PharmaSUG 2022 - Paper AD-084

A SAS Macro to Aid Assembly Review of M5 Module in Submission

Jeff Xia, Merck & Co., Inc., Rahway, NJ, USA

ABSTRACT

Assembly Review is the final check of the published electronic regulatory submission documents performed by the submission’s content authors/owners prior to release to health authorities and/or subsidiaries. Normally the programming group is the content owner of all data components, which includes SAS datasets in XPT format, annotated case report form, define documents as well as the reviewer’s guide, etc. in m5 module for all submitted studies. Therefore, the programming group is responsible to perform a thorough assembly review to ensure correctness and completeness of each data component in m5 module. Traditionally this was done by manually comparing the list of file names in each subfolder in m5 module in a published sequence against the corresponding folder in the final drop off location, and catching the difference, if any, to avoid any mistakes/overlooks in the regulatory publishing process.

This paper introduces a SAS macro that compares the list of file names in two folders (i.e., folder “New” and folder “Old”) including their subfolders, and flags any files not matching, i.e., a file exists in the folder “New” but not in the folder “Old”, and vice versa. Additionally, if a file exists in both folders “New” and “Old”, then it compares the file time stamp and flags if there is any difference. Lastly, this macro generates a report to list the name of each file and details of comparison by folder/subfolder.

There are multiple subfolders in the m5 module in an eCTD for a single study, and it is very common to include multiple studies in each submission. Manually browsing the file list in each subfolder in the m5 module is very time consuming and error prone. This paper provides an automatic method to aid the assembly review of the m5 module in a submission that greatly improves the efficiency and accuracy.

INTRODUCTION

Assembly review is defined as the final check of the published electronic regulatory submission documents performed by the submission’s content authors/owners prior to release to health authorities and/or subsidiaries. This check is the contributor’s last opportunity to ensure the complete, accurate inclusion and organization of all submission content files as well as metadata in the published submission sequence. Since assembly review is such a key step in the entire submission process, it deserves to allocate enough time and attention to identify any errors that could have been avoided during the publishing process.

Normally the programming group is the content owner of the data components in the m5 module in eCTD for all submitted studies in each filling, which includes SAS datasets in XPT format, annotated Case Report Form (acrf.pdf), define documents and associated supporting files (i.e., stylesheet), reviewers’ guides in PDF format and SAS programs in text format that are used to create all ADaM datasets and generate tables and figures associated with primary and second efficacy analyses. These datasets and companion files should be incorporated into eCTD in a way that meets the requirement specified by the agency in the Guidance to Industry, i.e., “Providing Regulatory Submission in Electronic Format: Certain Human Pharmaceutical Product Applications and Related Submissions Using the Electronic Common Technical Document Specifications”.

The programming team should ensure the data components are accurate and complete before delivering to the publishing group for assembly into the m5 module in eCTD. Once the submission components have been assembled and validated on the publishing side, the publisher notifies the programming group to perform the assembly review to make sure the assembled data components in m5 module are exactly the same as the ones that programming team delivered, nothing more, nothing less.

Traditionally this was done by manually comparing the list of file names in each folder in the m5 module in a published sequence with the corresponding folder in the final drop off location. Obviously, this eye-browsing task requires a lot of attention to catch the difference if any. Very often it is error prone, and any discrepancy could be overlooked. This paper introduces a SAS macro that compares the list of file names in two folders, i.e., folder “New” and folder “Old, and all the files in their subfolders are included in the comparison. This macro flags any files that are not matching. For example, a file that exists in the folder “New” but not in the folder “Old”, and vice versa. Additionally, if a file exists in both the “New” and “Old” folder, then it compares the file timestamp and flags if there is any difference. Normally when the contents of a file changes, the file timestamp changes as well.

STEP 1: RETRIEVE LIST OF FILE NAMES IN FOLDERS

The first step is to obtain the file names in a folder including all the subfolders. A macro is designed to automatically retrieve the file list in a folder, then a recursive call was used inside the macro itself to retrieve the file list in each subfolder.

```
%macro listfiles(dir);
    %local filrf rc did fnm i fid;

    %* Assigns a fileref to the directory and opens the directory *;
    %let rc=%sysfunc(filename(filrf,&dir));
    %let did=%sysfunc(dopen(&filrf));

    %* Loops through entire directory *;
    %do i=1 %to %sysfunc(dnum(&did));
        %* Retrieve name of each file *;
        %let fnm=%qsysfunc(dread(&did,&i));
        %let fid=%qsysfunc(mopen(&did,&fnm));

        %if %qscan(&fnm,2,.) ne %str() and &fid ne 0 %then %do;
            fileloc="%qsysfunc(finfo(&fid,Filename))";
            modifyDT="%qsysfunc(finfo(&fid,Last Modified))";
            output;
        %end;
        %* If directory name call macro again *;
        %else %if %qscan(&fnm,2,.) = %str() %then %do;
            subfolder = 1;
            %listfiles(&dir&mslash%unquote(&fnm));
        %end;
    %end;

    %* Closes the directory and clear the fileref *;
    %let rc=%sysfunc(dclose(&did));
    %let rc=%sysfunc(filename(filrf));
%mend listfiles;
```

Then call the macro %listfiles to obtain the list of files in the folder of “New”, see below for the code snippet for obtaining the list of file names in the folder of “&newdir”.

```
data files1; /*process new folders*/
    length fileloc ModifyDT $2000 root $1000;
    root = strip(lowcase("&newdir"));
    %listfiles(&newdir);
run;

data files1;
    set files1;
    length comppath $2000 ;
    len = length(root) + 2;
    comppath = lowcase(substr(fileloc, len));
    if index(comppath, "&mslash") > 0 then subfolder = 1;
    /*the value of macro variable mslash is / in Windows environment;*/
    /*the value of macro variable mslash is \ in Unix/Linux environment;*/
    else subfolder = 0;
run;

proc sort data = files1(keep = comppath ModifyDT);
    by comppath;
run;
```

And do the same to obtain the list of file names in the folder of “Old” and save the result in the dataset “files2”.

STEP 2: PERFORM THE COMPARSION

Merge the dataset “files1” and “files2” based on the merging key “comppath”. The variable “status” was created to record 4 possible outcomes. Ideally, we should see the first outcome for all files in an assembly review, which means all files are matching between the published submission sequence by the publisher and the files in the drop off location by the programming team. However, if any outcome of 2, 3, or 4 appears, it is alarming and should alert the reviewers’ attention. Further investigation is needed to ensure a complete and accurate submission.

1. A file exists in both folders and the time stamp is matching
2. A file exists in the New folder but not in Old
3. A file exists in Old folder but not in New
4. A file exists in both folders, but the time stamp is not matching

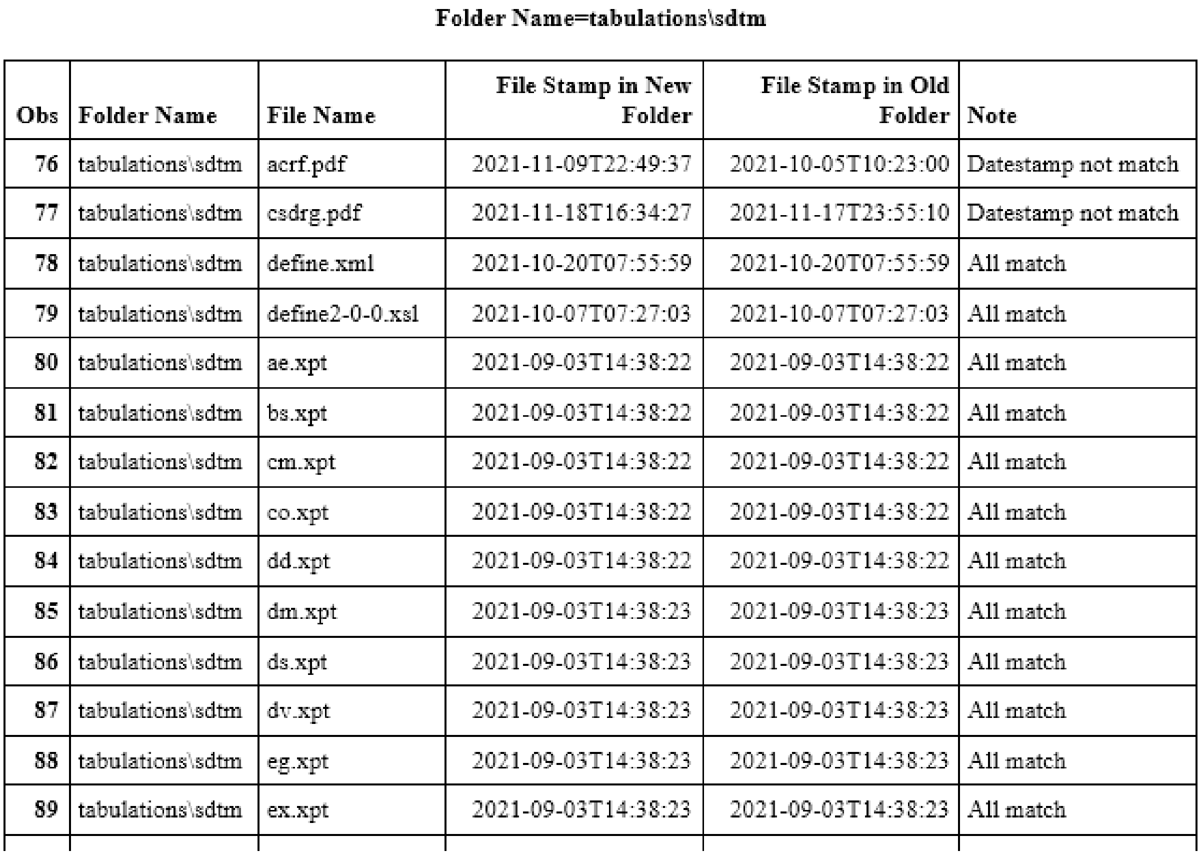
```
data final;
    merge files1(in = a ) files2(in = b keep = comppath ModifyDT2);
    by comppath;
    length status $100;
    if a and not b then status = 'In New folder but not in Old';
    else if b and not a then status = 'In Old folder but not in New';
    else do;
        if ModifyDT ne ModifyDT2 then status = 'Datestamp not match';
        else status = 'All match';
    end;
    ModifyDT_n = input(strip(ModifyDT), datetime19.);
    ModifyDT2_n = input(strip(ModifyDT2), datetime19.);
run;
```

The outcome can be generated in any format per reviewers’ preference, such as RTF, PDF, or HTML, And the order of records can be arranged in a certain way to draw reviewers’ attention. For example, the following code snippet will display the name of PDF file first, and then xml and its stylesheet, and then Excel spreadsheet if any, etc.

```
if index(comppath, 'pdf')           then  order = 1;
    else if index(comppath, 'xml')  then  order = 2;
    else if index(comppath, 'xsl')  then  order = 3;
    else if index(comppath, 'xls')  or
            index(comppath, 'xlxs') then  order = 4;
    else if index(comppath, 'txt')  or
```

```
            index(comppath, 'csv')  then  order = 5;
    else if index(comppath, 'xpt')  then  order = 6;
    else                                  order = 7;
```

See below for a sample output of the macro:



Folder Name=tabulations\sdtm

Obs	Folder Name	File Name	File Stamp in New Folder	File Stamp in Old Folder	Note
76	tabulations\sdtm	acrf.pdf	2021-11-09T22:49:37	2021-10-05T10:23:00	Datestamp not match
77	tabulations\sdtm	csdrg.pdf	2021-11-18T16:34:27	2021-11-17T23:55:10	Datestamp not match
78	tabulations\sdtm	define.xml	2021-10-20T07:55:59	2021-10-20T07:55:59	All match
79	tabulations\sdtm	define2-0-0.xsl	2021-10-07T07:27:03	2021-10-07T07:27:03	All match
80	tabulations\sdtm	ae.xpt	2021-09-03T14:38:22	2021-09-03T14:38:22	All match
81	tabulations\sdtm	bs.xpt	2021-09-03T14:38:22	2021-09-03T14:38:22	All match
82	tabulations\sdtm	cm.xpt	2021-09-03T14:38:22	2021-09-03T14:38:22	All match
83	tabulations\sdtm	co.xpt	2021-09-03T14:38:22	2021-09-03T14:38:22	All match
84	tabulations\sdtm	dd.xpt	2021-09-03T14:38:22	2021-09-03T14:38:22	All match
85	tabulations\sdtm	dm.xpt	2021-09-03T14:38:23	2021-09-03T14:38:23	All match
86	tabulations\sdtm	ds.xpt	2021-09-03T14:38:23	2021-09-03T14:38:23	All match
87	tabulations\sdtm	dv.xpt	2021-09-03T14:38:23	2021-09-03T14:38:23	All match
88	tabulations\sdtm	eg.xpt	2021-09-03T14:38:23	2021-09-03T14:38:23	All match
89	tabulations\sdtm	ex.xpt	2021-09-03T14:38:23	2021-09-03T14:38:23	All match

CONCLUSION

This paper presents a macro that can be used to programmatically to retrieve the list of files between two folders and their subfolders and produce a report to display any discrepancy. This macro is very helpful in the assembly review process for the programming team to ensure the data components in m5 module in eCTD are accurate and complete.

ACKNOWLEDGMENTS

The author would like to thank Mary Varughese and Lisa Pyle for their great support and valuable input into this paper.

CONTACT INFORMATION

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

Name: Jeff Xia
Enterprise: Merck & Co., Inc.
Address: 126 East Lincoln Avenue,
City, State ZIP: Rahway, NJ 07065-4607
Work Phone: 732-594-6439

E-mail: jeff.xia@merck.com
Web: www.merck.com