A SAS®-based solution to perform a documented selective copy of files on Microsoft windows platform

Frederic Coppin, MSD (Europe), Inc., Brussels, Belgium
Carl Herremans, MSD (Europe), Inc., Brussels, Belgium

ABSTRACT

Within clinical biostatistics departments and scientific programming organizations, SAS programs and data may be organized together in study-based standard directory structures (SDS) with access restricted to certain team members. In order to promote efficiency and standardization in the development of SAS programs across studies, it is desirable at some point in the study lifecycle to share all SAS programs with the whole community of SAS code developers within the department.

The PUBLISH and DIRCOPY macros, presented in this paper, allow the user to selectively copy, in a documented manner, the non-confidential files [mainly SAS programs] to an environment accessible to the members of the SAS programming community. This type of operation is not easily supported by the Microsoft Windows operating system itself.

The PUBLISH macro generates a graphical window interface for the user to browse the source study standard directory structure. The interface also allows the user to browse the non-confidential material, selecting the files to promote to the target environment. The DIRCOPY macro on the back end copies the files from the selected SDS directories, including their subdirectories. This is achieved using the SAS MODULEN function calling the WIN32 APIs CreateDirectoryA and CopyFileA.

Following a discussion of business needs for selective copy, this paper will detail the outer and inner macro design as well as the windowing interface of the PUBLISH macro. This paper will also demonstrate how to implement the selection of files. A technical section will be dedicated to the presentation of some critical SAS code. Finally, the possible extension of this tool will be discussed.

INTRODUCTION

Whenever a SAS developer is confronted with a new programming task when supporting the analysis of clinical trials, he will always consider the following question: "*Did I or a colleague of mine already face something like this before?*". In order to benefit from the development effort put in previous clinical trials, it is highly desirable that the entire community of SAS® program developers has access to all programs (including their related documentation files) supporting prior or concomitant trials. This encompasses the access to a departmental macro library but also includes all SAS regular open codes developed on a per-study basis.

On the other hand, regulatory matters and strategic perspectives may require restricted access to the clinical trial data and results prior to public disclosure. The access to the associated SAS® programs may also be subject to a restricted access.

This paper presents and discusses a macro which resolves the trade-off between efficiency [full read-access to all SAS codes for the entire SAS® programs developer community] and information security [a more restrictive access to more confidential assets like clinical and analysis data].

STANDARD DIRECTORY STRUCTURE

Within their clinical biostatistics departments most pharmaceutical companies have set up a submission development environment [referred to herein as SDE] [See, Barry R. Cohen (2000) and F. Coppin, C. Herremans (2003)].

In accordance to good programming development practices this SDE typically consists of several areas following the phases of the program life cycle:

- Development area
- Test area
- Production area

As the production area contains the unblinded results an increased access control level is implemented on this area. This control level limits the read-access to a limited set of persons from the project team.

As stated before, in order to promote efficiency and standardization in the development of SAS® programs across studies, it is desirable at some point in the study lifecycle to share all SAS® programs with the whole community of SAS® code developers within the department. While protecting the integrity of (unblinded) data and programs, the restricted read-access in the production area implies that the SAS® programs are not accessible to the other members of the SAS® programming community.

In order to circumvent this drawback an additional area can be put in place: "a Publish area" (as summarized in Table 1). This Publish area will allow full read-access to the non-sensitive SAS programs to the whole community of SAS® code developers within the department.

It contains a copy of the non-confidential files available in the production area, e.g., SAS® analysis programs, SAS® macros. This copy to the publish area should be scheduled by the program development team at the moment the programs are finalized.

Environment	Description
Development	Area where the program development takes place.
Testing	Area where testing rounds are performed and stored.
Production	Area where the program is run by the end-user on the unblinded data and where the results are to be incorporated in the CSR.
Publish	Area where a copy of the program is stored once the project is finalized. That area can also be used as a library of codes for the developer community provided that a read-only access is maintained in that area to the whole community.

Table 1— Collection of programming environments

SELECTIVELY COPY

Quite often the SDE environment can be mapped through a Microsoft Windows share composed of several standard directory structures [SDS] organized on a per protocol basis. For more details reference is made to Andrea Dynder, Barry R. Cohen, Gary Cunningham (2000).

For the transfer of these non-confidential files from the production area to the Publish area a file management tool such as MS/Explorer or MS/File Manager would typically be used. However, the manual and, at the same time, consistent selection of the files that are eligible to be copied from the production to the Publish area, constitutes a huge and complex task, and is a source of potential errors.

Typically the files eligible to be published are stored in dedicated sub-directories of the SDS. In order to avoid that confidential information is mistakenly published, certain file types identified by their file extensions are to be excluded. Files classified to be confidential may for example be SAS® datasets and SAS® transport files, but depending on the project this could also be MS-EXCEL or MS-ACCESS files. Therefore, in order to automate the process of publishing a project, a selective copy is to be made excluding all types of data files and limiting itself to a certain set of sub-directories within the project directory.

The requirements for the macro to make the selective copy from the production area can be summarized as follows:

The macro has to perform on a protocol basis a selective copy from the production area to the publish area. This selective copy shall be based on a set of selected directories containing non-confidential files. In other words, an exclusion of all type of data files based on their file extension shall be realized.

From a program design perspective, the job consists of two parts. The first part consists of the interface that allows the user to specify the project to be published. The second part consists of the actual selective copy of the non-confidential files.

For each of these tasks a dedicated macro was developed: the PUBLISH macro covers the interface part; the DIRCOPY macro covers the actual selective copy.

PUBLISH MACRO

The PUBLISH macro is responsible for the user interface enabling the selection of the project to be published as well as the identification of the directories containing the non-confidential files suitable for publication. This macro is also responsible for defining another set of parameters like the directories and file types to be excluded by default. As such, copying of confidential information due to negligence of the users is strictly avoided.

A typical macro call would look like this:

```
%publish(drveprod=,
         drvepubl=,
         nodir=data_ dat_,
         dirlist=,
         nofile=sd2 sd7 SAS7bdat xpt,
         dircopy=,
         logdir=,
         report=);
```

Parameter	Description
drveprod	Production area – source area
drvepubl	Publish area – target area
nodir	strings corresponding to the first characters of directory names not to be published by this macro
dirlist	List of protocol sub-directories that will be the list of default protocol sub-directories to be published

Parameter	Description
nofile	List of filename extensions to be excluded from publication
dircopy	Path where to find the DIRCOPY.SAS® macro
logdir	Path of the directory where to keep the log
report	Path of the folder containing the report of the publish

Table 2— PUBLISH macro parameters

The Graphical User Interface is generated through, the %WINDOW macro feature. Among the reasons for this choice of technology are:

- Limited set of selections to be made by the end-user.
- Restricted set of end-users (SDE share administrators).
- Robustness.

In Figure 1 the first screen displayed at execution of the PUBLISH macro is shown. It allows the user to select the protocol to be published based on the product name, submission and protocol identifier.

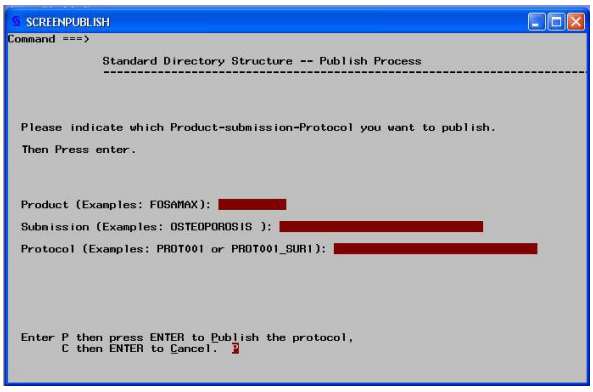


Figure 1: First screen produced by the PUBLISH macro allowing the selection of the protocol to be published.

After validation of these input parameters the second screen (see Figure 2) will display the list of directories available for selective copy in the protocol.

The user can make his choice of the directories to be published. Note that although a standard directory structure (SDS) is applied, the exact structure of the directories present within a protocol may be subject to small variations. As a consequence, the number of directories to be listed is variable. As such, a dynamic generation of screens using the SAS® macro language based on the retrieved directory content for the selected protocol is essential.

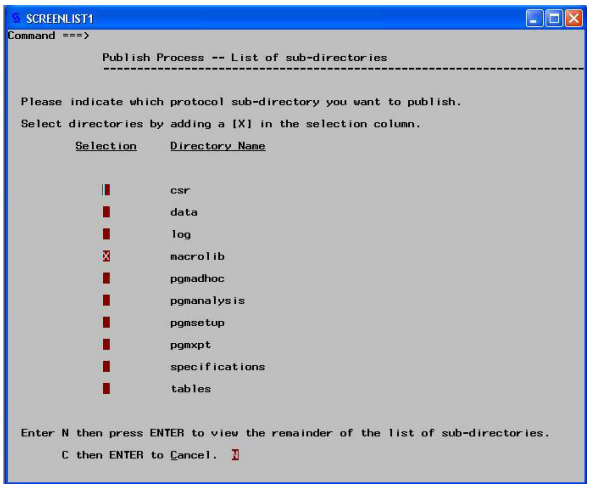


Figure 2: Second screen of the PUBLISH macro allowing selection of the directories containing files to be published

When using the SAS® macro windows (%WINDOW), there is no scroll bar available to handle the possibility that the number of directories to be listed exceeds the available screen size. Therefore, the number of screens to be displayed, in order to list the available source directories, has to be computed on the fly.

DIRCOPY MACRO

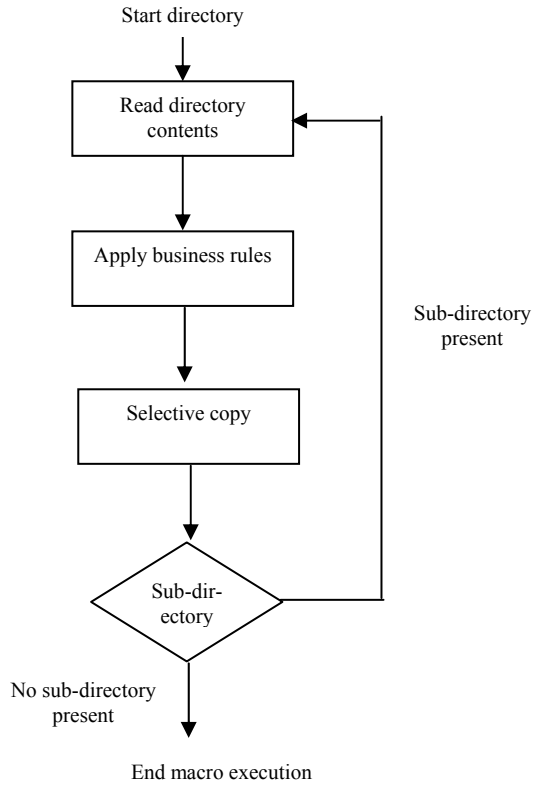
A Selective data copy from a chosen directory is a recursive process. The following tasks need to be applied at each iteration: (i) reading the directory content, (ii) applying the business rules for the selective copy excluding files and/or directories from the list of files to be copied, (iii) executing the actual copy, and (iv) identifying the presence of sub directories and initiating the same iteration on each of those retrieved sub-directories.

The DIRCOPY macro call looks as follows:

```
%dircopy(dirin=,
         dirout=,
         nodir=,
         nofile=)
```

Parameter	Description
dirin	PathName of the source folder.
dirout	PathName of the target folder.
nodir	list of words separated by blanks. sub-folders from the source folder having a name starting by one of these words will not be copied to the target folder.
nofile	list of file extensions separated by blanks. Files with such extensions will not be copied to the target folder.

Table 3— DIRCOPY macro parameters



TECHNICAL DISCUSSION – SAS® MODULEN

Inherent to this macro is the frequent communication between the SAS session and the file system interface of Microsoft Windows: reading contents of directories, creating directories, copying files, etc. In order to optimize these activities the authors made use of Microsoft Windows 32. The SAS® Modulen function allows to call directly a specific routine or module that resides in an external dynamic link library (DLL).

The reasons for using the more complex MODULEN Microsoft Windows API over the typical DOS based commands using pipes are:

- The Win32 API function operates much faster than the DOS Command.
- Using the DOS command forces SAS® to pop-up the DOS window during the execution of the DOS command.

When using the MODULEN function, a call is made to a routine that the SAS® system does not recognize. In order to explain to the SAS® system how to deal with this unknown function, a mapping using an attribute table is required. This attributes table specifies amongst other items (i) what is the DLL to communicate with, (ii) what are the input and output parameters used for the communication with this specific DLL, and (iii) the type of value returned. The SAS® system expects this attribute table to be a flat file referenced using the SAS®CBTBL filename.

An excellent tutorial on the use of the Microsoft API's from SAS® using the MODULEN function can be found in Roper (2001).

In order to illustrate the usage of the MODULEN function, two APIs used in the %dircopy macro are discussed in more details: the CreateDirectoryA API and the CopyFileA API.

CREATEDIRECTORYA

The creation of the directory structure to be copied from the production to the publish area is handled by the CreateDirectoryA API. The routine definition in the attribute table looks as follows:

```
routine CreateDirectoryA
   module=KERNEL32
   minarg=2
   maxarg=2
   stackpop=called
   returns=short
;
arg 1 input char format=$cstr200.;
arg 2 input num format=ib4.;
```

This CreateDirectoryA API is stored in the KERNEL32.DLL and expects two arguments: (i) the full path of the directory to be created, and (ii)- the security attributes to be applied.

The typical call of the CreateDirectoryA routine by the MODULEN function looks like this

```
retrunCode =
modulen('*e',
        'CreateDirectoryA',
        "&directory_to_be_created",
        0);
```

with &directory_to_be_created the full path of the directory to be created. The Returncode will indicate the success (0) or failure (1) of the transaction. Notice that no explicit reference is present in the MODULEN function to this attribute table.

COPYFILEA

The actual selective copy of the files containing non-confidential information is done using the CopyFileA routine. The routine definition in the attribute table looks as follows:

```
Routine CopyFileA
   module=KERNEL32
   minarg=3
   maxarg=3
   stackpop=called
   returns=short
;
arg 1 input char format=$cstr200.;
arg 2 input char format=$cstr200.;
arg 3 input num format=pib4. byvalue;
```

This CopyFileA API is stored in the KERNEL32.DLL

and expects three arguments: (i) the file to be copied, (ii) the destination, and (iii) the option to overwrite the existing file (1) or not (0).

The typical call of the MODULEN function to the CopyFileA routine looks like hereafter:

```
retrunCode =
modulen( '*e',
         'CopyFileA',
         "&file_to_be_copied",
         "&file_to_be_copied_to",
         0);
```

The macro parameter "&file_to_be_copied" contains the full filename of the file to be copied and the macro parameter "&file_to_be_copied_to" the full destination of the file to be copied. The Returncode will indicate the success (1) or failure (0) of the transaction.

FUTURE DEVELOPMENT

From a design perspective, the PUBLISH and DIRCOPY macro are excellent candidates for any type of documented copy of files from a source location to a target location. Potential extensions of the macros could additionally support this kind of task, for example, by enhancing the validation of the copy operation itself.

CONCLUSION

This paper has shown how the PUBLISH and DIRCOPY macros can make a selective copy, in a documented manner, of the non-confidential study files [mainly SAS® programs] to an environment accessible to the members of the SAS® programming community within a clinical biostatistics or scientific programming organization.

As such, previously generated codes are made available to the entire user community of SAS® program developers without disclosure of any confidential information present in the SDE environment. This promotes efficiency and standardization in the development of SAS® programs across clinical studies.

REFERENCES

Barry R. Cohen (2000), "A Submission Development Environment to Support SAS® Programming and Related Activities During Clinical Data Analysis", Proceedings of the Twenty-Third Annual SAS® User Group International Conference, 25, Application Development, paper 28.

Andrea Dynder, Barry R. Cohen, Gary Cunningham (2000), "SAS® Macro For Creating a Standard Directory Structure", Proceedings of the Pharmaceutical Industry SAS Users Group Conference 2000, Application Development, paper 1.

C.A. Roper (2001), 'Accessing and Utilizing the Win32 API From SAS®', Proceedings of the Twenty-Sixth annual SAS Users Group International Conference [SUGI], Systems Architecture section, paper 281-26

F.Coppin and C. Herremans (2003), 'A standard SAS® programming toolbar: A step forward for GPP/SOP compliant SAS® program development', Proceedings of the Annual Conference of the Pharmaceutical industry SAS® Users Group [PharmaSUG], Application Development section, p. 59-63.

ACKNOWLEDGMENTS

The authors thank the members of the Merck Research Laboratories SDE committee for their support as well as Gopal Rajagopal (Merck & Co., Blue Bell MRL office) for his detailed review of the macro.

CONTACT INFORMATION

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

Frederic Coppin
Associate Manager, Scientific Programming
Merck Sharp & Dohme (Europe), Inc.
Clos du Lynx 5,
1200 Brussels [Belgium]
Tel: ++32 27766307
Fax: ++32 27766186
E-mail: frederic_coppin@merck.com
Web: http://www.msd-belgium.be/

Carl Herremans
Statistical Programming Analyst
Merck Sharp & Dohme (Europe), Inc.
Clos du Lynx 5,
1200 Brussels [Belgium]
Tel: ++32 27766309
Fax: ++32 27766186
E-mail: carl_herremans@merck.com
Web: http://www.msd-belgium.be/