Multivendor Architecture -- Supporting Feature Rich Platforms with a Uniformly Architected System

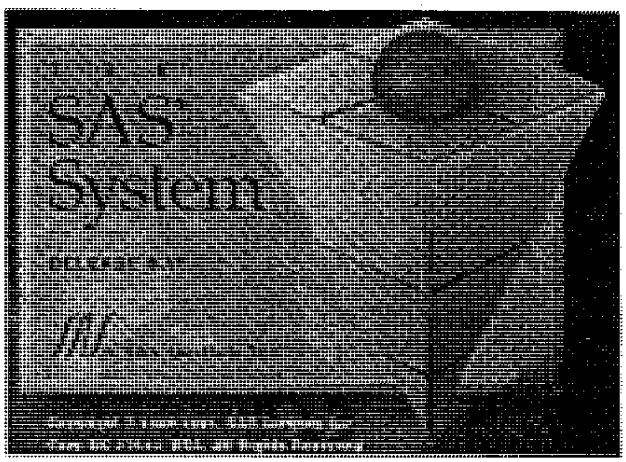
Mark W. Cates, SAS Institute Inc., Cary, NC

ABSTRACT

Multivendor Architecture (MVA™) is the foundation of the cross-platform portability and interoperability of the SAS® System, Version 6. This architecture provides customers with hardware independence and a flexible implementation. In 1993 alone, the Institute's MVA strategy delivered the SAS System on an unprecedented thirteen operating systems. However, with continued divergent development of operating system software in areas such as the graphical user interface, graphic and print driver managers, hypertext viewers, editors, and other services that are integral to the operating system, users are faced with a dilemma: They can develop well-integrated applications on individual operating systems, or concentrate on applications that are less well integrated into specific platforms, but port easily and uniformly to many operating environments. This paper discusses proposed enhancements to the Institute's Multivendor Architecture that will help bridge the gap between a well-integrated platform application and a highly portable application. The audience is invited to participate in a panel discussion of customer direction as related to this topic with Institute Host Division Managers immediately following this paper.

INTRODUCTION - What is MVA today ?

To understand what MVA is today, we must look back at the trends in the computer industry during it's inception in early 1985. In the computer hardware industry, a shake out of major players was beginning. Traditional mini computer markets began to shrink as economically priced Unix RISC based workstations and servers became increasingly more powerful. IBM PCs, PC clones, and Apple Macintosh computers began to quickly proliferate across desktops. Intel and Motorola began demonstrating processor performance increases at a clip of 200% per year. Mainframe computing was still near it's all time peak. These conditions in the hardware industry generated a sense of uncertainty for managers responsible for outlining their organization's computing architecture. In the software industry, a transition of languages used in the professional computing arena had begun. The C programming language was quickly



replacing Pascal and Assembler as the choice for professional applications development, and this shift in programming style was spurred on by the increasing popularity of Unix. This was a time of change. MIS decision makers and end users needed x-ray vision and a crystal ball to chart the course for their operations for the rest of the decade. What was certain, was that no longer would a company's information technology be provided by a single hardware vendor. Customers were looking for software solutions and tools that insulated them from hardware platforms.

This background provided the basis for MVA. At that time, the SAS System was just being finished converting from the PL/I programming language to the C programming language. The SAS System was architected to be portable across all hardware and operating systems platforms supported by the Institute at the present time as well as taking into consideration platforms that were on the horizon. The Institute's development teams went about researching and designing the building of a large application program that could be portable across all hardware and operating system platforms. Much research was spawned on the design of virtual operating systems. The dominant goal of Version 6 was portability across all platforms. Emphasis was placed on portability of software, feature set, data sets and catalogs. Subsystems for developing a portable

application program were designed with the intent that these subsystems would be developed optimally on each platform. Below are listed the first subsystems creating the foundation of MVA:

Subsystems designed for platform dependent optimizations

- Memory and Task Management
- File I/O, data set, catalog, and external files
- Dynamic Executable Loading
- Display Manager primitives
- Native machine code generator
- Communications Management
- Miscellaneous services

Another set of subsystems were designed portably with no intent that the subsystem would be replaced or significantly enhanced on the target platform. These subsystems were considered to execute and perform identically as possible across all SAS platforms. These subsystems in effect, addressed end user issues where there were no standards.

Subsystems designed for platform independence with little or no platform optimizations

- Help System Display and Authoring
- Graphics Device Drivers, Printers and Plotters
- Print Management using Forms
- Portable Full Screen Display Manager
- Message Text Manager

A major decision that in retrospect has proven to be very strategic and important, was made that the entire SAS System would be ported and available on all SAS supported platforms. At that time, this was a courageous decision. Critics claimed that an application developed on and for the mainframe market could not be ported and run on a PC with only 640k of memory. However, this was accomplished as a result of the MVA design.

The resulting SAS System Version 6, was a product that performed well, and was in fact portable across all supported platforms. SAS applications were portable across platforms and data could be moved across platforms. Hardware independence through portable software was achieved. Portability can be measured in many factors. The SAS System was and still is portable in all the following ways:

- Portable C source code
- Data set and catalogs are portable
- Procedure feature set and syntax
- Data Step feature set and syntax

Constantly at odds, however, was the competing goals of the SAS portable feature set compatibility and the uniqueness and advantages that each platform provided. Although MVA provided for the "hooks" to take advantage of platform specific features, the rate of platform evolution was astounding in the early 1990s. The workstation market including Unix, PCs, and MACs grew stronger each year. These platforms sought to create usability standards where none had either existed or been accepted before. It became clear at the Institute that the MVA design needed to evolve to support platform optimized subsystems that had previously been considered platform portable.

Evolution of MVA

Unix played an influential role in the design of the SAS System. Much of the layering of subsystems in SAS was derived from the Unix operating system model. In fact, the development platform at SAS Institute for Version 6 has been Unix workstations. Once the SAS System began to port to a variety of Unix versions, the MVA model was expanded to support software engineering techniques for portable build files. Each Unix operating system differed slightly requiring minor adjustments in the build processes. For a system of seven million lines of source code, make files had to be designed that would blueprint how to build and compile the system. The tools evolving on the Unix platforms during this time strongly influenced the design of the Institute's source management.

Meanwhile the Unix GUI market was being shaped by the development of System X window manager, commonly referred to as X-Windows or X. X spurred on the rapid growth of software at the Institute using window management techniques. Unfortunately X did not address the user interface look-and-feel issues, so the design of a consistent, well presented user interface did not mature until a few years later when the PC window operating systems became popular. In 1990 Microsoft introduced the third generation of Microsoft® Windows™ and IBM introduced the long awaited OS/2® 2.0. These two platforms, over a period of a few years, began to create software standards where none had existed or been accepted before.

The phenomenon of Microsoft Windows is the creation of widely accepted graphical user interface (GUI). The mouse quickly became a standard pointing device that is present today on nearly every workstation. Text and graphics were quickly merged within a single window. The GUI also began to standardize other application areas such as Help Managers, Device Management, Printer and Plotter Drivers, as well as Email and Fax support. Microsoft also began to develop extensions to Windows computing called Windows Open Systems Architecture (WOSA) services. These services provided application developers and end users with standards for areas such as:

- Object based programming
  ⇒ OLE COM
- Data base access
  ⇒ Open Data Base Connectivity (ODBC)
- Communications access
  ⇒ TCP/IP Sockets (Winsock)
  ⇒ SNA Server APIs

To respond to these software advances and the requirements from customers to create solutions based on available standards, MVA evolved. Designers and architects at SAS Institute began to extend the original MVA design to include subsystems not previously considered to be enhanced on the target platform. This change was not driven by absolute necessity; a portable subsystem was available on each target platform perfectly capable of performing the given task. The need for MVA enhancements was driven by the requirement that the SAS System adhere to user interface standards for the target platforms. Users also required that the SAS System utilize industry standard interfaces for data base and communication's access. The software world was changing. No longer was application portability the only primary objective; it was now also very important that the application fit well on the intended platform. This "fit" could be defined in many ways, from the viewpoints of many companies. Information Technology departments began to create feature checklists or software standards that applications that were developed and purchased must meet. In the absence of any existing platform standards for user interface, interoperability, and feature sets, end users were left to create their own standards.

Vendors such as Microsoft have begun to address the lack of software standards. Microsoft has proposed the Windows 95™ logo. This logo will represent software that has been developed for Microsoft Windows 95™ and adheres to a comprehensive list of standards. Independent testing of Independent Software Vendor's software must be completed to license the logo. This type of standardization program continues to demand that MVA addresses the nativeness of the target platform.

More recently, SAS Institute has recognized this need for the SAS System to "fit" well on the target platform. While most of the target platform enhancements emanated on the workstation platforms, several enhancements were considered for implementation as a portable subsystem across other platforms. This is again the case of the SAS System providing ease of use software in areas on platforms which do not have a clear industry standard for a given feature. Concerning the user interface, it is now clear that the Microsoft Windows GUI drives the direction of the user interface for the SAS System. With the expanding market for the SAS System on Windows, and the fact that the Windows API is being ported to many platforms, this seems an obvious choice. However, there are additional feature sets that MVA needs to address in order for the SAS System to ensure a good fit, on various target platforms. In the following section, specific feature sets and subsystems (components) will be introduced. These subsystems are candidates for a native implementation on the target platform. For each component, a discussion will be presented concerning the state of development with respect to the MVA evolution of this particular component.

Installation Programs

Installation programs have evolved from simple file copy scripts and diskette decompression algorithms to elaborate GUI based programs. The new class of installation program is responsible for not only file copy but also the initialization and setup of resource files, .INI files, and the system registry. Client site installation programs are popular allowing workstation clients to perform an installation of only the local client pieces of the software to be installed, leaving the majority of the application to load from a network server. Removing (or deleting) software components have become just as vital. It is no longer sufficient to delete the program files associated with a product. Now, it is desirable to uninstall the program by removing entries from the resource files and system registries.

Windows in particular have created a defacto standard for installation programs. The installation program for any application is immediately recognizable by the file

name setup.exe. The functionality of setup.exe has become so familiar that Microsoft is now requiring a minimum set of features in installation programs, in order to qualify for the Windows 95™ logo. The SAS System for Windows will be supporting a setup.exe installation program in a future release.

System configuration systems or tool kits are now available, but not widely used. These software systems allow software and network administrators to inventory workstation's hardware and software. They provide for automatic and unattended installation of software products locally or remotely. One of the most recent entries into this market is Microsoft's System Management Server (SMS). SMS is being researched for consideration of being supported by the SAS System in a future release.

Graphical User Interface

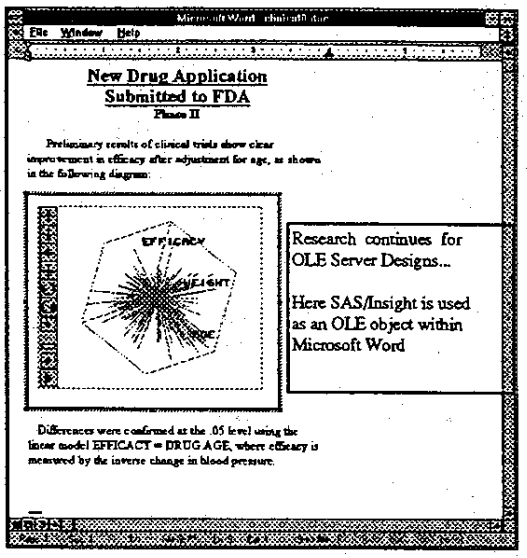
The Graphical User Interface on all platforms is increasingly important each year. Customers demand that the applications they are delivering meet industry standard user interface guidelines. The cost of training combined with the cost of supporting products that do not support industry GUI standards is prohibiting. Some elements of the GUI that must natively portray the look-and-feel include appropriate support for:

- Common dialogs for common operations
- Common GUI controls familiar to all users
- Common actions and expected behavior
- Common look-and-feel across all SAS products
- Appropriate support for mouse driven applications
- User interface responsiveness
- Appropriate use of System Colors and support of the system schema
- Appropriate support of Fonts in display and printing

The SAS System has addressed many of these evolving issues in Version 6. Much of the Institute's focus is dedicated to incremental improvements in the GUI nativeness. Customers can expect further refinements and adherence to industry GUI standards in future releases.

Object Based Industry Standards

Object based programming methodologies offer promise of creating applications that are more powerful, customizable, and integrated. Code can be developed as components available for reuse with less software maintenance and easily customizable for slightly different implementations of the same general solution. In the software industry today, many object based technologies are competing for market share. Unfortunately, not all of these competing technologies are portable across multiple platforms. The end user is left with a choice of pursuing platform specific implementations that integrate well, or choosing less native object implementations that perhaps are more portable.



Within a future release of the SAS System, objects can be classified as internal and external objects. Internal objects exist in the domain of the SAS System. These objects can be shared across SAS products better integrating the existing suite of SAS products. There is also a need for these objects to work with the external environment such as the operating system or other applications. This can be accomplished by supporting objects in a format that is native to the operating system.

Competing object based technologies provide yet another challenge to the evolution of MVA. The goal is to support platform specific objects within the SAS System while also providing some level of portability of the objects between SAS supported platforms. Objects also provide a convenient mechanism to more closely integrate the feature sets of SAS products. Research is currently proceeding at the Institute to meet these goals.

Microsoft's OLE (Object Linking and Embedding) and OpenDoc technologies are being researched as representative of the most significant object technologies

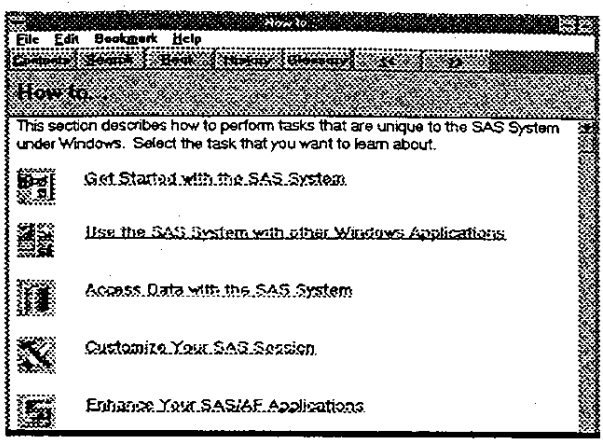
available today. The goal is to interoperate well with other applications that support these object technologies.

Editors

It is often said never to intervene between developers engaged in a discussion about which is the most powerful program editor. Discussing editors is considered as taboo as discussing religion or politics in open circles. But there is a great deal of value in providing the ability for the end user to use the editor of choice with the SAS System. Users have long stated that a consistent SAS program editor provides value for their users to be able to easily move between platforms and still be consistently productive developing SAS code. While others make the point that the choice of editor is platform dependent, and that the SAS program editor is severely limited when compared to alternative editors. There has been some attempts to resolve this dilemma by allowing the end user to use the editor of choice while using the SAS System. On the VAX VMS® system, the SAS program editor can be replaced by the a popular VAX programmable editor. And on Unix, the SAS System provides a command "HostEdit" that spawns the Unix editor of choice and returns the text to the SAS program editor. While on Windows and OS/2, adjustments have been made to allow the SAS program editor to be closed and replaced by the PC editor of choice. SAS code is submitted from the clipboard after a copy operation. Debate continues on the functionality and user interface of the SAS program editor in the next major version of the SAS System. It is clear that the Institute considers the functionality of allowing the SAS program editor to be replaced by a native editor is an important feature.

Help Systems

Traditionally, the SAS System help system and help authoring system was provided portably. This help was CBT (Computer Based Training) based using SAS/AF®. The advantage of CBT help is that it is portable across platforms, although it is unclear how many customers actually rely on this portability. A SAS CBT based viewer was feature set portable across platforms. With the acceptance of Windows Winhelp and OS/2's IPF (Information Presentation Facility), it was necessary to provide for SAS help in the native format using native operating system provided viewers. Conversions were performed at the Institute, converting catalogs of SAS help into formats that could be used in Winhelp and IPF.



Additionally, to provide for a Windows style help system across non-Windows SAS platforms, Helplus is being provided. Helplus is an Institute developed product that provides viewing capability of help (.hlp) files across Unix platforms. A disadvantage of providing native help, is that SAS customers cannot rely on Institute provided tools to create help source materials. Instead, a help compiler must be provided from an independent source. Providing help viewers in a target native implementation is a direction the Institute will follow for current and future releases.

File Systems

MVA has well supported differing file systems allowing the SAS System to take advantage of the full functionality that the file system offered. (Note this is the case because File I/O services have always been considered a platform dependent component of MVA, meant to be customized on each platform.) File I/O on the mainframe versions of SAS has traditionally been very rich, and recently the PC and Unix file systems have been extended to a 64 bit file address space.

File systems across all platforms have evolved faster than the SAS System's name space. File names across several platforms can be up to 256 characters long, with blanks embedded, and with case sensitivity. The SAS System name space is also being increased as SAS names are also being lengthened to support up to 32 characters for SAS names including variable names, filerefs, librefs, and labels.

Printing

Printing is an excellent example of a subsystem that the SAS System provides a portable implementation. This derived from the platforms where there was no standard print manager services available to applications. Thus the SAS Forms facility was created to allow for formatting capabilities of printed output. Now however, Print Managers are common across PC and MAC platforms, and various Unix implementations are proposing an X based print manager. The native implementations of these print managers are generally more powerful and comprehensive than the SAS Forms capabilities. In addition, modern print managers supply a collection of device drivers supporting all major printers and plotters. The need for SAS/GRAPH® printer and plotter drivers is becoming less necessary on workstation platforms. In fact, usually from an end user's point of view, it is desirable to print or plot using the platform's specific device driver rather than the SAS/GRAPH supplied device driver. The main case for using SAS/GRAPH provided device drivers is the case in which no platform specific device driver is available. On workstation platforms, replacing the SAS Forms facility with native print managers will not create significant portability problems. SAS options that are used to specify the parameters to native device drivers can be isolated in the SAS code or embedded in a SAS macro. The direction for the SAS System is to provide full support of native print managers on platforms where they exist.

Electronic Mail and Fax

Electronic Mail and Fax output support should be consistent with printing. If a data source can be printed, it should readily be available for email or fax output. Although email and fax are similar to printing, it may be possible and desirable to create a portable SAS interface to these services. This is possible due to the fact that the parameters or properties of email and fax are relative simple, contained, and easily identifiable. These properties can be managed in a portable fashion. Research is continuing at the Institute, working to create a standard programmable interface in the SAS System to create email and fax output. Currently, email support has been provided in some form on the PC and Unix platforms, and a MAC implementation is being developed. The platform specific messaging API is not as important. This low level API (VIM, MAPI, MIME, etc.) can be virtualized and shielded from the end user, by merely providing an access driver for the specific interface.

Data Base Access

The SAS System has provided data base access using a combination of proprietary data base interfaces and industry standard data base interfaces. The disadvantage of proprietary interfaces is that the development and support of proprietary interfaces is expensive in terms of development resources and is limited in the breadth of data sources one implementation can access. SAS Institute is pursuing more open data base standards such as SQL and more recently Microsoft's ODBC (Open Data Base Connectivity). Through the SAS SQL procedure pass through engine, ODBC is supported allowing access to a broad range of data base sources. This type of industry standard serves the end user well, providing standard interfaces across a broad range of data bases. The disadvantage with such standards is performance and throughput.. At this juncture in the industry, most ODBC drivers are yet another indirect step to access the data. Since data bases do have differing proprietary interfaces, usually an ODBC driver is another software layer between the requesting application and the proprietary data base interface of the data base. To support native open standards, the SAS System supports ODBC as both a client and server. The server implementation is referred to as the SAS ODBC Driver. This provides end users and other third party software packages access to SAS data sets, with read and write capability.

Communications Access

In the area of communications access software, industry standards have been driven as a necessity from end users. Standards are required so that the computing platforms of differing architectures can operate together. The SAS System has endorsed platform specific and industry standards in the area of communications access methods, providing more robust and economical solutions. The following standards have evolved from vendor specific implementations to an industry standard:

- TCP/IP sockets interface, and Microsoft Winsock
- IBM's APPC
- IBM's EHLLAPI
- Netbios

MVA provides well for the ability to use industry specific standards for implementing communications protocols. As more of the interfaces become industry standard, this drives the standard to be natively implemented for performance increases across multiple platforms. In essence, industry standards drive the cost of a optimized implementation down for each platform, while also providing portability across platforms. SAS Institute is committed to industry standards, for they well serve the end user as well has help to reduce the development cost of the product. MVA is architected to take advantage of industry standards.

Performance Scalability across platforms

The binary image of the SAS System may be run across a variety of hardware platforms. For example, the SAS System for Windows can run across desktop and laptop computers based on the Intel 386 with 8 Meg of memory running Windows 3.1 using Win32s, up to a Windows NT™ Advanced Server running on multiple processors with as much as 256 Meg of physical memory. The SAS System should scale well. This means that algorithms throughout the SAS System should be self configuring based upon the number of processors and the amount of memory. Symmetric Multiprocessing provides for multiple processors to share the same memory address space. Threads of execution could be dispatched to any processor at any point in time. To take full advantage of SMP computers, the SAS System needs to be sufficiently multi-threaded. Multi-threading an application the size of SAS is not an easy task. Discussions continue to be held determining the extent of multi-threading to be used in the SAS System for future releases. Customer direction for SMP based computers will help determine the Institute's future designs.

CONCLUSION

MVA has evolved in step with the requirements of the customer base and the software industry. From the modest inception of creating a data analysis tool portable across mainframe, mini computer, Unix and PC platforms, it has evolved into an architecture providing the flexibility to support industry standards. No where are industry standards more demanding than the graphical user interface standards being accepted across the workstation platforms. MVA will continue to evolve specifically in the user interface area. Support for major functional areas such as nativeness of the user interface, help systems, print managers, email and fax support are currently being developed. In addition, the SAS System will continue to incrementally provide for the support of industry standards. The product feature set of the SAS System is driven from customer requirements and suggestions. By using the SASWARE ballot and by corresponding with SAS Institute marketing and development managers, the future direction of MVA can be influenced.

REFERENCES

Kolb, David (1995), "Integrating the SAS System for Personal Computers into Your Enterprise", Proceedings of the Twentieth Annual SAS Users Group International Conference

Rigsbee, Carol Williams (1995), "Taking Advantage of the SAS System Release 6.11 for Personal Computers", Proceedings of the Twentieth Annual SAS Users Group International Conference

Zeigler, John Carl (July 1989) , "Equipping for a new platform," Unix Review, v7, n7, p56(7).

ACKNOWLEDGMENTS

All Host Division Managers and their staff provided input and suggestions for this paper. I also would like to thank the Institute Host Managers for participating in the panel discussion that follows this paper presentation.