Converting an Old MVS Mainframe Project to Run on Open-VMS
A Guidebook to Success

Steven Wright, Quintiles, Inc., Research Triangle Park, NC

ABSTRACT

Converting data from one platform to another can be a trying experience. In general I have found that the SAS® conversion, *per se*, is generally straightforward. The bigger problem has been the physical moving of the data, especially when tapes have been involved. This paper addresses some of the problems encountered and chronicles the successes of moving a project from MVS to VMS.

The first hurdle to jump was to find a vendor to do the MVS work needed to convert the project to an ASCII format. Tapes were predictably a source of difficulty. Interestingly, none of the tape issues related to readability or reliability of the tapes. Rather things like finding files "too small to read" and unforgiving VMS logicals caused the major problems.

The CPORT procedure was used to create transportable versions of both the catalogs and datasets. Its advantages and disadvantages over the XPORT engine are discussed. A new tool which moves MVS program libraries to a PKZIP® format was a great help.

THE PROBLEM

Back in the good ole days, in the 80's, at Quintiles we ran all our projects in an MVS time sharing environment. It worked well and allowed us to grow quickly because of the relatively massive computational and printing capabilities available. Once projects were completed, we archived them to tape. We used standard MVS procedures, including saving our datasets as MVS SAS tape databases. This made a lot of sense in a pure MVS environment.

As we grew, we brought our computing resources in house, moving to VMS on VAXes and now Alphas. Until recently, when we needed to look at old data, we simply used our MVS vendor or had them move the data for us. However, they have since been bought out and removed from our area and cost arena.

Recently we received a request to retrieve seven-year-old project data and compute some statistics for a different subgroup of the data. We expected some problems getting to the data, but the first hurdle of finding an MVS vendor was somewhat more difficult than expected.

Several problems were encountered reading the converted tape data that were unexpected. We found that not only had the MVS era passed us by, but so had the era of 9-track tapes. Murphy's law has always been hyperactive when dealing with 9-track tapes but now you cannot even find anyone to commiserate with you in your difficulties. The laughable thing now is that a micro cassette can hold at least 2 gig of data, and fit in my pocket without a bulge. A 9-track holds less than 200 meg.

THE PROCESS

We scoured the archives and pulled the relevant SAS tapelabel printouts and associated tapes.

We identified the needed program libraries, SAS datasets and format libraries.

When it came time to bring the data and programs back to real live electronic media, we found that in five years we had been warped into another era, where MVS had little relevant meaning, and where MVS vendors were as scarce as rainy days in San Diego. Our previous vendor had been devoured by a giant who only wanted to deal with other giants. Fortunately, a former employee of our past MVS shop had set up a small business, serving customers like ourselves who just needed a little MVS SAS help in real, not virtual, time. He also brought with him the current knowledge of MVS, which had become crusted over in my mind.

He was able to read the datasets and sent them and the format libraries back to us in ASCII transport format. Though the tapes were about seven years old, we had no problems reading them. Some surprises are truly pleasant.

READING THE CONVERTED TAPE

Reading a converted tape was a piece of cake, or maybe cherry pie. Cherry pies are great--if someone else gets the pits. Of course tapes are fun for just a few; the rest of us must endure.

The first chill came when I saw that the files on the tape were written with CPORT rather the more familiar XPORT format. This is enough to make one nervous and cloud the brain. The next little surprise came when we found that our seasoned MVS person wrote these files as standard labeled tapes. The header and trailer blocks are virtually useless on an ASCII system. They do have the dubious reward of teaching you how to count in 3s, starting at 2!

After testing a number of different VMS DCL permutations, we found it necessary on our system to set the block size on our MOUNT instruction to the block size used when it was created (typically 8000). This might be the default on your system. The code in the CIMPORT section of the *SAS Companion for the Open VMS Environment* was more than adequate to get the job done once the blocksize was specified. I would recommend that you read the section on DCL commands for tapes in this manual, starting on page 178 (2nd edition).

Back to counting by 3, starting at 2. The documentation does discuss how to read from a labeled tape, but I am not sure it is easier than treating the tape as if it were unlabeled. Further, the manual does not explicitly recommend using it with CIMPORT.

When using labeled tapes you need to understand that there are three files per dataset of interest on the tape. First there is the header, then data, and lastly the trailer. Thus when using labeled the tapes as **non**-labeled, the data of interest are on files 2, 5, 8, 11, and so forth. You start by skipping the first file. Then you get into a rhythm: read 1, skip 2, read 1, skip 2, read 1--until you are done. If you wanted to read the fourth data file you could skip 10 files, before you execute PROC CIMPORT to read the eleventh physical file. The VMS command for skipping 10 files is as follows:

```
set magtape device-name /skip=files:10
```

REVISITING CPORT

Data compression is the default for CPORT. After discovering this I was excited to find that these compressed transport files are only slightly larger than conventional compressed datasets. This is a huge advantage over the Version 5 export datasets created using the

XPORT engine in Version 6 of SAS. These can be 10 times as large as the compressed disk dataset. Example code:

```
libname myxpt XPORT 'directory & filename';
  proc copy in=myxpt out=work;
       <select/exclude dsns;>
  run;
```

There is a down side to CPORT. Unlike using PROC COPY, you cannot select or exclude a portion of the files in a directory. You can get either one file or all the files, but nothing in between. There is a SELECT and an EXCLUDE parameter in PROC CPORT, but it applies only to catalogs.

A reasonable work around to this would be to use PROC COPY with an exclude or include statement to move permanent files to the SAS work area, then execute CPORT against the work directory, specifying that you only want data files.

There is a more hazardous but less I/O intensive method. First, rename the files you wish to omit to a different subdirectory or with a different extension. Second, execute the CPORT program. Finally, rename the omitted files back to their old "condition." This is great for keeping I/O down, but there may be a validation issue raised by this method. In VMS, a DIRECTORY /FULL command will reveal that you have ***twice*** modified this file. This leaves these files open to suspicion. You could have completely changed the contents of the file using a SAS update procedure and only left evidence of only one versin change.

MOVING PROGRAMS

Have you ever tried to move an MVS program library to another architecture? In the past we would download the programs ***one at a time*** via our PC based 3270 link. What a drag (and there was no drop)! There is now on the market a PKZIP format tool which takes an MVS PDS and puts the files into the same directory structure, but on the PC. (The product, called "PKZIP MVS", is produced by Ascent Solutions Inc. of Miamisburg, Ohio; www.asizip.com.) Using the "create directory" option (-d) in PKUNZIP, the full MVS structure was mirrored on the PC.

We wanted these programs on our VMS Alpha system but since we did not license PKZIP for the VAX we had to start with them on the PC. Using one of several methods you can map a VMS Alpha volume in Windows. This permits you to move the unzipped files easily using the Windows® File Manager®. Alternately the files could have been unzipped with the PC, acting as a host, and writing directly to the Alpha disk, as a server volume.

I did not follow either method exactly. I wanted to get rid of the numbering in columns 73-80 which is maintained by the MVS software. These columns would quickly have become a problem in a free format ASCII editor. Therefore, I wrote a program in SAS 6.11 to clean up the files and at the same time to move them to the Alpha. The program read the directory structure and file names from a file that was created using a form of a DOS directory command. It then read the old programs, and wrote the first 72 characters out to the Alpha.

One nuance associated with this method was that I had to create the entire directory structure on the VMS side by hand. The first method above, using File Manager actually creates the parallel directory structure on VMS. Therefore if I were to do this again, I would move the

unzipped files to VMS using File Manager and then modify them to remove the line numbering their on VMS.

I decided to be a good efficient chap, so I tried to minimize the amount of disk space used. I was moving hundreds of program files, so I felt a little efficiency was appropriate. Therefore, instead of writing out fixed length records, I used the $VARYING format to try to reduce the size of these program files. This was of only limited value in this case since most programs are much smaller than the smallest block allocated on 8 gig drives, that being 16 blocks on our system. At least I tried!

MISCELLANEOUS PROBLEMS

Several frustrating problems were encountered that could potentially be a snag for others. So at the risk of spreading mud on my face, here are some of the problems which I encountered.

One file in the middle of the tape refused to be read. We finally found that the file had just one block and that the block was not 8000 bytes. The actually length was 7600 as diagnosed by an MVS utility. When I mounted the tape with a block size of 7600, and skipped over the preceding files, I was able to read the file just fine. A time saver: Remember to unmount the tape with the "/NOUNLOAD" parameter if you plan to remount it with different parameters.

It is a normal practice at our site to keep the users from having to deal with hardware names like "1MUA400". Therefore, our systems group assigns a standard logical for us. It looks like "username$tape." This works fine most of the time. But if you log out of VMS before you dismount the tape your logical can get "confused." When this happens SAS will respond to your filename statement with an error indicating that it cannot find the device. The telling thing about the error message is that the tape device name that SAS presents will generally include your default ***DISK*** directory. When this happens, get out of SAS, dismount the tape, deallocate all logicals associated with the tape, and start again. If you find this disconcerting, realize that you have gone from being an applications programmer to a systems programmer in solving this tape problem; and ask for more pay!

FINAL ADVICE

Addressing documentation up front can be a great help in dearchiving old projects. Convince your client that the *name* of the program that generates the final product be noted on that product. A footnote does wonders! For many good reasons most of the key staff that worked on this project still work at Quintiles. But seven years is a long time. The needed tables were "hand abstracted" and so had no program name "stamp" on them. This problem was not insurmountable, but do document more than you think is necessary. As has been said by many and in different ways, "Heroism is no substitute for hard and careful work."

CONCLUSION

Computers are right and programmers must conform. There are some areas of computing that are more fun and user friendly than working with tapes. I hope this paper will help some to be able to move their SAS system from MVS to VMS with almost the ease of a mouse click.

Steven A. Wright, Ph.D.
PO Box 13979
Research Triangle Park, NC 27709-3979
swright@Quintiles.com