Cross-Platform Data Migration in a Clinical Environment

Frederick Pratter, Destiny Corporation, Las Vegas, NV
Srinivas Chittela, Purdue Pharma L.P., Stamford, CT

ABSTRACT

Moving SAS® datasets, catalogs, macro libraries and other research data from one platform to another is sufficiently complex in itself, but adding the FDA requirement that all clinical data must continue to be available for years after the study is completed makes the process even more so. This paper relates the steps that were required to move 50,000+ SAS datasets in over 3,000 libraries from outdated HP-UX hardware to Linux. The files were created in Windows and UNIX using V6, V8 and V9 SAS. Each data type required individual handling, and datasets needed a different treatment from catalogs. The procedures developed and the results obtained should be of interest to any organization confronting this kind of cross-platform migration.

INTRODUCTION

The migration and validation of a production clinical data system can be complex at the best of times. It is particularly demanding when data and catalogs from earlier releases of the SAS system have to be taken into account. This paper describes the design and implementation steps that were required at a major US pharmaceutical firm to move a large number of programs, datasets, catalogs, spreadsheets and other files from an HP UX system running SAS 6.12, 8.2 and 9.1.3 to a Linux platform and SAS 9.2. The issue was complicated by several factors. The move included thousands of SAS files that were created in earlier SAS releases. Some of the datasets and catalogs were built using 32-bit versions of SAS while others were 64-bit. In addition, the HP system is available via Samba as a Windows share and there were also many datasets created in various 32- and 64-bit versions of SAS for Windows. The migration had to be accomplished in a single weekend, since some of the files are in daily production use, while others are maintained as a historical database for FDA submission purposes. The process was completed successfully, but several months of planning were required. The major steps (and missteps) in the project are detailed below.

PLANNING AND DESIGN

The staff at SAS have long recognized the need for migration support and there is in fact a PROC MIGRATE, a "Migration Planning Worksheet," and the "Compatibility Calculator," an interactive tool to help in migrating data. Unfortunately, these tools were of limited assistance in the current project, since in order to move across platforms it is necessary to have SAS/Connect® or SAS/Share® enabled on the remote host. In this case, neither was available, so a manual process was required.

The first steps were to identify the files that needed to be transferred and to create a planning document. The German military strategist Helmuth von Moltke (1800-1891) famously wrote that no operational plan survives contact with the main body of the enemy. **Figure 1** illustrates the initial design (which was subsequently modified in the light of experience).

All of the files to be copied were initially in two root directories on the HP host: **/app** and **/data**. A shell script was created to list these. It was determined that there were approximately 20 different SAS file extensions that needed to be considered. Some of these could not be migrated, such as audit files, SAS views and indexes. In practice, it turns out that only the datasets and catalogs can be successfully exported, and of the latter, there was no way to transfer the SAS macro catalogs. Format and source catalogs can be exported however. The final list of extensions for migration was thus reduced to the following short list of six:

- sas7bdat
- sas7bcat
- ssd01
- sct01
- sd2
- sc2

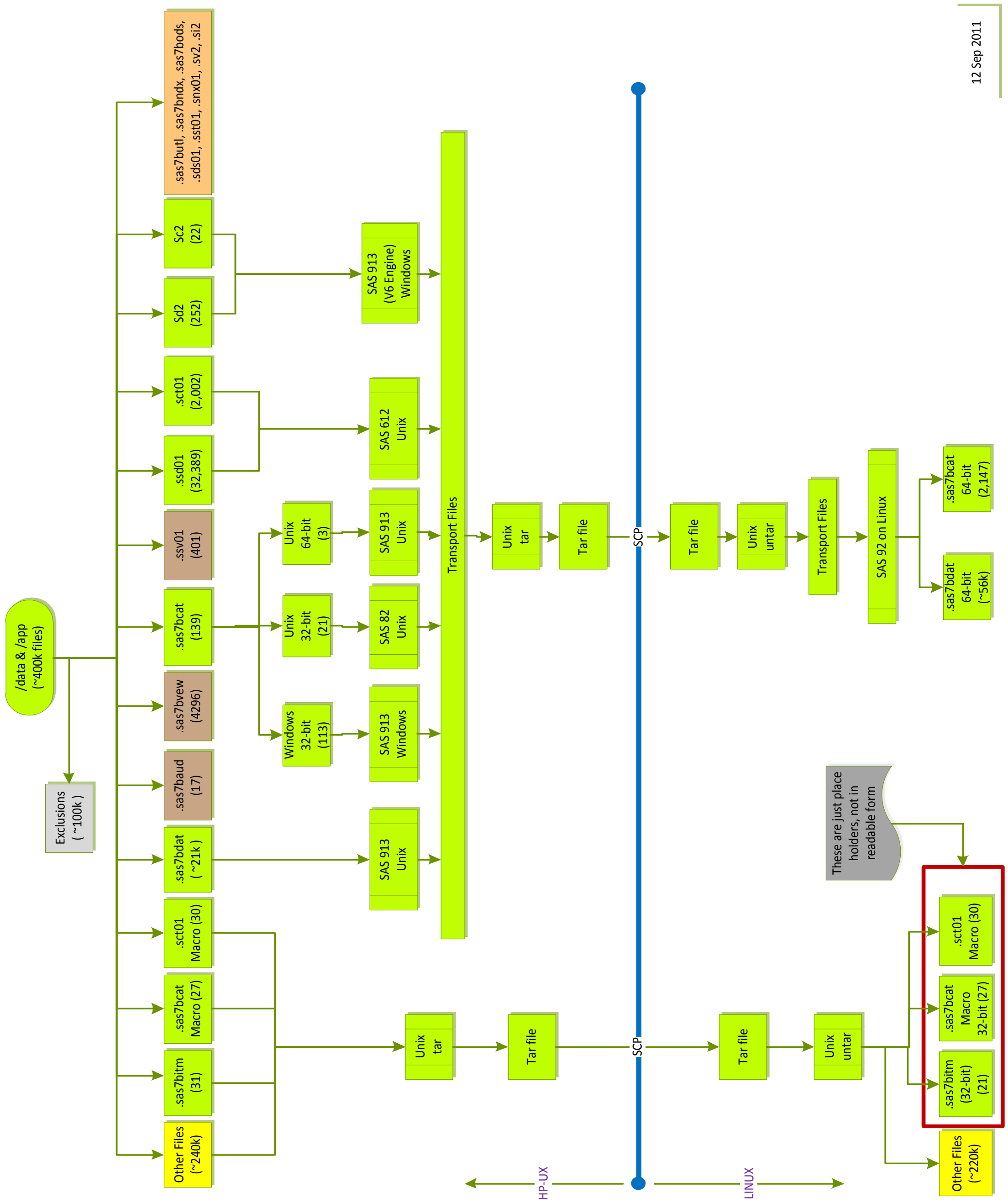

Figure 1. Initial Migration Design Document

For all the other files, the original plan was to create one big archive file using GNU tar utility and copy that over to the new system using `sftp` and expand it. In practice, creating and copying the tar files took so long than an alternative strategy was adopted. A shared network drive was configured, using NFS, available both to the HP-UX source system and the Linux target. All the files in directories (excluding the SAS files) were individually saved to a tar file using the piping feature and shared drive and then expanded into their proper place on the new system.

The next step was to determine how best to accomplish exporting the SAS files given the variety of types involved. **Table 1** shows the results of an empirical investigation, using five different versions of the SAS System.

SAS Extension	OS-version	V612-Unix	V82-Unix	V913-Unix	V913-Windows	V913-V6 Engine-Windows
formats.sas7bcat	(UNIX–32-V8)	Ignored	Processed	Error	Error	Ignored
grafcat.sas7bcat	(UNIX–32-V8)	Ignored	Processed	Error	Error	Ignored
sas7baud	(Win–32-V9)	Ignored	Ignored	Ignored	Ignored	Ignored
sas7bcat	(UNIX-64-V9)	Ignored	Error	Processed	Error	Ignored
sas7bcat	(Win–32-V9)	Ignored	Error	Error	Processed	Ignored
sas7bcat	(Win–32-V9)	Ignored	Error	Error	Entry Type Macro not supported	Ignored
sas7bcat	(Win–32-V9)	Ignored	Error	Error	Processed	Ignored
sas7bdat	(UNIX–32-V8)	Ignored	Processed	Processed	Processed	Ignored
sas7bdat	(Win–32-V9)	Ignored	Processed	Processed	Processed	Ignored
sas7bdat	(Win-64–V9)	Ignored	Processed	Processed	Processed	Ignored
sas7bitm	(UNIX–32-V8)	Ignored	Ignored	Ignored	Ignored	Ignored
sas7bitm	(Win–32-V9)	Ignored	Ignored	Ignored	Ignored	Ignored
sasmacr.sas7bcat	(UNIX–64-V9)	Ignored	Error	Warning	Error	Ignored
sc2	(Win–32-V6)	Ignored	Ignored	Ignored	Ignored	Processed
sct01	(UNIX–32-V6)	Processed	Ignored	Ignored	Ignored	Ignored
sd2	(Win–32-V6)	Ignored	Ignored	Ignored	Ignored	Processed
ssd01	(UNIX–32-V6)	Processed	Ignored	Ignored	Ignored	Ignored

Table 1. File extension by OS -Results

The code used for these tests was as follows: `proc cport lib=INLIB file=OUT_FILE;`

Note that while running SAS V8.2, the program will stop processing at the first encounter of error, whereas SAS V9.1.3 will continue. For macros, the program generated warnings, but the macro entry types are not exported.

Clearly, a strategy had to be found to identify, for each file, the engine and data representation required. This, it turned out was a two step process. For SAS datasets, the option was to use PROC CONTENTS and the SAS Output Delivery System (ODS) to determine this information. There is a sample program in the *SAS 9.2 Procedures Guide* to accomplish this (see the references at the end of the paper). The following code[1] is taken from the example there:

```
      ods output attributes=ATR enginehost=ENG;
      ods listing close;
      proc contents data=IN._ALL_; run;
      ods listing;
```

The output file `ATR` in the example contains the data representation (e.g., `WINDOWS_32`) while the file `ENG` has the release (`9.0201B0`) and the host (`XP_PRO`). In this way, it was possible to decide for each member which SAS version would be used to run the export.

For catalog files, the problem was a little more complicated. Fortunately, there is a SAS Usage note " Sample *34443:* Determine the operating system in which a format catalog was created" that can help (see the references). The following code is based on the sample in the note:

[1] The code examples in this paper are similar to those used in practice but have been modified to make the structure clearer.

```
  /* Read the catalog version and operating system. */
  filename fmt "formats.sas7bcat";

  data;
       length version $120;
       infile fmt lrecl=1000 truncover obs=1;
       input theline $1000.;
       version_loc = index(theline,'2E'x);
       if version_loc>0 then
           version = substr(theline,version_loc-5,30);
       if version ne ' ';
       keep version;
run;
```

In this case, the value of the variable `version` shown in the example is "`9.0101M3XP_PRO`". The hex code `'2E'x` is an ASCII 46 or the dot character. It should be noted that this does not always work, since there may be a period character in the line besides the one embedded in the release name. Looking for an '8' or a '9' instead seems to work reliably.

With these two strategies it was possible to identify which release should be used for each of the file types encountered.

TRANSFER

Following the planning step, a series of SAS programs were created to accomplish the transfer. **Figure 2** illustrates the process in overview.

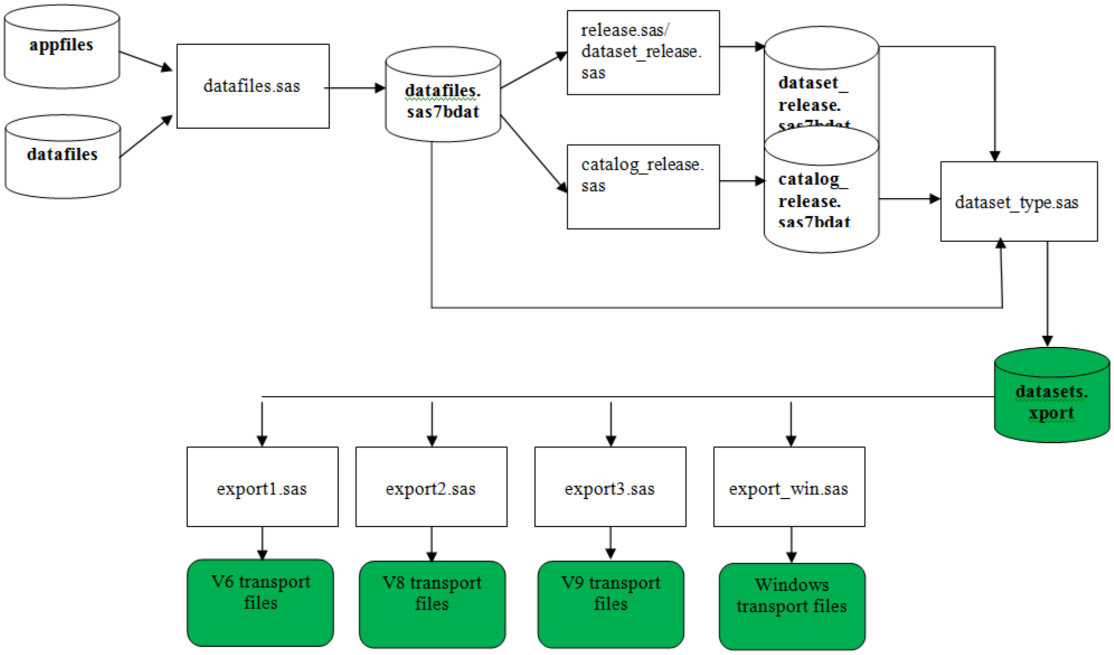

Figure 2. Data Migration Process

The listing of SAS files generated by the shell scripts were input to a SAS dataset called **datafiles**. The code shown above was used to identify the release, engine and data representation, resulting in a combined lookup table **datasets** in SAS export file format. (The latter file had to be a transport file, since it was created in SAS 9.1.3 on HP-UX but read using SAS 9.2 on Linux, and the dataset formats are incompatible.)

Separate programs were then used to generate each set of transport files. All of the SAS transport files were created on the shared mount, so that they could be accessed from either side of the transfer. Note that the **datasets** lookup table included a field that referenced the original directory name, so that the files could be recreated in the equivalent data structure.

Once all the export files were saved to the shared drive, a complementary program was run on the LINUX end to CIMPORT them, using the **datasets** export file as a guide to where they should be located.

VALIDATION

One critical component of any clinical data migration project is validating the resultant data. This was accomplished in two ways. For the datasets, it was possible to create comma delimited files using PROC EXPORT. A random sample of each type of source datasets was created on the shared drive, and on the Linux target host. The Linux `diff` command on the target server could then be used to automate the comparisons.

For the catalogs, the following strategy was adopted: PROC CATALOG with the CONTENTS statement was used to send the contents to an output dataset. This in turn was exported to a comma delimited file and the `diff` command used to compare the results, in the same way as the datasets. It was necessary to specify the `–w` option on the `diff` command to ignore changes in the amount of whitespace and the `–i` option to consider upper and lower case equivalent, since the two platforms produce slightly different output.

Using these two strategies, it was determined that the results of the process were equivalent between the systems. In addition, the structure of the new data could be compared with the original output from the shell scripts, to demonstrate that all of the directories and files that existed on the old server were also on the new. The one complication in this process was that regardless of the extension of the source files, the target files always ended either `sas7bdat` or `sas7bcat`, since they were created by PROC CIMPORT using SAS 9.2 on the host.

CONCLUSION

This complex process was achieved successfully, thanks to careful planning and attention to detail. It is hoped that the description of the migration in this paper will be of use to organizations contemplating a similar move.

REFERENCES

SAS DOCUMENTATION

- 2011. SAS Institute Inc. *SAS® 9.3 Intelligence Platform: Migration Guide*. Cary, NC: SAS Institute Inc.
- 2010. SAS Institute Inc. "Using SAS Files from Other Versions with SAS 9.2 for Windows." *SAS® 9.2 Companion for Windows, 2nd Edition.* Cary, NC: SAS Institute Inc.
- 2009. SAS Institute Inc. "Overview: Migrate Procedure." *Base SAS® 9.2 Procedures Guide*. Cary, NC: SAS Institute Inc.
- 2009. SAS Institute Inc. "The DATASETS Procedure, Example 8: ODS Output." *Base SAS® 9.2 Procedures Guide*. Cary, NC: SAS Institute Inc.
- 2009. SAS Institute Inc. "Sample 34443: Determine the Operating System in which a Format Catalog was created." *Knowledge Base/Samples & SAS Notes*. Cary, NC: SAS Institute Inc. http://support.sas.com/kb/34/443.html.

WEB RESOURCES

- 2011. Virginia Commonwealth University. "Moving SAS Files Between Different Types of Computers." http://www.ts.vcu.edu/kb/2074.html.
- 2011. University of Massachusetts School of Public Health. "Transporting SAS Libraries." http://www.umass.edu/statdata/software/handouts/SASTransport.pdf.
- 2010. Raithel, Michael. "PROC DATASETS; The Swiss Army Knife of SAS® Procedures." *Proceedings of the 2010 Western Users of the SAS System.* http://www.wuss.org/proceedings10/ESS/3037_3_ESS-Raithel.pdf.

- 2004. University of Delaware. "Migrating SAS Files to SAS 9." http://www.udel.edu/topics/software/special/statmath/sas/migrate.html.
- 2004. Karin LaPann. "Handling SAS Formats Catalogs Across Versions." *Proceedings Of The 2004 Philadelphia SAS Users Group*. http://www.philasug.org/ps0410-2.pdf.
- 1998. Ma, J. Meimei and Sandra Schlotzhauer. "Tips and Techniques for Moving between Operating Environments." http://www2.sas.com/proceedings/sugi23/Begtutor/p56.pdf.
- 1996. NC State University Dept. of Statistics. "Moving SAS Data Sets: Transport Files." http://www.stat.ncsu.edu/working_groups/sas/faq/trans.html.

ACKNOWLEDGMENTS

The co-authors would like to thank Lou Florio and Charles Wilmer of Purdue Pharma LP for their assistance and support. Dana Rafiee and Ashli Smith of Destiny Corporation were also instrumental in assuring the success of the project. Finally, the migration could not have been accomplished without the assistance of SAS Technical Support, in particular Jody Graham who remained cheerful and helpful in the face of repeated requests for more details and examples of SAS procedures.

CONTACT INFORMATION

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

Name:	Frederick Pratter
Enterprise:	Destiny Corporation
City, State ZIP:	Las Vegas NV 89117
E-mail:	fpratter@destinycorp.com

Name:	Srinivas Chittela
Enterprise:	Purdue Pharma L.P.
City, State ZIP:	Stamford CT 06901
E-mail:	Srinivas.Chittela@pharma.com