

Pros and Cons of X command vs. SYSTASK command

Irina Walsh, ClinOps, LLC (formerly Pacific Data Designs, LLC and ClinPro, Inc.)

ABSTRACT

Moving and managing files across a network can be easy or sometimes challenging. There are a few ways to copy and move files across a network or between directories. This paper details and compares some applications of SAS X command and SYSTASK command. The pros and cons of both commands are discussed. Examples of techniques to achieve successful command execution and error handling control are offered.

INTRODUCTION

The SAS X command and SYSTASK command are used in order to execute operating system commands. These commands provide a fast approach when you need to copy, move or manage files and directories. There are many ways to copy and move SAS files across a network. One way is to use PROC DATASETS within the SAS program. However, SAS X command and SYSTASK command are two statements that can manage SAS files as well as files originating from other software programs. The fact that these two commands can manage non SAS files within the SAS environment makes them a powerful tool to use.

This paper highlight the difference in use of SAS X command statement vs. SYSTASK command and also provides some simple applications of both these commands. It is primarily directed at the novice user of the SAS Software. The examples in this paper are processed by using the PC MS Windows XP operating system and the SAS version 9.1.3. These examples also executed successfully on SAS network version 9.1.3 on Windows server 2003.

COPY FILES WITH X COMMAND

The simple way to copy, for example all files from one folder to another using SAS X command is:

```
X "copy c:\temp\*.* C:\temp\CopyFiles\*.*";
```

This command will copy all files located in "C:\temp" folder. It could be any files i.e. jpeg, xls, mp3 etc. Although this command is simple to use when coping files from one destination to another, it has some flaws. When using X command there are no error code returned to the SAS environment. For example if a file is open by someone on the network while the X copy statement is trying to execute the commander window will return the following message:

```
c:\temp\comp.txt
c:\temp\compdoc.txt
c:\temp\comprtf.txt
c:\temp\test1.doc
The requested operation cannot be performed on a file with a user-mapped
section open.
c:\temp\test1.rtf
c:\temp\test1.sas
c:\temp\test2.doc
c:\temp\test2.rtf
c:\temp\test2.sas
      8 file(s) copied.
```

However, there will be no error message in the SAS log. This issue could be overcome by writing the output of commander window to the text file. Adding symbol '>' after the command line will send output of the command window to the designated text file. If file already exist it will be overwritten. Let's see step by step how we can create a simple error control from the X command.

At first we need to read all files in the source directory that we want to copy. We use /b option in order to list one file name with extension per line and suppress printing of headlines and summary. We also use option /o:g in order to sort file names and group subdirectories before files. We will output the list of files in the "c:\temp" folder to the report.txt file.

```
X "dir c:\temp /b/o:g > C:\temp\CopyFiles\report.txt";
```

Other useful options available for dir statement

/o:n	Sorting order by name
/l	Displays unsorted directory names and file names in lowercase.
/n	Displays a long list format with file names on the far right of the screen.
/c	Displays the thousand separators in file sizes.
/4	Displays year in 4 digit format

After X dir command is executed the report.txt file is created. The content of the file is as follows:

```
CopyFiles
comprtf.txt
compdoc.txt
comp.txt
test1.doc
test1.rtf
test1.sas
test2.doc
test2.rtf
test2.sas
```

You can see the list sorted by file names with subdirectories listed before files. Now we can read this text file into the SAS dataset and count number of files need to be copied:

```
data soursel;
    infile 'C:\temp\CopyFiles\report.txt' trunccover lrecl=512;
    input @1 datafld $30. ;
    ***Delete subdirectories;
    if index(datafld, '.')=0 then delete;
run;

*** Count number of files that needs to be copied;
proc sql;
    select count(*) into: totfl
    from soursel;
quit;
```

This code can be easily modified if only subset of files need to be copied. For example if only excel files need to be copied the code inside the data step can be modified to:

```
<if index(datafld, '.xls')=0 then delete;>
```

Now we can execute the copy command and redirect the output of command window to the text file and use the same approach as above to see if all files were copied successfully. If we need to copy all files in the directory we could use wild character *.* , if we needed to copy only excel files use *.xls or to copy a specific file we will use filename.txt. As an option we can add "abort abend;" statement in order to stop program execution and close SAS if not all files are copied. This option we can use if we intend to run the program independently and don't want the execution of the any following code if the copy command wasn't successful.

```
options noxwait ;

X "copy c:\temp\*.* C:\temp\CopyFiles\*.* > C:\temp\CopyFiles\reportcp.txt";

data distlst;
    infile 'C:\temp\CopyFiles\reportcp.txt' trunccover lrecl=40;
    input @1 pathfld $30.;
    if (index(pathfld, 'file(s) copied.')>0
        and compress(pathfld, 'kd')<compress("&totfl"))
    then do;
        put 'ERROR: Not all files have been copied';
        abort abend;
    end;
```

```

else if (index(pathfld,'file(s) copied.')>0
        and compress(pathfld,,'kd')=compress("&totfl"))
then put 'NOTE: All files have been copied successfully';
run;

```

In order to return control from command window to SAS without typing 'exit' in command window we need to use NOXWAIT option. By default X command executes synchronously with SAS, therefore any SAS proceeding code after an X command will be executed after an X command is completed. If you don't need to wait until X command is executed and need to continue to run proceeding code than use NOXSYNC option. You can also use SYSRC macro variable. The return code from X command will be stored in SYSRC automated macro variable and will have value of 0 if X command executed successfully. However SYSRC variable hold a return code from the last operating system command and only useful when you employ it after each X command; if you run few X commands simultaneously SYSRC will return only code from last X command. Please note the copy command will allow only copying files in source directory with size more than 0 bytes long.

COPY FILES WITH SYSTASK COMMAND

The same task as above can be easily executed using the SYSTASK command in SAS. By default SYSTASK command executes asynchronously with SAS, therefore if we need to wait for the result of the SYSTASK command before we continue we need to use WAITFOR or WAIT options. WAITFOR option suspends the execution of the SAS program until one or few SYSTASK commands are completed. In the following example we use WAIT option to ensure that data _null_ step continue after SYSTASK command is completed.

```

systask command "copy c:\temp\*. * C:\temp\CopyFiles\*. *" wait status=copyfl;

data _null_;
  if &copyfl>0 or &sysrc>0 then do;
    put 'ERROR Occured during the copy statement';
    abort abend;
  end;
  else put 'NOTE: All files have been copied successfully';
run;

```

The returned code from SYSTASK command is stored in automatic macro variable SYSRC. If the value of SYSRC=0 then the SYSTASK command has been executed successfully, otherwise if SYSRC>0 some problems have occurred. SYSRC variable works the same way for SYSTASK command as for the X command. The status macro variable 'copyfl' indicates if the copy command in this case was successful or not and returns an error code from the operating system command.

MOVE FILES

In a similar way as we processed the copy statement we can process the move statement. In X command we will add option "/y" to suppress a confirmation to override existing file if file with the same name is exist at the destination. When using SYSTASK COMMAND we don't need to use /y. We could get information we need from reportmv.txt to retrieve results and create a report.

```

options noxwait;

X "move /y c:\temp\*.doc C:\temp\CopyFiles\ > C:\temp\CopyFiles\reportmv.txt";

systask command "move c:\temp\*.doc C:\temp\CopyFiles\" wait status=movefl;

data _null_;
  if &movefl>0 or &sysrc>0 then do;
    put 'ERROR Occured during the move statement';
    abort ;
  end;
  else put 'NOTE: All files have been moved successfully';
run;

```

OTHER USEFUL APPLICATION OF X COMMAND OR SYSTASK COMMAND

ZIP FILES

Creating zip files programmatically using the X command or SYSTASK command can help to reduce errors caused by manual process. In the following example we zip all files in the "c:\temp\CopyFiles" folder and zip only text files from the same folder.

```
systask command 'c:\progra~1\winzip\winzip32.exe -a -s"true111"
C:\temp\CopyFiles\Transfer.zip C:\temp\CopyFiles' taskname=zipall

systask command 'c:\progra~1\winzip\winzip32.exe -a -s"true113"
C:\temp\CopyFiles\Savetxt.zip C:\temp\CopyFiles\*.txt' taskname=ziptxt
status=zptxt;
waitfor _all_ &zipall &zptxt;
```

In order for this code to be executed on the network version of SAS, the WinZip have to be installed at the same root directory on the server where the SAS is installed. In this example we assign taskname macro variable and use WAITFOR option in order to wait until both SYSTASK commands are completed.

APPLY ATTRIBUTES

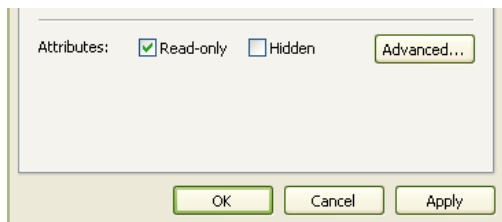
Either statement can be used to apply ATTRIBUTES to the file or files in directory:

```
x "attrib -r c:\temp\*.*";
systask command "attrib +r c:\temp\*.*" wait status=attrfl;
```

The following list of attributes is available:

+r	Sets the read-only file attribute.
-r	Clears the read-only file attribute.
+a	Sets the archive file attribute
-a	Clears the archive file attribute.
+s	Sets the system file attribute.
-s	Clears the system file attribute.
+h	Sets the hidden file attribute.
-h	Clears the hidden file attribute.

Applying attributes statement will be the same as applying attributes using file properties



ARCHIVE OR BACKUP DIRECTORY

Because of the speed and ability of the X command and SYSTASK command to manage non SAS files these statements are very useful when you need to archive files or directories, reduce the storage space and create report. The following program shows how you can manage your archival process.

The copy command can copy only files, whereas the xcopy command can copy files, directories with subdirectories and files with a size of 0 bytes. To make sure the xcopy command works properly use command/c in your statement to initiate new DOS command-processor. Use backslash (\) in destination directory name or use /i to specify that the directory needs to be copied. If you want to output the list of the files to be copied redirect output to the text file as in the following example .

```
X "command/c xcopy c:\temp c:\trynew\ /s/i/e > c:\List.txt";
```

Below listed some options that can be used with xcopy command:

/s	Copies directories and subdirectories, unless they are empty.
/e	Copies all subdirectories, even if they are empty
/t	Copies the subdirectory structure (that is, the tree) only, not files.
/l	Displays a list of files that are to be copied.
/c	Ignores errors.
/y	Suppresses prompting to confirm that you want to overwrite an existing destination file.
/i	If Source is a directory or contains wildcards and Destination does not exist, xcopy assumes destination specifies a directory name and creates a new directory.

The following example is a program for routine archival of directories with output report file. This example uses a SYSTASK command because of its flexibility; alternatively the X command could be used. In this program we copy files to archival directory ArchiveLocation, apply read only attributes to avoid anybody making a change, compress the directory and create a report.

```
systask command "xcopy C:\temp C:\ArchiveLocation\ /s/i/c/e" wait
status=archiv;
systask command "attrib +r c:\ArchiveLocation" wait status=attrib;
systask command "command/c compact /c /s:C:\ArchiveLocation >
C:\ArchiveReport&sysdate..txt" wait status=comprs;

libname archiv "C:\ArchiveReports";
data archiv.report;
if &archiv=0 and &attrib=0 and &comprs=0 then do;
  infile "C:\ArchiveReport.txt" truncover lrecl=1024;
  retain DIRLOC FILENAME ORIGSIZE REDCSIZE RATIO DIRLOC TRANSFDT TRANSFTM
  USERINFO;
  input @1 DIRECTORY $60. @;
  if index(directory, 'Compressing files in')>0 then do;
    DIRLOC=substr(directory, index(directory, 'C:\'));
    TRANSFDT= "&sysdate9";
    TRANSFTM="&SYSTIME";
    USERINFO="&SYSUSERID";
    input FILENAME $1-21 @;
    delete;
  end;
  else if index(directory, '=')>0 and index(directory, '.')>0 then do;
    input FILENAME $1-21 ORIGSIZE 22-29 REDCSIZE 33-41 RATIO $44-53 @;
    if origsize=. then delete;
  end;
  else delete;
  keep DIRLOC FILENAME ORIGSIZE REDCSIZE RATIO DIRLOC TRANSFDT TRANSFTM
  USERINFO;
end;
else do;
  put 'ERROR is ocured during the sustask command execution';
end;
run;
```

In this example we use WAIT option in SYSTASK command to make sure each statement executes only after previous statement is completed.

You can use PROC APPEND to append all consecutive reports to one dataset and that way keep track of your archives in one place. The following dataset is stored in ' C:\ARCHIVEREPORTS'

	DIRLOC	FILENAME	ORIGSIZE	REDCSIZE	RATIO	TRANSFDT	TRANSFTM	USERINF
1	C:\ArchiveLocation\	comp.txt	91	91	1.0 to 1	10JUL2007	19:19	IVW
2	C:\ArchiveLocation\	compdoc.txt	30495	20480	1.5 to 1	10JUL2007	19:19	IVW
3	C:\ArchiveLocation\	comprtf.txt	2639	2639	1.0 to 1	10JUL2007	19:19	IVW
4	C:\ArchiveLocation\	CopyFiles	0	0	1.0 to 1	10JUL2007	19:19	IVW
5	C:\ArchiveLocation\	CopyFilesNew	0	0	1.0 to 1	10JUL2007	19:19	IVW
6	C:\ArchiveLocation\	reportcp.txt	246	246	1.0 to 1	10JUL2007	19:19	IVW
7	C:\ArchiveLocation\	test1.doc	198144	172032	1.2 to 1	10JUL2007	19:19	IVW
8	C:\ArchiveLocation\	test1.rtf	338406	299008	1.1 to 1	10JUL2007	19:19	IVW
9	C:\ArchiveLocation\	test1.sas	9022	8192	1.1 to 1	10JUL2007	19:19	IVW
10	C:\ArchiveLocation\	test2.doc	198144	172032	1.2 to 1	10JUL2007	19:19	IVW
11	C:\ArchiveLocation\	test2.rtf	338418	299008	1.1 to 1	10JUL2007	19:19	IVW
12	C:\ArchiveLocation\	test2.sas	9022	8192	1.1 to 1	10JUL2007	19:19	IVW
13	C:\ArchiveLocation\	ZeroBytes.txt	0	0	1.0 to 1	10JUL2007	19:19	IVW

The compact statement will reduce the disk space used by the file. This statement will be the same as if you open a file properties and in Advance attributes tab check " Compress contents to save disk space".

If you check any file properties you will see the size of the file is reduced:

Location:	C:\temp
Size:	193 KB (198,144 bytes)
Size on disk:	168 KB (172,032 bytes)

If you need to uncompress files use option /u as it is shown in next example.

```
systask command "command/c compact /u /s:C:\ArchiveLocation " wait
status=uncompr;
```

CONCLUSION

Whether you choose to use X command or SYSTASK command you have to remember that by default the SYSTASK command executes asynchronously, whereas the X command is synchronous therefore you have to wait for control to return to SAS. However, XWAIT/ NOXWAT, XSYNC/ NOXSYNC options for X command and WAIT/ NOWAIT, WAITFOR options for SYSTASK command can help you to control the way you want either of these commands to be executed. The SYSTASK command provides great flexibility and error control with macro variables assigned to SYSTASK status and taskname. When you execute one command at the time the error control for both X command and SYSTASK command can be handled with automatic macro variable SYSRC. However, when you need to execute a few statements simultaneously X command is not a best choice to use. The X command does not provide any command-specific returned code. In order to have some error control from X command you have to output the results of the command to a text file and then investigate this file. Good advantage of the SYSTASK command that it doesn't use a command-prompt window. Also SYSTASK command is a better choice when you need to run a few SYTSASK commands one after another. Using the status and taskname macro variable you can easily control a process of your program.

REFERENCES

Stephen Hunt, Brian Fairfield-Carter (2007), "Zipping Right Along: Push-button SAS® Transfers via Command-line Invocation of the WinZip32 Executable" SAS Global Forum, Paper 005-2007

Na Li, "Old Fashion but Still Useful: Applications for Running the X Statement to Issue DOS Commands in the Windows Operation System", WUSS 2005

Judie Gilmore, (2004) "Painless Windows: A handbook for SAS® Users", Third Edition, Cary, NC: SAS Institute, Inc.

SAS Institute Inc. (2004), "Running DOS or Windows Commands from within SAS" SAS OnlineDoc, Version 8, Cary, NC: SAS Institute, Inc.

Microsoft Windows XP Professional Product Documentation.

<http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ntcmds.msp?mfr=true>, © 2009 Microsoft Corporation.

CONTACT INFORMATION

If you have any comments and questions regarding this paper, please contact:

Irina Walsh
ClinOps, LLC (Pacific Data Designs, LLC and ClinPro, Inc. are now ClinOps, LLC)
353 Sacramento Street, Suite 800
San Francisco, California 94111
(415) 776-0660 ext. 199 FAX (415) 776-0746
E-mail: Irina.Walsh@clinops.com

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.