Paper PH-02

A GUI-Based Utility Macro for Creating a Version Controlled Project Directory Structure and Copying in Standard Tools and Template Files

Hisham Madi, INC Research, Wilmington, NC

Anna Maeser, MS, INC Research, Wilmington, NC

Michael Zichy, MS, INC Research, Wilmington, NC

ABSTRACT

There are many considerable advantages to standardizing a directory structure for all projects/studies. A well defined project directory structure enhances the organization of study files such as data, SAS programs, output and study documentation. This paper illustrates an approach to creating a standardized directory using a utility macro, a default list of tools and files to be copied into the newly created study directory and an excel spreadsheet which defines the directory structure. Additionally, the process is driven by a SAS pop up window interface that collects user specified options, a management controlled sponsor list/root directory naming convention, and call system commands to create the necessary subdirectories and copy template files, programs, specs, etc. into the newly created study directory. Lastly, since each component of the setup process is version controlled, the macro dynamically selects the most recently approved version of the sponsor, tool and directory/subdirectory lists upon each execution of the setup macro.

INTRODUCTION

A simple way to optimize operational activities is to create and use a clearly defined directory structure for the organization of project files. Regardless of whether a Biometrics group belongs to a small niche company or a large clinical research organization (CRO) providing a variety of services, their commonality is that both typically work with multiple sponsors and each sponsor is likely comprised of a variety of projects or studies. Without a standardized directory structure, one frequently incurs the dilemma of working in a study directory which includes multiple parent directories and many nested subdirectories within each respective folder. The maintainability of the project directory becomes questionable if a user is unable to quickly locate and use programming files, data and project documentation. The problem is all too common when an individual is working with a Biometrics staff across multiple geographic locations, where each office or individual project group utilizes a different directory schema. This guess work is virtually eliminated when the directory structure is standardized across all sponsors and project folders. Additionally, a standardized directory is conducive to project reviews, enabling management to easily ascertain a file or output count, a metric that is useful when quantifying performance or project completion. A structured directory allows any user to easily identify programs which are in development versus completion, quantify outputs finalized and sent to the sponsor or end user, and retroactively confirm project milestones from a standardized archival process.

A directory structure that is reusable across all projects is one that includes a designated repository for each file created, collected and archived within the duration of the project. When a company is able to extensively define a directory structure, a standardized process to create a new project folder can then be utilized by anyone within the organization. The approach illustrated in this paper includes a utility macro *designed to be run on SAS installations using the Windows O/S,* which imports an excel spreadsheet that defines the directory structure. Upon batching the study setup macro, the user follows a series of Graphical User Interfaces (GUI) or SAS pop up windows, collecting user input data such as a sponsor code, protocol number and additional options to further customize the newly created project directory depending on the study construct (i.e. whether to include folders for blinded/unblinded data/output, interim analyses, etc.). Using a management controlled sponsor list and root directory naming convention, certain constraints are forced to ensure the user follows the standardized convention. Lastly, the macro copies standardized template files from a template directory into designated folders within the newly created project directory. A distinguishing feature of this macro is that since each template file is stored in a version controlled folder, the macro selects the most recent version of the file. This allows any user to execute the macro while a Global Standards Team simultaneously develops or modifies any of the particular components of the template tools. Once all of the input data is collected from the user and the newest version of each template file is located, call system commands create the necessary subdirectories and copy template files, programs, specs, etc. into the newly created project directory as illustrated in **Figure 1**.

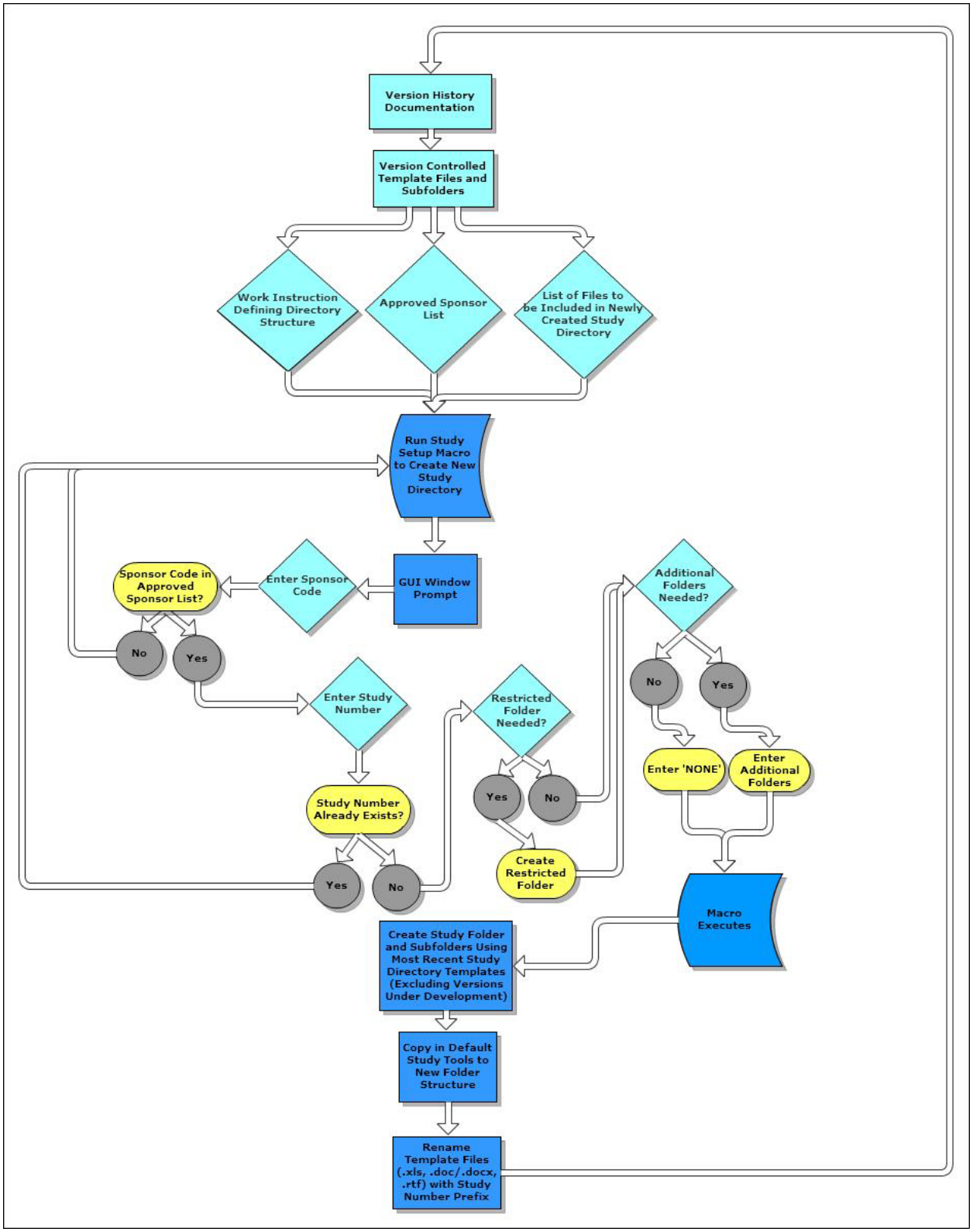


Figure 1. Study Setup Macro Process.

DIRECTORY STRUCTURE

An organized and well defined directory structure is one of the most important aspects of global standardization, providing efficiencies across offices both locally and internationally. In order to define the directory structure, it is beneficial to have a Work Instruction or SOP which lists the levels, whether a folder is required or not, and purpose of all folders which will be contained in the folder structure created by the study setup macro. The purpose section of the Work Instruction or SOP consists of a detail-oriented description of each folder so that users, new and old, can easily understand where study files should reside. This is also a good self-paced training document to orient new associates to the clinical study environment. As time goes on, users may find that additional folder options are useful or new processes dictate a change. The user can request an update to the directory structure documentation and a new version of the directory structure will then be released as shown in **Display 1**. The most recent version of the Work Instruction will always be referenced by the study setup macro and will then be used for all subsequent studies. An example of the contents of this directory structure document is provided in **Display 1**.

Level 1	Level 2	Level 3	Level 4	Level 5	Purpose	Optional (O) or Required (R)?
ABC123					Sponsor level folder from approved sponsor list	R
	012345				Study number nested within the sponsor folder	R
		sasfiles			Overall location for SAS files	R
			1-main		This folder is for main study work	R
				Data	This folder is for main study data	R
			9-restricted		This folder is for restricted study work	O
				Data	This folder is for restricted study data	O

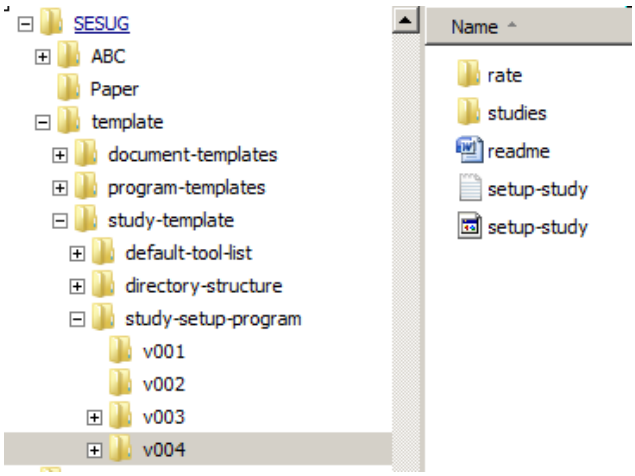
Display 1. Import Spreadsheet which defines the directory structure levels.

The directory structure created by the study setup macro will generally begin with a sponsor-level folder with a project sub-folder nested within. The sub-folders are organized in a way that allows for easy navigation of data, documents, programs, and any other files needed for the project duration. The directory structure is also adaptable in that additional sets of folders can easily be added if needed for instances such as sub-study work or integrations. Restricted folders can also be included if applicable to the study by designating the need for this functionality during the GUI window prompt when running the study setup macro.

The standard folder structure is not the only element that is added with the study setup macro to complete the directory framework. Many standard tools are used within all studies as a result of SOP requirements. Consequently, a list of version controlled default tools is called by the study setup macro so that these can be added to the appropriate standard folder with the study number appended as the prefix of the file name. This is applicable to files such as standard macros, document templates and trackers, and other utility files that are accepted as a standard for all studies universally. The most up-to-date version of all templates and tools will always be added to a new study to ensure consistency and SOP compliance.

TEMPLATES

All files relating to the general set up of a new study directory are located in a template directory. These utility tools, document and programming templates are located in a folder independent of the project parent directory. Each template document and/or program is version controlled, thus a designated subfolder will exist for each new version or iteration of a modified template approved by the standards team as seen in **Display 2**. Ideally, to track modifications approved for each version, the version subfolder should also include a 'readme' word file describing the new update.



Display 2. Each template file is stored in a version controlled folder.

File	Description
default-study-tools.xlsx	Includes a list of files from the template directory to be copied into a newly created study directory
sponsor-list.xlsx	Approved Sponsor List
WI_Directory_structure_27Sep2011.xls	Defines the directory structure
setup-study.sas	Creates the new study directory

Table 1. Template files used when creating a new project directory.

DEFAULT TOOL LIST

The default-study-tools.xlsx spreadsheet includes a list of files from the template directory to be copied into a newly created study directory. This spreadsheet is imported by the setup-study.sas tool to identify which template files to copy and where to allocate the files in the new study directory. All template files with the extension: (.xls), (.doc/docx), (.rtf), will be renamed with the study number as a prefix. Some of the template files to be copied into each new study directory include:

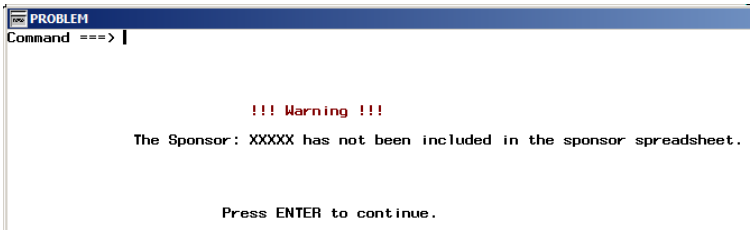
Name of File	Parent Storage Directory of File	Study Directory Location for File [Assumes \ZZXXX\YYYYYY\]	Description of File
setup.sas	program-templates\1-setup	sasfiles\0-macro	Default study setup macro program. This file is required for all project and can undergo project specific customization as necessary once copied.
-create-batch-file.sas	program-templates\5-other\create-batch-file-program	sasfiles\1-main\production\programs\analysis	This program will be placed in each study directory where SAS programs reside and can be used to create .BAT files used for batch processing.
analysis-dataset-metadata-template.xlsx	document-templates\ad-specs	documents\ad-specs	This document should be used for creation of study specific analysis dataset specifications as necessary.

status-tracker.xlsm	document-templates\veri-specs	documents\veri-specs\prog-status\1-main	The purpose of the status tracker is to identify who is responsible for programming, validating, reviewing each mapped dataset, analysis dataset and TFL and what stage each of these processes has reached.
standardheader.sas	program-templates\5-other\program-header	sasfiles\0-utility	This program shall be used as a general reference for the program header for all sas programs.

Table 2. Sample default tool list.

APPROVED SPONSOR LIST

The study setup macro, used to create the directory structure for each new study, references several documents in order to limit user-error and ensure appropriate usage of the tool. One of these documents is the approved sponsor list which is a read-only file that can be modified only be designated users, such as management. This document is read into the study setup macro and compared by the user input when asked for the sponsor code. If the code is not found in this approved sponsor list, the macro will output a warning, as shown below, to inform the user that this sponsor does not exist within the spreadsheet. The user must then double-check the sponsor list to ensure they did not make an error, or they can request an update to the list from the appropriate party. This approved sponsor list limits the creation of extraneous folders or multiple folders for one sponsor.



Display 3. Window prompt for warnings.

The study setup macro will also check the existing folder structure to determine if the approved sponsor code already exists to prevent overwriting existing project work. Similarly, the macro will confirm that a study number does not exist within a particular sponsor folder prior to creating. If the sponsor or study inputs are incorrect, the user will be informed via warnings similar to that in **Display 3**. If warnings do occur, the user will need to re-evaluate their request and re-run the macro until the checks run clean.

SIMPLE BATCH TOOL

Preliminary to illustrating a dynamic GUI approach, the simple batch tool in **Output 1** provides an example of how to manually create a standard directory. The command MKDIR creates each directory folder. Similar to the MKDIR command, the MS-DOS command MD creates the specified directory. The COPY command copies pre-existing files and places them in a specified folder. Additionally, the copied file can be renamed when placing into newly created directory by specifying not only the designated path name, but also including the new name of the file and file extension. Assigning directory attributes is another option by using ATTRIB command, +R to set read only file attributes, -R to remove read only file attributes, +A to archive file attributes and –A to remove the archive file attributes.
Output 1 shows an example of a simple batch tool to manually create a study directory.

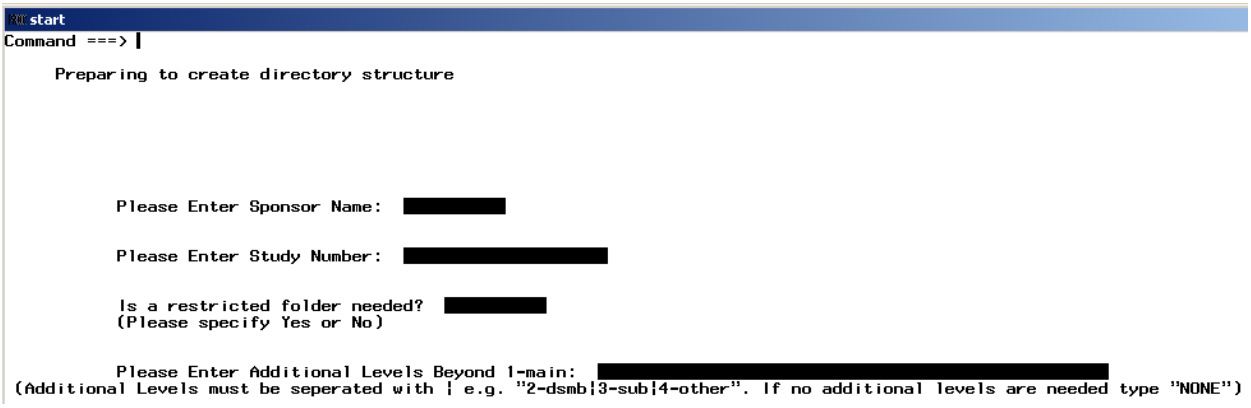
```
mkdir archive
mkdir archive\backup
mkdir archive\delivery

mkdir "documents"
mkdir "documents\correspondence"
```

```
mkdir "documents\specs"
mkdir "documents\sap"
mkdir "documents\tracker"

copy "\SESUG\template\document-templates\specs\v002\*.*"
"documents\specs"
copy "\SESUG\template\document-templates\tracker\v001\*.*"
"documents\tracker"

mkdir sasfiles
mkdir sasfiles\1-main
mkdir sasfiles\1-main\data
mkdir sasfiles\1-main\data\analysis
mkdir sasfiles\1-main\data\mapped
mkdir sasfiles\1-main\data\rawdata

mkdir sasfiles\1-main\macros
copy "\SESUG\template\program-templates\macros\v004\*.sas"  sasfiles\1-main\macros\

mkdir sasfiles\1-main\utility
mkdir sasfiles\1-main\utility\format
mkdir sasfiles\1-main\utility\titles

copy "\SESUG\template\program-templates\program-header\v002\standardheader.sas"
sasfiles\1-main\utility\

mkdir sasfiles\1-main\output
mkdir sasfiles\1-main\output\listing
mkdir sasfiles\1-main\output\table
mkdir sasfiles\1-main\output\figure

mkdir sasfiles\1-main\programs
mkdir sasfiles\1-main\programs\table
mkdir sasfiles\1-main\programs\listing
mkdir sasfiles\1-main\programs\figure

copy "\SESUG\template\program-templates\create-batch-file-program\v001\*.sas"
sasfiles\1-main\programs\table\
copy "\SESUG\template\program-templates\create-batch-file-program\v001\*.sas"
sasfiles\1-main\programs\listing\
copy "\SESUG\template\program-templates\create-batch-file-program\v001\*.sas"
sasfiles\1-main\programs\figure\

mkdir sasfiles\1-main\validation
mkdir sasfiles\1-main\validation\table
mkdir sasfiles\1-main\validation\listing
mkdir sasfiles\1-main\validation\figure

copy "\SESUG\template\program-templates\create-batch-file-program\v001\*.sas"
sasfiles\1-main\Validation\table\
copy "\SESUG\template\program-templates\create-batch-file-program\v001\*.sas"
sasfiles\1-main\Validation\listing\
copy "\SESUG\template\program-templates\create-batch-file-program\v001\*.sas"
sasfiles\1-main\Validation\figure\
```

Output 1. Simple batch tool for creating a study directory

STUDY SETUP MACRO

A new study is setup by any associate batch submitting the read-only setup-study.sas program prompting the interface:



Display 4. Interface for input.

This allows the program to not only create the macro variables that will be used throughout the run but also to establish certain checks to ensure the study is setup correctly.

```
data input;
  window start
  #2 @6 "Preparing to create directory structure"
  #10 @12 'Please Enter Sponsor Name:' +2 sponsor $10. AUTO=YES A=REV_VIDEO REQUIRED=YES
  #13 @12 'Please Enter Study Number:' +2 study $20. A=REV_VIDEO REQUIRED=YES
  #16 @12 'Is a restricted folder needed?' +2 restrict $10. A=REV_VIDEO REQUIRED=YES
  #17 @12 '(Please specify Yes or No)'
  #20 @12 'Please Enter Additional Levels Beyond 1-main:' +2 addlev $50. A=REV_VIDEO REQUIRED=YES
  #21 @2 '(Additional Levels must be seperated with | e.g. "2-dsmb|3-sub|4-other".
         If no additional levels are needed type "NONE")';

  display start;

  call symput("sponsor",strip(upcase(sponsor)));
  call symput("study",strip(study));
  call symput("restrict",upcase(strip(restrict)));
  call symput("addlev",strip(addlev));
  output;
  stop;
run;
```

Output 2. Code to create input interface.

The use of REQUIRED=YES forces the fields to be populated, and checks are put in place to ensure all the macro variables are properly defined in the next step of the program. For example, if the restricted folder response is not populated with “Yes” or “No” then the program terminates and prints to the log that the field was not correctly populated.

```
 data _null_;
  set input;
  if missing(restrict) or upcase(restrict) ^in("YES" "NO") then do;
  call symput("FAIL",1);
  window start
  #2 @6 "Please mark if a restricted folder is needed";
  display start;
  stop;
  end;
 run;

 %if &FAIL.=1 %then %do; %ABORT; %end;
```

Output 3. Sample input check.

Once all the fields have proper entries, the program next checks to ensure that the sponsor name is one that is approved by the sponsor list.

```
%let PASS = 0;
%macro dummy;

libname spons "R:\Biostat\sponsor-list.xlsx"
      command_timeout=60
      access=readonly
      scan_textsize=yes
      mixed=no;

data sponsor;
  set spons.'Sponsor Abbreviation$'n;
  if Abbreviation="&sponsor." then do;
    call symput("PASS",1);
  end;
  run;
run;

  %if &PASS.=0 %then %do;

    %window problem color=white
            #5 @28 '!!! Warning !!!' attr=highlight
               color=red
            #7 @15
               "The Sponsor: &sponsor. has not been included in the sponsor spreadsheet."
            #12 @25 'Press ENTER to continue.';

   %display problem;

    %ABORT;

  %end;
%mend;
%dummy;
```

Output 4. Check to ensure sponsor name is approved.

This restricts an associate from producing a folder for a study that has not been approved and allows management to have some control over the folder creation while still allowing any associate rights to create a folder.

Next, a pop-up box is presented to allow the associate one last look to check the entries they submitted before the program commences. When the information is verified, the program does a final check to ensure that the study

number does not already exist (i.e. has not already been created previously).

```
%LET sponfind = N;
FILENAME dirlist pipe "dir &writepath.\biostat\&sponsor. /b/d/on";
DATA folders;
 INFILE dirlist;
 INPUT VAR:$200.;
  if INDEX(VAR,'.') = 0; ** remove all extraneous files having extension;
  if TRIM(LEFT(UPCASE("&study."))) = TRIM(LEFT(UPCASE(var)));

  CALL SYMPUT('sponfind',"Y");
RUN;

%macro dummy;
%IF &sponfind. = Y %THEN %DO;  ** precautionary termination due to study number already existing;
  %PUT %UPCASE(erro)R: Study number found - please correct value of "study"  variable;
    %window problem color=white
              #5 @28 '!!! Warning !!!' attr=highlight
                 color=red
              #7 @15
                 "The Study Number chosen already exists"
              #12 @25 'Press ENTER to continue.';

    %display problem;
  %ABORT;
%END;
%mend;
%dummy;
```

Output 5. Check to ensure study number is not present already.

It should be noted that the decision to restrict study numbers that are already present was made after lengthy discussion to the pros/cons of such a check. The program itself will never delete any folders; it will only create folders if they are not currently present. There was discussion on the utility of being able to rerun the study setup on an already created folder (for example if it was determined that a restricted folder was needed after the study had already been setup) which would allow automation of additional standard folders. Because the study setup program also copies in templates of files, it was decided that the risk of overwriting an existing work file was not worth the risk in our environment. There certainly exists merit in the possibility of splitting the two processes to allow for folder structure reconciliation mid-study.

If all proper information has been entered and verified, the program then uses a macro to automatically load the latest version of the standard directory structure.

```
%MACRO LATEST_VERSION(dirz=);

    %global version;

    filename dir_name pipe "dir &dirz /A:D /B";

    data dir_name(where = (upcase(dir_name) ne "NOTINUSE"));
      length dir_name $30.;
      infile dir_name;
      input dir_name $;
      dir_name = upcase(dir_name);
    run;

    proc sort data = dir_name;
      by descending dir_name;
    run;
    data _null_;
      set dir_name;
      if _n_=1 then call symput("version", trim(dir_name));
    run;

    %put &version.;
%MEND LATEST_VERSION;
%LATEST_VERSION(dirz=%bquote(&writepath.\1-templates\1-study-template\directory-structure\));
```

Output 6. Macro to find latest version..

The macro loads the latest approved version of the standard folder structure excel sheet and uses that as a template to write out the folders onto the general programming area using a slightly more complex process than from the simple batch tool code from **Output 1**.

Afterwards, the program then uses the same macro to grab the latest version of the default tool list. The program uses the locations marked in the default tool list to designate where the tools should be copied. There are checks to ensure that a warning is issued if a location listed by the default tool list does not exist in the current standard folder structure. Otherwise, the tools are copied into the folder structure and renamed with the associated study number defined from the original prompt.

All of the aforementioned coding is designed to produce the standard “1-main” area that all studies will be expected to have. The efficiency of the study setup program comes from the final section where any additional subfolder requests are created. Rather than cycling through the entire process again, call commands are utilized to create renamed copies of the main folder.

```
%macro dummy;
%if %quote(&addlev.) ^= NONE %then %do;  ** if additional folders are requested, copy from "main";
 data extralev;
   length command $1000.;
   if upcase("&addlev.") ne "NONE" then do;
    levcnt = count("&addlev.","|") + 1;
     do i = 1 to levcnt;
      command= "xcopy " || "&writepath.\biostat\&sponsor.\&study.\sasfiles\&runname. /s /e
&writepath.\biostat\&sponsor.\&study.\sasfiles\"||
      scan("&addlev.",i,"|")||"\";
      output;

       command= "md " || "&writepath.\biostat\&sponsor.\&study.\documents\veri-specs\prog-status\"
       ||scan("&addlev.",i,"|");
       output;

       command = "copy " || "&writepath.\biostat\&sponsor.\&study.\documents\veri-specs\prog-status\1-
main\&&study.-status-tracker.xlsm
                        &writepath.\biostat\&sponsor.\&study.\documents\veri-specs\prog-status\"
       ||scan("&addlev.",i,"|");
       output;

       command = "copy " || "&writepath.\biostat\&sponsor.\&study.\documents\veri-specs\prog-status\1-
main\&&study.-qc-document-tracker-template-v1.xls
                        &writepath.\biostat\&sponsor.\&study.\documents\veri-specs\prog-status\"
       ||scan("&addlev.",i,"|");
       output;

     end;
   end;
 run;

  data _null_;
    set extralev;
    call system( command );
  run;
%end;
%mend;
%dummy;
```

Output 7. Call commands to mirror completed folder structure.

This allows each of the additional required levels to be mirrored off the main directory, including all subfolders and default tools.

The end result is that any level associate has the power to create a folder for a study while management still has control over which studies are created and a standards team still has control over the overall structure and tools of the folder. Not only does this guarantee that all folders are uniform, but also that the latest approved tools are accessible immediately to all new studies.

CONCLUSION

Since projects within clinical research vary in therapeutic areas and study design, process variation is inevitable. In spite of the fact, an organization can act upon the variability in work environments by creating and using a standardized directory, enabling any associate to intuitively recognize the placement of project files. Continuous process improvement for all components of standardized process is necessary. A tactical approach encompasses an environment conducive to version control and allows constant development and enhancement to pre-existing tools. These tools should be maintained in a central repository, otherwise version control is compromised when a user is forced to make assumptions of which project to copy the most recently adapted tool. This central repository entices a close collaboration between a standards team that maintains and documents these triage tools with all associates who actively use these tools. The programming code to create and maintain directory folders is readily accessible with SAS call system commands.

REFERENCES

Bandi,Ranganath and Kunduru, Harini. PharmaSUG2011 - Paper AD06. Better Ways to Speak to Your System Using SAS:Automate Routine Tasks by using X, SYSTASK & FILENAME." *Proceedings of the PharmaSUG 2011 Conference*

Available at http://www.pharmasug.org/proceedings/2011/AD/PharmaSUG-2011-AD06.pdf

ACKNOWLEDGMENTS

A special thank you to Edward Bakewell and Matthew Psioda, who have provided a conglomeration of leadership, continued mentorship and support.

CONTACT INFORMATION

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

Hisham Madi
INC Research
1013 Ashes Dr.
Wilmington, NC 28405
Work Phone: 910-509-4893
E-mail: hishamkamalmadi@gmail.com

Anna Maeser, MS
INC Research
1013 Ashes Dr.
Wilmington, NC 28405
Work Phone: 303-444-4035
E-mail: maeser.anna@gmail.com

Michael Zichy, MS
INC Research
1013 Ashes Dr.
Wilmington, NC 28405
Work Phone: 910-509-4714
E-mail: michael.zichy@INCresearch.com