Converting from a Windows Based SAS® Computing Environment to a UNIX Based SAS® Computing Environment

Ginger Redner, Merck & Co., Inc., Upper Gwynnedd, PA
Dan Maloney, Merck & Co., Inc., Upper Gwynnedd, PA

ABSTRACT
The process of migrating from a PC based SAS computing environment to a UNIX based environment can be overwhelming for SAS developers, especially when the majority of the staff has little to no knowledge of UNIX. In order to satisfy the needs of the users while maintaining business continuity during this transition, careful planning is required. The pain-points of migrating can be considerably decreased when the appropriate computing architecture is employed, minimal SAS coding standards are followed, SAS session initialization programs are utilized properly and programs to be migrated are thoroughly tested. This paper describes the author’s experiences moving from a PC SAS based computing environment to UNIX, from the perspective of SAS programmers working in a large global pharmaceutical company which is successfully making this conversion.

INTRODUCTION
Our company is in the process of converting from a PC SAS based computing environment to a UNIX based computing environment. This paper discusses the experiences we have had with this transition, from a SAS programmer’s perspective. An overview of the UNIX computing architecture is discussed, along with details on our choice of SAS program development tools. Issues found while migrating our SAS programs to the new platform and how we were able to mitigate them are also covered. The intent of the paper is not to discuss the various tools used in detail, but rather to present an overview of our environment, along with techniques we’ve found that help to minimize the impact of making the transition.

BACKGROUND
Merck and Co., Inc. is a large pharmaceutical company. The main objective of the statistical department is to perform analysis and reporting of patient data collected during clinical trials of our products, and generate tables, listings and figures showing the safety and efficacy results of the trials. These results are then submitted to regulatory agencies, such as the Food and Drug Administration (FDA), to other research organizations and for publication.

In 2011 plans began to upgrade our statistical computing environment for three main reasons. One was due to the adoption of the Standard Data Tabulation Model (SDTM) industry standards. Data stored in the new SDTM format is much larger and requires more time and resources to process. The second reason is that we are a global organization and our analysis and reporting is done in multiple locations around the world. Plans to increase our off-shore staff were underway and we needed a high performing computing environment that would support efficient analysis and reporting across all regions.

Also in 2011, the merger of Merck with another large pharmaceutical company, Schering Plough, was announced. The merger not only increased the number of programmers and statisticians that our computing platform needed to support, but also increased the amount of clinical trial data that needed to be analyzed.

The statistical computing platforms used by the two companies were vastly different. In Merck statistical computing was done in a PC environment. The PC SAS Display Manager System (DMS) was used as the program development editor and as the main tool for program execution. SAS program streams were submitted in batch only for final execution, or when running against large datasets, and the batch programs were executed on the PC or on Virtual Personal Computers. Schering Plough’s statistical computing was based in a UNIX environment.

REQUIREMENTS
Considerable effort was put into determining the requirements for the new integrated computing platform. The basic system needs were evaluated along with the expertise and preferences of programmers and statisticians.
The basic requirements dealt with the volume of data we need to process and the size and geographical locations of our staff. The system needed to efficiently process data from large clinical trials, as well as substantial volumes of data from different trials, concurrently. The computing platform also needed to support the work of our in-house and outsourced programming and statistical staff of approximately 450 located world-wide, plus a varying number of contractors.

Having predominately PC based users who were familiar with the SAS Display Manager System (DMS), we wanted to use a similar program editor and still be able to execute our programs on a quicker and more efficient computing platform. We wanted to avoid having to train all our PC programmers to use another operating platform.

Requirements of the programming environment included:

- Utilizing a navigational tool similar to Windows Explorer for finding and retrieving files and programs
- Executing portions of SAS programs at a time
- Reading and writing to other PC file types, such as Excel
- Accessing files on the PC and future computing platform without porting the data
- Executing programs on the PC and the future computing environment with no modifications

SOLUTIONS

Given our computing requirements and user community, it was quite easy for us to select the UNIX operating system to serve as the foundation of our environment. Most of the data, programs and output are stored on the UNIX server. An overview of the SAS computing architecture is shown in the following display.

SAS COMPUTING ARCHITECTURE OVERVIEW
Overview of Architecture:
- SAS Server: Where all SAS processing occurs.
- Windows Client: Using either SAS Enterprise Guide or PC SAS the users can develop and edit SAS programs, submit programs for execution and access SAS datasets.
- NAS (Networked Attached Storage): File server where all programs, datasets and output are stored, and is directly accessible from the UNIX server and the Windows Client PC.

Key Benefits of Architecture:
- The SAS programmers can take advantage of the user friendly PC development tools and the power of the UNIX server when running their programs.
- Having the same drives accessible to the UNIX server and the Windows Client makes access to programs, input, and output nearly seamless whether you are on the PC or logged onto the UNIX server, without having to FTP any files.
- The SAS programmers have flexibility in determining what tools they want to use to develop and submit their SAS programs.
- The PC SAS engine can be used when necessary.

One of our major concerns when converting from a Windows based SAS computing environment to a UNIX based SAS environment, was to create a SAS development environment which required minimal training for our predominantly PC SAS based users, and still provided an efficient working environment utilizing the powerful UNIX server for program execution. The thought of having to do SAS development directly on UNIX was intimidating to programmers who lacked UNIX experience and were comfortable with the user friendly Windows environment. Luckily SAS provides two methods to develop and run SAS programs with practically no UNIX experience or knowledge:
- PC SAS Display Manager System using SAS/CONNECT technologies
- SAS Enterprise Guide

Within Merck both of these development tools are used and it is the developer’s personal preference as to which tool they use. The purpose of this section of the paper is not to fully explain the SAS/CONNECT or Enterprise Guide technologies but instead to explain how we used those technologies in our development environment.

PC SAS Display Manager System and SAS/CONNECT Technologies
One option that is available to the developers is to use the same PC SAS Display Manager System they use when running SAS on Windows. When using PC SAS DMS to run programs on Unix SAS/CONNECT technologies and commands are used to submit the programs remotely and SAS datasets are accessed via remote libraries. The choice of using this development tool is appealing because it utilizes a tool that SAS programmers are already familiar with, and any customizations in DMS are retained and usable.

Performing the work on a SAS UNIX computing environment from PC SAS requires that the SAS programs be accessible through Windows. This requirement is enabled by mapping the NAS file system drives that hold the programs to drive letters in Windows Explorer. Once these drives are mapped the developers can open the programs in the PC SAS editor as if they were on any other Windows drive.

Accessing datasets on UNIX and running programs using PC SAS requires using several SAS/CONNECT commands and assigning library references that point to libraries on UNIX. We decided to incorporate SAS/CONNECT commands so that we do not need to embed SAS/CONNECT statements directly into the programs. To make issuing these SAS/CONNECT commands as efficient and as transparent as possible we recommend that the developers add the necessary commands as custom toolbar buttons in the SAS Enhanced Program Editor, or associate them with functions keys. The following describes the SAS/CONNECT commands the developers use to run the SAS program statements on UNIX.

1. SIGNON: Connect to UNIX and start a remote SAS session on UNIX
   The first thing that the programmers need to do is connect to UNIX and initiate a remote SAS session on the UNIX server that is linked to the client PC SAS session. This is done by issuing the SIGNON command. When the SIGNON command is issued two options are specified:
   - connectremote: The name of the UNIX server.
- cscript: Identifies the SAS/CONNECT script file used to initiate the session on UNIX. The file can be identified in multiple ways including a file reference or the filename (including the full path) enclosed in quotes.

An example of the SIGNON command follows:

```
signon connectremote=uptwm999 cscript="\prog\share\tcpunix93.scr"
```

2. RSUBMIT: Submit SAS statements on UNIX
Once a remote session is established, SAS programs can be executed in the remote UNIX session by issuing a RSUBMIT command. The difference between the RSUBMIT and the SUBMIT commands is where the program statements are executed.
- When a RSUBMIT command is issued the program statements are run in the remote UNIX session.
- When a SUBMIT command is issued the program statements are run on the client PC session.

Note: Another method to run program statements on the remote session is to include RSUBMIT and ENDRSUBMIT statements within the program code. If developers are using Enterprise Guide or working directly on UNIX these statements cannot exist in the code, therefore we do not use this method.

3. SIGNOFF: End session on remote server.
Once work on the remote UNIX session is no longer needed the SIGNOFF command is issued to end the connection between the client PC session and the remote UNIX session.

The screenshot below shows an example of custom tools created to issue the SIGNON, RSUBMIT and SIGNOFF commands.

Once a remote UNIX session is established accessing datasets on the remote UNIX server is done by utilizing remote libraries. Once the librefs are associated with the remote libraries they can be used like any other library. There are multiple ways to assign librefs to remote libraries. Below are examples of two methods:

- The first method does NOT require a libref to be associated to the library in the remote session. To create a remote library when a libref does not exist on the remote session can be done using the following LIBNAME syntax:

```
libname libref "directory" remote=servername;
```

Example:
```
libname myrlib "/opt/prog/test/stds/data" remote=uptwm999;
```

- The second method does require a libref be associated to the library in the remote session. After the libref has been assigned on the remote UNIX session the following statement will allocate the remote libref to the client PC SAS session:

```
libname libref remote=servername;
```
Example:
libname lptss remote=uptwm999;

SAS ENTERPRISE GUIDE
Another option available to develop and run SAS programs on the UNIX server is SAS Enterprise Guide 4.3. Although Enterprise Guide requires some learning and training, it is a tool that we recommend to our developers because at its core this tool was built to develop and run programs in the client server environment. From a developer viewpoint, to accessing, developing and running programs on a remote server (UNIX) is no different than accessing, developing and running programs locally on a client (PC). In addition, Enterprise Guide provides tools and functionality that does not exist in PC SAS.

The Server List window is one of the first windows that the developer becomes familiar with when using Enterprise Guide. The Server List window provides a clear example that Enterprise Guide was designed to work seamlessly in the client server architecture. This window is used to access SAS programs, libraries and datasets on either the client (identified as the local server) or any remote server that has been configured. From the perspective of an Enterprise Guide user nothing different is done to access a file, library or dataset on a remote UNIX server or the client server.

Note: The servers that are listed in the Server List window are controlled by the SAS system administrator. The details for controlling the list are outside the scope of this paper.

Another example of the client server architecture directly built into Enterprise Guide can be found on the Program tab. On the Program tab menu there is a drop down list box that lets the developer easily select which server the program should be run on. In the Enterprise Guide screen shot example, server UCTPV is the server selected to run the program statements. When running programs using Enterprise Guide there is no need to specify SAS/CONNECT commands to work on the remote SAS server because the functionality is built directly into the development environment.

Enterprise Guide also contains useful and powerful development features that do not exist using PC SAS. Some of the features helpful to programmers are listed below:
- Code completion: As the programmer types in the editor, SAS will display a pop-up list of SAS statements that can be selected, that are appropriate based on the letters typed. As more letters are typed the list is narrowed. Some examples of this would be:
  - If "proc" is entered a list of the SAS procedures is displayed.
  - If "proc print" is entered the list of available proc print options is displayed.
  - If a libname is entered, after the period is entered a list of the datasets in that library are displayed.
• Context sensitive help: When the mouse cursor is hovering over a SAS statement, a pop-up help window is displayed showing information about that statement.

• Status of running code automatically displayed: When SAS program statements are run, the icon displayed next to the program name in the project tree indicates whether there are any errors or warnings in the log. This saves the programmer from having to manually search to determine if errors or warnings exist in the log. In the Enterprise Guide screen shot the red X in the icon next to the test1 program indicates that an error was written to the log when the test1 program was executed.

MIGRATION
Merck's plan for migration was to use a staggered approach. The global macros and programs were tested on Unix, updated as needed, and then copied to UNIX. A few pilot studies were then migrated to the new platform. Once the pilots were successfully completed, all new studies were started up on UNIX. We still have some ongoing studies working solely on the PC, but at the time of this paper a good majority of our work has been successfully migrated to UNIX.

SAS PROGRAM AND MACRO CONSIDERATIONS
One of our major efforts in converting from the Windows based computing environment to the UNIX based computing environment was the migration of the existing programs and macros that were developed and validated in the Windows environment and needed to be tested in the UNIX environment. Our goal was to utilize our existing macros and programs and to minimize the amount of changes needed to our code.

Fortunately, in our Windows based computing environment each project utilized a “startup” program which is where all filerefs and libnames are defined. The “startup” program for each project required modification to run on UNIX, but once the startup programs were updated, only a minimal number of other programs and macros required modification. Since our programmers have the option to work using SAS/CONNECT or SAS Enterprise Guide, we have two separate startup programs. One is the basic initialization program which assigns the standard libname and filerefs, sets some standard SAS system options and allocates the macro autocall search sequence. This standard startup is for use when working with SAS Enterprise Guide or running in batch mode on UNIX. The second startup is used with SAS/CONNECT. This program includes everything in the standard startup, plus the statements for signing into and out of UNIX, as well as statements that assign libnames in the PC SAS Display Manager session to SAS libraries located on the remote server, as mentioned above.

Prior to migrating to the new UNIX computing platform, 182 global macros and standard programs were validated on the UNIX system. The testing involved running a validation program or two on both operating systems and comparing the output. Fortunately, we did not run into many issues during the testing as our macro programming guidelines include items to assist with SAS program portability, such as using only libnames and filerefs in SAS programs instead of actual path names, passing input and output filenames into the macros via SAS parameters, and limiting the use of issuing operating system commands from within SAS programs.

One common problem encountered when migrating SAS programs from one operating system to another is that different systems often have different specifications for how file paths and file names may be defined. On the PC file paths and file names may contain spaces and different folders in a path name are separated by either a forward slash or a backward slash. On UNIX, file names may not contain a space, and the folders in a path name may be separated by a backward slash only. Changes to our standard SAS program which allocates file and library references were required to handle these differences, and in a few cases where hardcoded output file names were specified within the programs, updates were needed.

There’s a small difference in how the higher level folders on the NAS server are defined to SAS when running on UNIX and when mapped to the PC. An example of two of our production folders as defined on UNIX and the PC is shown in Figure 1.
Both of our startup programs can be run on either the PC or UNIX without making system dependent coding changes. To achieve this, we’ve used macro variables to define the high level folders, and used these macro variables in the libname and filename statements. A portion of the SAS code follows:

```
Define Macro Variables for Project and Protocol

%let project  = 0111;
%let protocol = 001-csr;

Define Macro Variables for Standards and Protocol Folder

data _null_;  
  if compress(upcase("&sysscp")=="WIN" then do;  
    call symput("globalstandardspath", "//prog-prod/ stds");
    call symput("protpath", "//prog-prod/ &project/ &protocol");  
  end;
  else do;  
    call symput("globalstandardspath", "/opt/ prog/prod/ stds");
    call symput("protpath", "/opt/ prog/prod/ &project/ &protocol");
  end;
run;

Allocate the Filrefs and Libnames

filename fplyms "&globalstandardspath/macros";
filename fplymssb "&globalstandardspath/macros/submacros";
libname lptda "&protpath/analdata";
libname lptmt "&protpath/metadata";
libname lptss "&protpath/sdtmdata";
```

The second common issue encountered when running SAS programs on different operating systems is the variation found in the higher level decimal digits in output from statistical procedures and functions, due to slightly different algorithms used across systems. We merely needed to be aware of this and assure the differences were, in fact, insubstantial. Use of the fuzz option on the SAS Compare procedure assisted with this effort.

Another area where we encountered problems was reading and writing Excel files. We upgraded our PC environment from Microsoft Office 2003 to Microsoft Office 2010 and found we cannot read/write Excel files that contain multiple sheets and are stored in Microsoft Office 2010 format. The first release of SAS V9.3 does not have a SAS Engine that works in PC and Unix to read/write multi-sheet Excel files created in Microsoft Office 2010 for-
mat. We have heard subsequent maintenance releases of SAS do have the appropriate SAS engine to handle these Excel files. To resolve this issue using the first release of SAS V9.3 we store our Excel files in Microsoft Office 2003 format with the .doc extension prior to reading into SAS programs, and our programs that generate Excel files create them in the MS Office 2003 format.

Some less well-known programming differences were encountered during the testing. One concerned a difference in the default buffer size between the two systems. One of our macros uses the buffer size to split data sets to meet size requirements mandated by regulatory agencies. The size of the split datasets was drastically different when running on the two operating systems, and a program change was required.

SAS catalog names may also be different between operating systems. The SAS code shown in figure 3 was included in many of our macros to delete temporary SAS macros created within the program. The code attempts to delete the temporary macros from the macro catalog. On the PC the macro catalog is called “SASMACR”. When we migrated to UNIX we thought the macro catalog was always called “SASMAC1”, but quickly found that the macro catalog name changes in some cases with the digit on the end of the name being incremented. Rather than adding code to determine the catalog location of the temporary macros, we decided to eliminate this code from all of our programs.

**CONCLUSIONS**

In summary, careful planning combined with the appropriate setup and tools can make the task of converting from a PC based SAS computing environment to a UNIX based SAS computing environment quite seamless for a SAS programmer. Little to no knowledge of UNIX is required, minimal training is needed and business continuity can be achieved. Mapping NAS drives from the PC, using well designed startup programs and utilizing either SAS/Connect or SAS Enterprise Guide software enables program execution on the UNIX box while maintaining the look and feel of PC program development. Lastly, some programming issues may arise when running programs developed on the PC in UNIX, but for the most part they are minimal and easily corrected.

**REFERENCES**


**ACKNOWLEDGMENTS**

The authors would like to thank Ram Radhakrishnan, Sergei Khangulov and Guowei Wu for their valuable input to this paper, as well as their other colleagues who reviewed this paper and provided comments.

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.

**CONTACT INFORMATION**

Your comments and questions are valued and encouraged. The authors may be contacted at:

Ginger Redner  
Merck & Co., Inc.  
351 Sumneytown Pike  
North Wales, PA 19454  
267-305-7634  
Virginia_redner@merck.com

Dan Maloney  
Merck & Co., Inc.  
351 Sumneytown Pike  
North Wales, PA 19454  
267-305-1209  
Daniel_p_maloneyr@merck.com