

Transferring vs. Transporting Between SAS[®] Operating Environments

Mimi Lou, Medical College of Georgia, Augusta, GA

ABSTRACT

Prior to SAS version 8, permanent SAS data sets cannot be moved “directly” from one platform to another even though SAS supports various operating systems such as Windows, UNIX/Linux, Macintosh and Mainframe. SAS has multiple ways of moving SAS data sets between SAS operating environments: the cross-environment data access (CEDA) facility, the XPORT transport engine with COPY procedure, the CPORT and CIMPORT procedures, and SAS/CONNECT software.

The purpose of this paper is to briefly outline the pros and cons of moving a SAS file, mainly with the CEDA and with the XPORT engine in BASE SAS.

INTRODUCTION

One of the most important features of SAS today is the Cross-Environment Data Access (CEDA). CEDA, as a part of BASE SAS software, is a facility that allows any Version 7 or later SAS data set created on any directory-based host, such as Windows, z/OS UNIX System Services, HP/UX, Solaris, to be read by SAS running on any other directory-based host.

Although different operating environments represent data differently, CEDA can detect the format of the host machine and automatically transfer the format from “native” to “foreign”. The users do not need to know precisely the format for the file that is currently being transferred.

A file or an environment has a “native” format if its data representation is comparable with the host which is processing the file, while a file or an environment has a “foreign” format if its data representation contrasts with the host which is processing the file. For example, a file that is in Windows data representation is native to a Windows environment, but is foreign to that of a UNIX environment.

ADVANTAGES AND RESTRICTIONS FOR CEDA

CEDA has following advantages:

- CEDA translation is transparent, so the user can directly read a SAS data set without knowing if its format is native or foreign.
- A read operation is better performed with CEDA, which uses only one step, than by using a transport file, which needs two more steps.
- CEDA can be used to directly access a foreign file from a Network File Services (NFS) mounted drive, or to process a file that uses File Transfer Protocol (FTP) to go through a binary transfer between hosts.

CEDA has the following restrictions:

- CEDA is available only for operating environments that use directory-based file structures.
- CEDA does not support SAS Version 6 or earlier files.
- CEDA does not support stored programs, DATA step views, or catalogs. It supports MDDb files for read only access.
- Update processing for any SAS files is not supported. CEDA cannot process damaged foreign data set in order to repair it since the update processing is required in this case.

- Indexes are not supported, so there is no WHERE expression optimization with an index.
- The translation might reduce system performance since multiple procedures require SAS to read and translate the data multiple times.

MOVING A SAS DATA SET BETWEEN OPERATING ENVIRONMENTS

The following example shows the output in UNIX as we utilize FTP to move the SAS data set from a UNIX environment to a Windows PC.

```
ftp> get test.sas7bdat
200 PORT command successful.
150 Opening BINARY mode data connection for test.sas7bdat (16384 bytes).
226 Transfer complete.
local: test.sas7bdat remote: test.sas7bdat
16384 bytes received in 0.039 seconds (411.98 Kbytes/s)
```

SAS uses CEDA to automatically recognize the foreign data representation, HP/UNIX, and translate it to the native format of the Windows environment.

However, if you would like to open the same Test.sas7bdat by the SAS System Viewer in Windows, the following error message would stop you:

```
SAS Data set file format not supported
```

CREATING A NEW SAS FILES IN A FOREIGN DATA REPRESENTATION

By default, SAS creates new files by using the native data representation of the CPU currently running the SAS. In order to define the format of data representation, we need this new option in the CEDA technology – OUTREP, which can be used either in the data set or with LIBNAME statement options. This option enables you to create a file within the native environment using a foreign data representation and allows the creator of a SAS data file to decide how the data should be represented in that file.

Some values allowed for OUTREP are:

- ALPHA_VMS
- HP_UX
- MAC
- MVS
- OS2
- RS_6000_AIX
- SOLARIS
- WINDOWS

The OUTREP= option in the LIBNAME statement applies to all files being created in the specified library. For example:

```
libname petstore '/home/mlou' outrep=windows;
```

All data set outputs in the library PETSTORE are represented in Windows format.

The OUTREP= option in the DATA step applies to a specific data set being created. For example, to create a Windows SAS data set in a UNIX environment, we can use following method. This method can enhance system performance because the file no longer requires format conversion when being read by a UNIX machine. Later, you can open the file in Windows with no problem.

```
data petstore (outrep=windows);
input storeName $ cat dog goat rabbit;
datalines;
petlover 12 15 5 20
petco 25 22 15 33
run;
```

The data set PETSTORE is created in Windows format. The following message is shown in SAS Log file:

NOTE: Data file WORK.PETSTORE.DATA is in a format native to another host or the file encoding does not match the session encoding. Cross Environment Data Access will be used, which may require additional CPU resources and reduce performance.

DETECTING THE USE OF CEDA

Because CEDA runs transparently, you might not notice when CEDA is being used. However, detecting the use of CEDA would help you to be aware of the fact that using CEDA might cause the requirement of additional resources and reduction of system performance.

You can set the MSGLEVEL= system option in SAS 9 to detect the use of CEDA.

```
options msglevel=i;
```

After setting the option MSGLEVEL=I instead of its default value, N, once any foreign file was read in SAS 9 a message would appear in the Log file. For example, if we would like to print the results in Windows from the test.sas7bdat, which has a UNIX data representation format, the following message would be shown in the Log file:

```
INFO: Data set WORK.TEST.DATA is in a format native to another host or the file
encoding does not match the session encoding. Cross Environment Data Access will be
used, which might require additional CPU resources and reduce performance.
```

IDENTIFYING A FILE'S FORMAT

You can use either the CONTENTS procedure or the CONTENTS statement in PROC DATASETS to identify the format used. The PETSTORE data set is created in UNIX with a Windows data representation.

```
proc contents data=UNIX.petstore;
run;
```

Following SAS log shows that the output of the data set (Windows) is in a format that is foreign to the native environment (UNIX).

The CONTENTS Procedure

Data Set Name	UNIX.PETSTORE	Observations	2
Member Type	DATA	Variables	5
Engine	V9	Indexes	0
Created	Tuesday, June 6, 2006 10:19:30 AM	Observation Length	40
Last Modified	Tuesday, June 6, 2006 10:19:30 AM	Deleted Observations	0
Protection		Compressed	NO
Data Set Type		Sorted	NO
Label			
Data Representation	WINDOWS_32		
Encoding	wlatin1 Western (Windows)		

Engine/Host Dependent Information

Data Set Page Size	8192
Number of Data Set Pages	1
First Data Page	1
Max Obs per Page	203
Obs in First Data Page	2
Number of Data Set Repairs	0
File Name	/home/mlou/petstore.sas7bdat
Release Created	9.0101M3
Host Created	SunOS
Inode Number	6054
Access Permission	rwxr-x---
Owner Name	mlou
File Size (bytes)	16384

Alphabetic List of Variables and Attributes

#	Variable	Type	Len
2	cat	Num	8
3	dog	Num	8
4	goat	Num	8
5	rabbit	Num	8
1	storeName	Char	8

The data set is represented in Windows_32 format, which is "foreign" to the native UNIX environment, and the native format is SunOS. For the data set TEST, which is "native" to the UNIX environment, the output from the same procedure includes the following information,

```
Data Representation  HP_UX_64, RS_6000_AIX_64, SOLARIS_64, HP_IA64
Host Created        SunOS
```

ALTERNATIVES TO CEDA

If the target machine uses a format that is different from the file format of the host machine (in our example, UNIX), you can read and write, but you cannot update the files.

CEDA vs. XPORT engine

CEDA uses a one-step translation between the source operating environment and the target operating environment, while the process of transporting a SAS file with the XPORT engine involves three general steps:

- Convert the SAS file to the intermediate form known as transport format. For example:

```
libname guestlib 'C:\SASfiles';
libname tranfile XPORT 'A:\example.txt';
proc copy in=guestlib out=tranfile memtype=data;
run;
```
- Physically move the transport format file to the other operating environment.
- Convert the transport format file into a normal, fully functional SAS file, in the format required by the other operating environment. For example:

```
libname hostlib 'C:\Datasets';
libname tranfile XPORT 'A:\example.txt';
proc copy in=tranfile out=hostlib;
run;
```

CEDA provides easier and more precise data representation than does the XPORT engine with PROC COPY. However, due to its restrictions, it might not be feasible to use CEDA sometimes. For instance, update processing for SAS files is not supported by CEDA. In the following example we would like to modify variable TOTAL in the data set TEST, which has been created in UNIX:

```
data UNIX.test;
modify UNIX.test;
total = total + newScore;
run;
```

The following message is written to the SAS Log file in Windows:

```
ERROR: File UNIX.TEST cannot be updated because its encoding does not match the
session encoding or the file is in a format native to another host, such as
HP_UX_64, RS_6000_AIX_64, SOLARIS_64, HP_IA64.
```

Now, one of the solutions is to use the XPORT engine with the DATA step or PROC COPY in order to move files across operating environments, regardless of the version of SAS on the target machine. Also, you can create the transport file one time and target it to multiple platforms that run different SAS releases. Note that the XPORT engine does not support SAS 7 and later features, such as long file and variable names.

CONCLUSION

CEDA provides transparent processing of SAS data set and library members with a foreign format. It creates an easier and faster method to access data between SAS operating environments. Meanwhile, using the XPORT engine makes up the deficiency of CEDA, especially, when sending a transport file to a destination host whose SAS release is unknown.

REFERENCES

SAS Institute Inc (2004), "SAS ® 9.1.3 Language Reference: Concepts", Cary, NC: SAS institute Inc.

SAS Institute Inc (2004), "Base SAS ® 9.1.3 Procedures Guide", Cary, NC: SAS institute Inc.

SAS Institute Inc (2004), "Moving and Accessing SAS ® 9.1 Files", Cary, NC: SAS institute Inc.

SAS Institute Inc (2004), "SAS ® 9.1.3 Language Reference: Dictionary", Cary, NC: SAS institute Inc.

CONTACT INFORMATION

Mimi Lou
Department of Biostatistics, Medical College of Georgia
1469 Laney Walker Blvd.
Augusta, GA 30912-4900
Work Phone: (706) 721-4453
Fax: (706) 721-6294
Email: mlou@mcg.edu
Web: <http://www.mcg.edu/research/biostat/>

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 trademarks of their respective companies.