INTRODUCTION

This paper discusses the connection between the SAS® System Release 6.07 under VMS and Digital Equipment Corporation's computing strategy of Network Application Support (NAS). Through MultiVendor Architecture™ (MVA™), conformance to guidelines defined by the NAS strategy, and interfacing to many NAS products, the SAS System fully exploits platform-specific capabilities and gives you a unified, consistent approach to accessing data, other applications, and other systems.

OVERVIEW

This paper first defines Digital's NAS strategy, lists the NAS products, and shows advantages of the NAS strategy. The paper then explains the SAS MultiVendor Architecture and shows the interconnection of the SAS System to the NAS architecture. The majority of the paper discusses how the SAS System interfaces with many NAS products. Future directions of the SAS System with respect to the NAS architecture are also discussed.

WHAT IS NAS?

In today's world, a business is faced with maintaining and utilizing applications in a distributed, multivendor computing environment. Software applications need the ability to communicate to one another across a network; workers need the ability to exchange information quickly and easily in different computing environments; users need the ability to take advantage of resources on different systems without spending time to learn a new system; and businesses need to upgrade to state-of-the-art applications without scrapping current investments and interrupting operations. In order to respond to these problems, you need to impose a standard methodology and provide a standard set of services. NAS is Digital's solution to these problems. It is a solution for integrating applications running on distributed, multivendor systems.

NAS, then, is composed of three parts: first, it is a strategy. The strategy consists of integrating applications across multivendor environments such as VMS, UNIX, OSF/1, and MS-DOS. The most important aspect of this strategy is that it is based upon open, international standards. Second, NAS is an architecture. This architecture defines how the various NAS services work together and make the operating system transparent to the application running on it. Finally, NAS is a set of services and products. The services are actual code, provided to the user as runtime libraries or developer's toolkits.

In general, applications need to prompt the user for input and display information, read and write data, obtain system resources, and communicate with other applications. It follows, then, that NAS services can be grouped into four major categories: application access services (user interface), communication and control services (application interface), information and resource sharing services (data interface), and system access services (system interface). Application access services utilize standards in which the application requests and presents information to the user; communication and control services utilize standards in which the application communicates with other applications; information and resource sharing services utilize standards in which the application stores, retrieves, and accesses data; and system access services utilize standards by which the application can access system resources.

NAS Products

Digital supplies an extensive list of products to support the four different areas of NAS. Application access services are implemented using the following products: DECwindows Motif, DECforms, Terminal Emulation and DEC GKS, DEC PHIGS, and DEC GKS-3D. Communication and Control services are implemented using MAILbus and DEC/ED!, DECmessageO, ACA, DECdtm, and ACMS Desktop. Information and resource sharing services are implemented using CDA Toolkit, Viewers, Converters and DECimage, SQL, CDD/Plus, PATHWORKS, VMS services for DOS, VMS/ULTRIX Connection, ALL-IN-1 File Cabinet, NFS for ULTRIX, and DECprint. Finally, system access services are implemented using POSIX, DCE, EMA, and DECmec.

Most of the Digital products listed above are currently available under the VMS and ULTRIX operating systems. Other operating systems that support most, if not all, of the NAS services include UNIX, MS-DOS, OS/2®, and the Macintosh.

NAS Advantages

By using existing or evolving standards, NAS provides a truly open, universal environment. NAS services and products are available on many systems, therefore, multivendor environments using NAS products behave as one large, homogeneous system. Interoperability is also addressed by
NAS. Since NAS products are built using accepted standards for network access, an application’s function can be easily distributed to other hardware or software environments. Using NAS, then, gives you easy access to other applications on other systems, thereby exploiting all system resources available. This relieves you from having to learn different application packages to perform the same task on another machine, and enables you to share information quickly and efficiently with others in your department.

NAS and the Institute’s MVA

Because of SAS Institute’s MultiVendor Architecture (MVA), the SAS System’s interface to the NAS architecture is well defined. Let me turn your attention now to the Institute’s MultiVendor Architecture to explain the interconnection of the SAS System to the NAS architecture.

Portability, or the degree to which a program can be moved from one system to another, is a major concern at SAS Institute. Like many other software companies, SAS Institute’s products are designed in as portable a method as possible so they can be migrated to other environments in a timely and efficient manner. To help achieve this goal, we have implemented MultiVendor Architecture. This architecture defines three different layers of code: application, core supervisor services, and host supervisor services. Application code contains code necessary to support the various procedures, data steps, functions, formats, informats, and the like. Anything you can put in your SAS program is supported through code at the application layer. Core supervisor services contain code necessary to perform functions applicable on any system, such as memory management services, input and output operations, and loading executable code into memory. Although the actual work of getting memory or opening a file is done at the host services layer, the core supervisor services manage these operations and keep up with returning the information to the application in a defined manner. Note, then, that the core supervisor services keep the host services layer from duplicating functionality needed across all systems. All interfacing to the native environment is done at the host services layer. To move the SAS System from one platform to another requires only modifying the host services layer.

The NAS architecture compliments the MVA architecture by conveniently plugging-in at the host services layer. It is the responsibility of the host layer to interface to the Application Programming Interfaces (APIs) established with NAS. The interface can be surfaced in the form of an engine, (for example, the CDD engine) an access method (for example, the DDIF access method, DECnet, TCP/IP), a windowing environment (DECwindows, terminal mode), or even separate integration files (ALL-IN-1).

Once an interface to a particular NAS product is complete on one platform, the same code can be moved to another machine and plugged-in to that machine’s host services layer. The support, then, of each NAS product becomes the main responsibility of only one host group that can then port the code to all other machines on which the NAS product exists. And because the port requires virtually no changes to the NAS interface code, this allows time for host developers to work on other projects.

NAS Products Supported by the SAS System

Now that NAS and the MVA architecture have been defined, let’s look at how the SAS System interacts with the overall NAS environment, and what NAS products the SAS System currently interfaces with.

The SAS System interacts with four major areas: the user, the data, other systems, and other applications. Note that these areas are quite analogous to the four categories of the NAS services. In each of these areas, the SAS System interfaces to many of the NAS products currently available. In the user area, it supports a DECwindows interface as well as various terminal modes. In the data area, it supports accessing files in CDA format, files described by the COD, and an SQL/Rdb database interface. To communicate with other systems, SAS/CONNECT® and SAS/SHARE® software use DECnet and TCP (supported by DEC TCP/IP Services or DTS) access methods. And through the use of the DECwindows interface, the SAS System interacts smoothly with the PATHWORKS product. Finally, the SAS System supports interfacing to the ALL-IN-1 application. Each of these interfaces is described in the following sections.

User Area

DECwindows

DECwindows is a windowing interface built on the X Window System from MIT. It provides a graphical point-and-click style interface and allows you to run multiple applications simultaneously. The SAS System under DECwindows still has the traditional look and feel of SAS software under any other windowing device; when the SAS System is invoked, all windows, including the OUTPUT, LOG and PROGRAM EDITOR windows, are displayed with the same visual layout. However, running the SAS System under DECwindows has some advantages inherent to DECwindows applications. Output for the display can be
directed to any reachable node using the $SET DISPLAY command. In a similar fashion, the SAS System can be run as a network process (with no attached terminal) and displayed on any X terminal or workstation.

Release 6.07 of the SAS System offers some enhancements. The SAS System can now be invoked directly from the session manager window by defining an action to invoke the SAS command verb. Control-C and Control-Y are now operable as procedure and system interrupts from the scope of any SAS window. The X command is now supported through invocation of a new DECterm window, as opposed to reusing the invocation window. This is due to the fact that the SAS System does not need to be invoked from a DECterm window, as illustrated previously via the session manager window. Support of the host editors (TPU and LSEdit) also includes creating a new DECterm window. In this case, when you exit out of the editor, the DECterm window will be deleted for you. Marking an area by clicking and dragging the mouse is supported under Release 6.07. Cutting and pasting between SAS and non-SAS applications using the mouse is also supported.

Note also that certain procedures require use of a highly interactive graphical user interface. Under VMS, SAS/INSIGHT software and the ISHIKAWA procedure in SAS/QC® software are only supported under the DECwindows interface.

For more information on using the DECwindows interface of the SAS System, please refer to SAS Technical Report P-220, "Changes and Enhancements to the SAS System for the VMS Environment, Release 6.07."

Terminal mode support

The SAS System has long supported a windowing interface under the VT devices, such as the VT100 or VT220. VT terminal emulation is available on VMS, ULTRIX, MS-DOS, and OS/2 environments. On any of these systems, then, you may run the SAS System in the windowing environment by utilizing the appropriate VT device. Although the SAS System supports most common terminal types, Release 6.07 also provides the FSDEVICE procedure which allows you to define your own terminal device for use in the windowing environment.

Data Area

CDA

Digital's Compound Document Architecture (CDA) consists of a series of applications supporting the total manipulation of revisable compound documents. Compound documents are composed of text, data, graphics, and images. CDA services, then, allow for the compound documents to be exchanged or shared among various applications in a single or mulitplatform environment. The CDA architecture defines two main interchange formats — DDIF (Digital Document Interchange Format) for compound documents, and DTIF (Digital Table Interchange Format) for tabular data such as spreadsheet data. The SAS System under Release 6.07 provides an interface to both DDIF and DTIF files.

DDIF files

DDIF files provide the ability to store text, graphics, and images in a single document. The SAS System interfaces to DDIF files through two different SAS abstractions; the first is the DDIF access method. This access method is utilized via the DDIF device-type keyword on the FILENAME statement. The DDIF access method can read DDIF files containing any combination of text, graphics, and images, although only the text portions will be processed. Similarly, the DDIF access method creates DDIF-format files containing text only. Once you have created a DDIF file, you could then use DECwrite, for example, to view and update your file.

The second method in which DDIF files are supported is through two metagraphics drivers, DDIF and DDIFC. The DDIF driver produces monochrome output, while the DDIFC driver produces color output. These drivers are used to create graphics-only DDIF files. By setting up the METAFILE and METAOUT filenames and specifying DDIF or DDIFC as the graphics device, you can output a SAS graph to a file in DDIF format, which can then be visualized using DECwrite or the CDA viewer, for example. Currently, however, the SAS System cannot read and display a graphics-only DDIF file; the metagraphics driver is an output only device. For more information on using the DDIF access method or the DDIF metagraphics drivers, please refer to SAS Technical Report P-220.

DTIF files

DTIF files contain information in tabular form. The DTIF table is a two-dimensional array organized in columns and rows. Additionally, each table is defined by cells, windows, and ranges. A cell is an intersection of a column and a row; a single data element resides in a cell. Windows contain information about what cells in the table to display on a particular device. A logical group of cells within a table or window is called a range. In the SAS System, rows in DTIF tables correspond to observations in a SAS data set, and columns correspond to individual variables within the observation.

Files in the DTIF format can be accessed as if they are SAS data sets using the DTIF engine. The DTIF engine is both an input and output engine. And since it is an engine, you
use a LIBNAME statement to assign the DTIF engine to a physical directory that contains DTIF-compliant files. The DTIF engine cannot access files across DECnet, meaning files found on another node cannot be read or written by the DTIF engine. Also, the DTIF engine is a sequential engine, therefore, access by observation number is not supported.

When using the DTIF engine, all window information is ignored. The only range information used by the DTIF engine is that which describes variable names for the columns.

A cell value within a DTIF file may contain either an actual value or a formula or expression. If a cell value read by the DTIF engine does not contain an actual value, its value is set to missing; any formula or expression information is not supported. DTIF files may also contain Edit String formats. If the DTIF engine finds such a format, it tries to translate it into the comparable SAS format. If this is not possible, the engine issues a warning that the format is being ignored.

Because DTIF files support many VMS data types not supported by the SAS System, data types are converted, thereby potentially causing an overflow. If this happens, a warning is issued and the value is set to missing. If a data type is not supported, like the H-Floating data type, all values for that column are set to missing.

The DTIF engine has some limitations and restrictions. For detailed information on using the DTIF engine, please refer to SAS Technical Report P-220.

SQL and Rdb

SQL (structured query language) is a language used to define and manipulate data stored in relational databases. It describes a standard, easily read language for accessing data. Digital's SQL Services provide an API for VMS, ULTRIX, PC, and Macintosh operating systems. Users on remote systems can use SQL Services to access and update data in a VAX Rdb/VMS database. SQL Services utilize VAX SQL to actually access the data stored in the Rdb or other DRI-compliant database.

The ACCESS procedure in SAS/ACCESS® software uses VAX SQL by dynamically building SQL statements to access data from an Rdb/VMS database. PROC ACCESS enables you to create a view descriptor that is used to directly read, insert, update, and delete Rdb/VMS data, or extract the data into a SAS data file. The interface view engine is part of SAS/ACCESS software but is used transparently to access the data. Version 6 of the SAS System is designed to allow you to use view descriptor files to access Rdb/VMS data directly instead of forcing you to extract the data into an intermediate SAS data set.

Note here that the view descriptors created by PROC ACCESS can then be used throughout the SAS System to read and manage the data. The SAS System supports its own SQL procedure, which can utilize the view descriptors to help you manipulate the data. The SQL procedure is compatible with ANSI standards, and is available on all environments supported by the SAS System.

The DBLOAD procedure enables you to load the Rdb database with data from SAS data sets, as well as send VAX SQL statements (except the SELECT statement) to Rdb/VMS within your SAS session. Supporting VAX SQL statements within the DBLOAD procedure keeps you from having to exit your SAS session, interact directly with the Rdb/VMS database, and then reinvoke the SAS System to manipulate the data.

With Release 6.07 of the SAS System, the SAS/ACCESS interface to Rdb/VMS provides additional functionality by including a line-mode interface to PROC ACCESS. This no longer limits you to using the windowing interface to create the view descriptors. Also, the SQL procedure, in conjunction with the SAS/ACCESS interface to Rdb/VMS, has been enhanced in Release 6.07 to enable you to send VAX SQL statements directly to Rdb/VMS from within the SAS System. This new feature is called the SQL pass-through facility and it enables you to connect to Rdb/VMS, execute query or other SQL statements, and disconnect from Rdb/VMS. You can use the pass-through query statements directly in the SQL procedure, create an SQL view of the data, or retrieve data and store them in a SAS data set. For complete usage information, please refer to SAS Technical Report P-221, "SAS/ACCESS Software: Changes and Enhancements, Release 6.07."

By using the VAX SQL interface to the Rdb/VMS database, the SAS System provides connectivity to Rdb/VMS databases. Data stored in the Rdb/VMS databases can then be accessed via the SQL Services on local or remote systems.

CDD

In order to more effectively share data between multiple applications, users should be able to define data in a consistent manner. Defining data means, among other things, coupling data descriptions with data to indicate the kind of information stored. NAS Dictionary Services provide this functionality, in addition to defining a controlled method to access the data.
VAX CDD/Plus, the Common Data Dictionary, is Digital's implementation of the dictionary services. Data stored in a dictionary not only contain the data definitions, but also the interrelationships between the data. VAX CDD/Plus, then, is ideally suited to store and retrieve information via many applications to show dependencies and relationships among the data.

Implicit in the sharing of data between different applications, VAX CDD/Plus must be able to handle applications running on different environments. The dictionary may reside on any or all of the systems. VAX CDD/Plus supports a distributed environment, thus providing the mechanism to manage and analyze the computing environment.

Within the CDD dictionary, the metadata for an RMS database consist of field definitions, record definitions, a database scheme, and a database definition. Records are composed of a series of fields; a database scheme contains a template for a possible database, and a database definition connects a database scheme to the actual RMS data file. To make an analogy to SAS terms, the CDD field definitions are SAS variables, the CDD record definitions are SAS observations, and the actual RMS database corresponds to a SAS data set.

The SAS System's CDD interface

The SAS System supports the CDD architecture by providing an engine to read RMS databases whose record definitions are stored in the Common Data Dictionary (CDD). Note that although the CDD supports many different kinds of databases, the CDD engine will only read RMS databases. The CDD engine is an input-only engine, meaning the SAS System can read CDD files but cannot create or update CDD files.

A LIBNAME statement is used to point a libref to the VMS directory containing the dictionary. This directory is known as the root or anchor directory. You may also point the libref to any CDD subdirectories within the VMS directory. Note here that CDD subdirectories do not correspond to VMS subdirectories; instead, they are maintained internally by CDD and reside in the same VMS anchor directory.

To access the database within the SAS System, the member name should be the same name as the database definition name defined in the dictionary. If a CDD database name cannot be interpreted as a valid SAS name, you should use the DBPATH data set option to fully specify the database name. The CDD engine supports files of sequential, relative, and indexed organization. Fixed and variable record formats are also supported by the CDD engine.

Like DTIF-format files, the CDD supports many VMS data types other than those supported by the SAS System. If possible, data types are converted to either D_FLOAT or character-type representation, thereby potentially causing an overflow. If this happens, the SAS System issues a warning message and sets the value to missing. Other data types, such as Bound Label Value or H-Floating, are not supported, and all values are set to missing.

The CDD engine does have some limitations due to the requirements of the SAS System's data set definition. Please refer to SAS Technical Report P-220 for a complete usage information on the CDD engine.

Other Systems

DECnet

In order to support seamless integration of applications across systems, you need a method by which one system can communicate (that is, share files and information) with another system. By connecting the systems together in a network, you expand the power and capability available on the system. A DECnet network provides peer-to-peer communication, meaning each node can communicate directly with any other system without the necessity for a central or master system. By using DECnet, users can easily access software applications anywhere on the network. Using the DECnet network gives the user the ability to exchange information, share resources, and perform distributed processing, meaning that information can be located on any node in the network. Typically, a DECnet network is used to connect VAX systems together, but it can also be used to connect a VAX system to other systems such as PC DOS, MS-DOS or OS/2. Additionally, special interconnect products permit nodes in a DECnet network to communicate with non-Digital systems and networks. For example, the DECnet/SNA gateway allows communication between a DECnet network and an IBM® System Network Architecture (SNA) network.

Let me now explain how the SAS System provides distributed processing capabilities via SAS/CONNECT software. Later, I will show how SAS/CONNECT software can use DECnet as a communication method between two SAS sessions.

The SAS System has always provided the capability of accessing both data sets and external data files across DECnet. However, if you want to perform distributed processing, you must use SAS/CONNECT or SAS/SHARE software. For purposes of this discussion, I will concentrate on using SAS/CONNECT software. SAS/CONNECT software enables you to run the SAS System on your local
node and connect to one or more SAS sessions running on remote nodes. This allows you to develop SAS programs on your local node, and then submit them to run on a remote node where the data are stored. You can also transfer SAS data sets, SAS catalogs, or external files between systems via PROC UPLOAD and PROC DOWNLOAD. Finally, you can make use of your local system’s graphics capabilities while running jobs on a remote system. In this way, you can execute the SAS job remotely, but display the graph locally on your terminal, plotter or printer. Please refer to “SAS/CONNECT Software: Usage and Reference, Version 6, First Edition” for complete usage information.

SAS/CONNECT software and the DECnet access method

In order for the two SAS sessions to communicate with one another via SAS/CONNECT software, there needs to be a communication method. This is where the DECnet access method can be used. The SAS System under Release 6.07 supports a DECnet transport method. By using this transport method, you can use SAS/CONNECT software to make a SAS session running on one system communicate with a SAS session running on any other reachable DECnet node.

DTS

DEC TCP/IP Services, or DTS, (also known as the VMS/ULTRIX connection and UCX) is Digital’s package for allowing VMS and UNIX systems to communicate and share data. It utilizes the TCP/IP protocol. DTS defines a client-server model between other hosts and VMS hosts. Other clients communicate with the VMS host using the Internet network protocols (IP). DTS consists of the following components: Internet network, Network File Systems (NFS) server, File Transfer Protocol (FTP) client and server, Telnet client and server, rlogin server, Berkeley Internet Name Domain (BIND) resolver, and Programming Interface.

The Network File System (NFS) provides UNIX users with a transparent method of accessing files and data that reside on the VMS server. Through NFS, the UNIX user can access both VMS files and files residing in an ULTRIX file system on the VMS host.

The File Transfer Protocol (FTP), another component of DTS, enables you to move files from one host to another host. FTP supports transferring files between the VMS host and any other FTP client.

The Telnet utility within DTS uses the Telnet protocol to access other systems on your network. Telnet creates a virtual terminal connection between your terminal and the remote system, making your terminal appear to be directly connected with the remote system.

The rlogin server provides similar functionality as the Telnet utility, where your terminal appears to be directly connected with the remote system. The rlogin server specifically enables UNIX users to access the VMS host system.

Within the SAS System, the TCP access method is used as the communication method to connect two systems supporting the TCP/IP protocol. This means that if you have DTS installed on a VMS node and want to use SAS/CONNECT software to create a remote session on a node supporting the TCP/IP protocol, you would use the TCP access method to accomplish this. Note also that through the NFS component of DTS, you could directly access certain files on the VMS system from the UNIX system. Please refer to “DEC TCP/IP Services for VMS User’s Guide” for more information on the DTS product.

PATHWORKS

PATHWORKS is a software package that connects PC DOS or MS-DOS systems to a VMS host; PC users can run applications locally while connecting to the VMS system to utilize its resources. PATHWORKS provides two kinds of services: file services and disk services. File services allow files on the VMS system to be accessed by both the VMS and PC hosts, provided the filenames meet PC DOS filenames conventions. With file services, raw or flat data files can be shared between the VMS and PC systems without limitations. Disk services provide an area on a VMS disk that is reserved for storing files to be accessed by the PC hosts only: you cannot directly read files stored in this area by the VMS system. Types of files stored using disk services include application software or read-only data files. These files are shared among all PCs connected via PATHWORKS.

SAS System users can use the PATHWORKS product to communicate and share data with other SAS users on the VMS system. SAS data sets can be shared through the PATHWORKS file services, but they must be stored in transport format to be accessed directly. Here, you would use the transport engine to read and write the SAS data sets. Catalogs cannot be shared between the PC and VMS systems; to transport them, you must use the CPORT procedure to put them in a transport format. Then, this file must be placed in a file service directory where the destination system can use the CIMPORT procedure to read the files to the native format. The catalogs are then usable on the destination machine.

If the data sets are not in a transport format, you can still use SAS/CONNECT software to share the data between the PC DOS and VMS environments. PATHWORKS provides a DECnet to DOS software package which enables your PC to...
look like an end node in a DECnet network. On your PC, you
would use the CTERM protocol within the DECnet access
method to process data remotely using SAS/CONNECT
software, and then download the data to your local node.

Finally, since your PC is treated as a reachable DECnet
node, you can use the PATHWORKS DECwindows server
to direct your SAS DECwindows display to your PC while
running the SAS System under VMS. The PATHWORKS
DECwindows server makes your PC behave as a
workstation, including a fully functional window manager.
Under Release 6.06 of the SAS System, you had to set up a
special PATHWORKS DECwindows key map. This map
made keys available to the SAS System that were otherwise
undefined. If you were missing this map, the SAS System
would display numerous warning messages, one for each
key that did not map correctly. Under Release 6.07, the
warning messages are no longer displayed due to a
difference in the keyboard handler routine. However, some
of the PC keys still cannot map to the appropriate key name.
Therefore, the key mapping file is still required in order to
fully use all the function keys defined within the SAS System.
This file can also be used to personalize the key mapping to
your own preferences.

Other Applications

ALL-IN-1

The SAS System now supports an integration with the
ALL-IN-1 product. From within ALL-IN-1, you can use the
familiar menu-driven system to invoke the SAS System in
interactive or batch mode, or with prewritten SAS programs
in noninteractive mode. Output from your SAS job, such as
logs, listings, or even the SAS job itself, can be stored in an
ALL-IN-1 File Cabinet. Files stored in the File Cabinet can
be accessed from within your SAS session, whether they
were created by a SAS System application or another
application. Graph stream files created by SAS/GRAPH®
software can also be stored in the File Cabinet. The
ALL-IN-1 Integration of the SAS System allows you to modify
the format of these files to ReGIS or sixel format where you
can then use ALL-IN-1 mail to send these files to other
ALL-IN-1 users. The ReGIS or sixel format files may also be
included in other File Cabinet documents. In addition, files
can be imported from VMS to you File Cabinet, or exported
from your File Cabinet to VMS. As a convenience feature,
the production version of the ALL-IN-1 integration will
provide an access method to more easily access files stored
in the File Cabinet by allowing you to specify ALL-IN-1 folder
syntax instead of the actual VMS filename.

Another useful feature of the ALL-IN-1 integration is the use
of scratch pads. You can write SAS statements to the
scratch pad, and later include them in a SAS program via
the %INCLUDE statement or the INCLUDE command. The
SAS System also supports an INTERRUPT command via
GOLD-I. When issued, the ALL-IN-1 interrupt menu is
invoked, which allows you to look at mail or do other
ALL-IN-1 functions. When finished, you can quickly resume
your SAS session. This functionality prevents you from
having to exit all the way out of the SAS session, read your
mail, and then reinvoke another SAS session.

Finally, the ALL-IN-1 integration of the SAS System allows
you to run the SAS System from a remote node while
executing ALL-IN-1 on a local node. This is done by creating
defined to invoke the SAS System on the remote node. The ALL-IN-1 integration will
take care of copying any files across the network to your
local machine. Note that if you wish to run the SAS System
interactively from the remote node and display the SAS
windows on your local node, you must use the DECwindows
interface of the SAS System. For information on using the
ALL-IN-1 integration of the SAS System, please refer to
SAS Technical Report P-207, "Using the ALL-IN-1
Integration of the SAS System under VMS, Release 6.06."

Future Direction of the SAS System

Future enhancements to the SAS System include further
support for NAS products. We will fully support Motif using
the X11R4 toolkit and widget set. Note that current support
for the XUI-based version of DECwindows will continue.

Additionally, we will ship an enhanced ALL-IN-1 integration
of the SAS System. The most visible change of the new
integration will be the availability of a File Cabinet access
method. This access method will support the ALL-IN-1 File
Cabinet syntax for specifying document names within the
SAS System. This means you do not have to know the
actual VMS filename of the folder.

Future SAS System development will also include additional
exploitation of mixed text and graphics using Compound
Document Architecture, interfaces to enterprise-wide
system management via DECmec, and expanded support
for using Common Data Dictionary technology. Additional
research is being done in support for multiapplication
message passing, data cut-and-paste, and hyperapplication
support as the SAS System design continues to evolve.

Finally, as standards evolve and new technologies become
part of NAS, we will examine each one and integrate them
into the SAS host services layer of the MultiVendor
Architecture as appropriate.
CONCLUSION

SAS Institute is committed to supporting the NAS architecture by continuing to support interfaces to NAS products. The result of this support is that the SAS System will provide a consistent, familiar user interface under the DEChowes, MOTIF, and VT terminal emulation formats. By supporting the CDA architecture, the CDD dictionary services, and an interface to the Rdb/VMS database via VAX SQL, the SAS System provides easy access to data stored in other formats. The SAS System uses DECnet, DTS, and PATHWORKS to communicate with other systems, and with the capabilities of SAS/CONNECT software, you can easily share data on those other systems. Finally, by interfacing with other applications, the SAS System can be invoked as needed to analyze data.

As NAS products are ported to other systems, the SAS System will already support those products in the new environment. NAS will make it simpler for the SAS System to communicate with its environment through the use of standard products. NAS, then, will work along with the SAS System to provide the user with a consistent, standardized approach to computing.

Reference

Digital Equipment Corporation (1990), NAS Handbook: "Developing Applications in a Multivendor Environment"


Trademarks

SAS, SAS/ACCESS, SAS/GRAPH, SAS/QC, SAS/SHARE, MultiVendor Architecture, MVA, SAS/CONNECT and SAS/INSIGHT are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. IBM and OS/2 are registered trademarks or trademarks of International Business Machines Corporation.

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