface in the base SAS System enables application development in the ISPF dialog manager environment.
Ongoing research ensures that SAS Software for MVS is positioned to exploit future MVS operating system capabilities and interoperability with other platforms.

ACKNOWLEDGMENTS

The following MVS Host Group members contributed to this paper:

Daniel Berry  Brian Bowman
Gary Burchett  Keefe Hayes
Dale Ingold  Lisa Ripperton

SAS, SAS/ACCESS, SAS/DMI, SAS/EIS, SAS/SESSION, MultiVendor Architecture, and MVA are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. DB2, IBM, OS/2, and OpenEdition are registered trademarks or trademarks of International Business Machines Corporation. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.