Automating SAS® Program Header Updates with Macros

Kexin Guan is a Statistical Programmer at Merck, where she has been part of the Oncology Early Development group since December 2022. She holds a Master's degree in Biostatistics and is proficient in SAS and R. Kexin is passionate about leveraging programming to enhance efficiency and compliance in pharmaceutical programming.

MERCK

Automating SAS® Program Header Updates with Macros

PharmaSUG SDE NC 2025

Kexin Guan

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

BACKGROUND & MOTIVATIONS

MERCK

Program Header: Background

- A standardized block of code at the top of every SAS® file containing key details
  - Usually includes purpose, author, inputs, macros, outputs, revision history, etc.
- Ensures audit trails & program traceability
- Meets regulatory specifications (e.g. FDA Study Data Conformance Guide)

```
/*-------------------------------------------------------------------------------
   Study:

   Program:

   Keywords:

   Function:

   Author:

   Version Date:

   SAS Version:

   Platform:

   Input Data:

   Macros Called:

   Program Output:

   Program Log:

   Macro Parameters:

   Program Flow:

   Limitations/Cautions:

   Comments:

   Revisions:
   Name                    Date              Description
-------------------------------------------------------------------------------*/
```

MERCK

Challenges in Manual Updates

- Complexity increases → manual header updates become tedious
- Time-consuming & error-prone in large-scale studies
- Hard to track many inputs, outputs & macro calls across files
- Iterative changes (new datasets, updated outputs) amplify inconsistencies and omissions

THE MACRO SOLUTION
%generate_program_header

MERCK

Macro Overview: % ***generate_program_header***

- Auto-inserts or updates headers
- Captures: inputs, macros, outputs, steps, program flows
- Retains or purges old revision entries
- Processes single file or entire directory

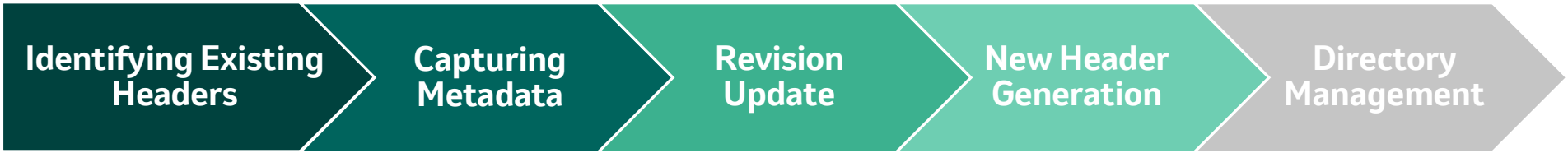
The Macro Inputs

- File_or_directory: Path to the SAS file or directory.
- Study: Name of the associated study.
- Update_author: Author name to include. Defaults to the existing author if left blank.
- Revision_name: Name of the person revising the header.
- Delete_revisions: Option to delete previous revisions (Y/N).

- An Example macro call:

```
%generate_program_header (
                          file_or_directory = /path/to/file.sas
                          ,study            = Study123
                          ,update_author    = Jane Doe
                          ,revision_name    = John Smith
                          ,delete_revisions = Y
                          );
```

Macro Program Flow: Overview



Identifying Existing Headers

- Scan for /*-- and --*/ delimiters (can be changed)
- No header found → create new header block
- Header exists → extract current info & update
- ADaM files assumed to always have a header → only update certain parts

Metadata Capture

Input datasets:

- Detect via libref patterns (e.g. LPTDA./LPTSS.)

Output datasets:

- List .lst & .log references, or other forms

Program Flow:

- Pieces of comments within the code that mark the steps (common in ADaM program)
- Detect “STEP + Number:” comments and present them in order

Metadata Capture

Macros Called:

- Only Include external macros

- Exclude locally defined macros: searching for pairs of “%macro” and “%mend” in the code.

- Exclude SAS built-in macros: Regex to capture most commonly used ones

```
prxmatch('/%str|%nrstr|%bquote|%nrbquote|%quote|%nrquote|%superq|%l
et|%put|%macro|%mend|%global|%local|%eval|%sysevalf|%superq|%qscan|
%scan|%substr|%qsubstr|%length|%index|%verify|%if|%then|%else|%do|%
end|%until|%while|%goto|%window|%display|%abort|%include|%inc|%inpu
t|%list|%run|%return|%qsysfunc|%unquote|%upcase|%lowcase|%trim|%lef
t|%qupcase|%qlowcase|%qtrim|%qleft|%sys(?!\w)|%sym(?!\w)/', line)
```

Revision Update & Header Generation

Revision update:

- Optionally delete old entries
- Append new line: date, reviser, change note

Header generation:

- Pull in all metadata (including SAS built-in macro variables such as &sysdate9, &sysver, &sysscpl)
- Insert standardized block at top

At last, loop through all SAS files under the directory if applicable.

IMPLEMENTATION & CONCLUSION

MERCK

Example 1: Creating New Header for Macro-Call Program

- Code without pre-existing header
- The info we want to capture in code:
  - Input datasets
  - Macros called
  - Program outputs & log filenames
- We want to exclude SAS built-in macros

```
%generate_program_header(
    file_or_directory = /studypath/test01.sas
    ,study            = MKXXXX PROTXXX
    ,update_author    = Jane Doe
    ,revision_name    = Kexin
    ,delete_revisions = N
);
```

```
/*Test01*/

%rtfsymbl;

proc printto  log ="%sysfunc(pathname(fptolg,f))/s0output01.log"
              print ="%sysfunc(pathname(fptol,f))/s0output01.lst"
              new;
run;

%test0macro1(
    ,population_from = LPTDA.ADSL
    ,population_where =%str(&XXX_FILTER)
    ,observation_from =LPTDA.ADAE
    ,observation_where = %str(XXXFL="Y")
    ,therapy_des_var =VAR1
    ,therapy_cd_var =VAR1N
    ,subtitle =%str(XXXXX)
    ,end_notes =%str(XXXXX)
) ;

proc printto ;run;

%wrapup;

proc printto  log ="%sysfunc(pathname(fptolg,f))/s0output02.log"
              print ="%sysfunc(pathname(fptol,f))/s0output02.lst"
              new;
run;

%test0macro2(
    ,population_from = LPTSS.DM
    ,population_where =%str(&XXX_FILTER)
    ,observation_from =LPTSS.AE
    ,observation_where = %str(XXXFL="Y")
    ,therapy_des_var =VAR1
    ,therapy_cd_var =VAR1N
    ,subtitle =%str(XXXXX)
    ,end_notes =%str(XXXXX)
) ;

proc printto;run;

%wrapup;

%logchecker (input_fileref=fptolg, filelist=s0output0*,
output_filename=s0output0_logchecker);
```

Example 1: Creating New Header for Macro-Call Program

- Original code remains untouched; macro only prepends header
- Revision section appended with current author & date

```
/*-------------------------------------------------------------------------------
Template ID:          001

Study:                MKXXXX-PROTXXX

Program:              /studypath/test01.sas

Keywords:

Function:

Author:               Jane Doe

Version Date:         09JAN2025

SAS Version:          9.4

Platform:             Linux

Input Data:           LPTDA.ADAE, LPTDA.ADSL, LPTSS.DM, LPTSS.AE

Macros Called:        %logchecker, %rtfsymbl, %test0macro1, %test0macro2, %wrapup

Program Output:       s0output01.lst, s0output02.lst

Program Log:          s0output01.log, s0output02.log

Macro Parameters:

Program Flow:

Limitations/Cautions:

Comments:

Revisions:
Name                      Date                 Description
Kexin                      09JAN2025       Updated program header.
-------------------------------------------------------------------------------*/
```

Example 2: Updating an Existing ADSL Header

- This ADSL program header is outdated
- For ADaM programs, make no updates on outputs and logs
- Expect to capture macros called, program flow, and delete previous revision history

```
%generate_program_header(
   file_or_directory = /studypath/ADSL.sas
  ,study             = MKXXXX PROTXXX
  ,update_author     =
  ,revision_name     = Kexin
  ,delete_revisions  = Y
);
```

```
Macros Called:      %add0attribute

Program Output:     lptda.adsl

Program Log:        fptolg(adsl.log)

Macro Parameters:   None

Program Flow:
  Step 01: Get data from SDTM
  Step 02: Simple conversions
  Step 03: Derive population flag RANDFL and Randomization Date RANDDT
  Step 04: Add EOSDT, DCSREAS, EOTSTT, DCTREAS
  Step 05: Derive the Treatment Start dates and End Dates
  Step 06: Add baseline variable from VS
  Step 07: Derive subject ENRLDT / ENRLFL
  Step 08: Derive the cause for Death using DD domain (If available)
  Step 09: Add ECOG performance status at baseline from QS
  Step 10: Derive Last Known Alive Date variables
  Step 11: Add death related variable from DM and UNCUT Death Date Variable
  Step 12: Merge data together
  Step 13: call %add0attribute to add variable attributes
  Step 14: Create Metadata dataset: asr0fram
  Step 15: Data clean up

Limitations/Cautions:
   1.  Make sure using read0adamspec0xls2ds.sas macro to generate lptmt.adsl0spec from current ADaM specification spreadsheet.
   2.  This program is a template. Users need to modify it accordingly based on your study requirement.

Comments:

Revisions:
Name                 Date              Description
Editor1              25OCT2022         Updated for interim analysis, add standard variables
Editor1              25MAY2023         Updated for IA05 on categories from CC
Editor1              05JUN2023         Updated EOSDT
Editor1              06JUN2023         Added IA5FL
Editor2              19Jul2023         Added IA5INVFL & IA5IRCFL in place of IA5FL based on the spec update;
Editor2              12Jan2024         Updated braf derivarion and removed epoch=screening ;
Editor2              16Jan2024         Added latest arms format values to trt01p/a;
Editor2              13Feb2024         Added CLNPTH & CUCLPTS based on study requirement;
-------------------------------------------------------------------------------*/
```

MERCK

Example 2: Updating an Existing ADSL Header

- Reflects accurate program flows and macro called
- Refreshes the revision history upon user request

```
Macros Called:       %add0attribute, %build0sdtmplus, %getdata, %logchecker, %mrace, %nctx, %read0adamspec0xls2ds

Program Output:     lptda.adsl

Program Log:        fptolg(adsl.log)

Macro Parameters:    None

Program Flow:
 Step 01: Get data from SDTM
 Step 02: Simple conversions
 Step 03: Derive population flag RANDFL and RANDDT
 Step 04: Add EOSDT, DCSREAS, EOTSTT, DCTREAS,
 Step 05: Derive the Treatment Start dates and End Dates
 Step 06: Add baseline variable from VS
 Step 07: Derive subject ENRLDT / ENRLFL
 Step 08: Derive the cause for Death using DD domain (If available)
 Step 09: Add ECOG performance status at baseline from QS
 Step 10: Derive Last Known Alive Date variables          -*;
 Step 11: Add death related variable from DM and UNCUT Death Date Variable
 Step 12: PD-L1 related: CPS/CPSN, PDL1/PDL1N
 Step 13: Derive STRATUM/STRATUMN PRADJPD1/PRADJP1N LDHBL/LDHBLN
 Step 14: Add Overall cancer stage OSTAGE & OSTAGEN from CC
 Step 15: Add  BRAFM  - BRAF Mutation from PF
 Step 16: Add BMETS - Brain Metastasis from MH
 Step 16.1: Add Sum of Target Lesion at baseline from TR
 Step 16.2: Add STRATAR/STRATARN
 Step 16.3 Derive Tumor stage and Tumor Stage Number
 Step 16.4 Derive Nodal Involvement
 Step 17 Derive DLTPOPFL
 Step 18: call %add0attribute to add variable attributes
 Step 19: Create Metadata dataset: asr0fram
 Step 20: Data clean up

Limitations/Cautions:
1.  Make sure using read0adamspec0xls2ds.sas macro to generate lptmt.adsl0spec from current ADaM specification spreadsheet.
2.  This program is a template. Users need to modify it accordingly based on your study requirement.

Comments:

Revisions:
Name                    Date                 Description
Kexin                   09JAN2025            Updated program header.
------------------------------------------------------------------------------*/
```

Conclusion & Next Steps

- **Saves time & effort:** streamlines header maintenance for clearer, more traceable SAS programs; Automates hours of manual header edits and QC
- **Versatile:** works for ADaM & TLF call programs, single files or whole directories
- **Best practice:** use for major updates & final QC, avoid during early development; Make sure to rerun outputs/validation after development header update
- **Limitations:** rely on standard existing header layouts, may stumble on unusual built-in macros
- **Next steps:** improve error handling, broaden regex/customization options

Thank you

MERCK

Contact Information

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

Kexin Guan
Merck & Co., Inc., Rahway, NJ, USA
kexin.guan@merck.com

MERCK