Macro version management done right in SAS LSAF (and other SAS environments)

Caro Sluijter

Acknowledgement

Thank you to my two colleagues who are the creators of
“macro version management according to OCS Life Sciences”:

- Radoslaw Pszczolkowski, OCS Life Sciences
- Kai Wanke, OCS Life Sciences

What do we (programmers) want:

1. We want to work with macros
2. We want to not always need to copy a standard macro into every study folder
3. We want to not have multiple same versions of the macro circulating
4. We want to store standard macros in general location
5. We want to use the macro from general location
6. We want to not (accidentally) alter the standard macros
7. We want to update these standard macros when needed
8. We want to be able to view older versions of the macro
9. We want to not impact older studies with updates on standard macros
10. We want to not need to update programs when a macro is updated

- This presentation will provide a solution
  be able to give programmers all of these functions!

Standard Macro's

- Programmers distinguish different macros:

- General macro
  - Macro that can/will be used in multiple instances
    - Multiple programs
    - Multiple studies
  - Present on a general location
    - Macro library

- Study macro
  - Macro that is created/adapted for a specific study
  - Present in study specific macro folder

Options off macro version management

- Traditional macro version management
- LSAF file versioning
- OCS Solution to macro version management

Traditional macro version management

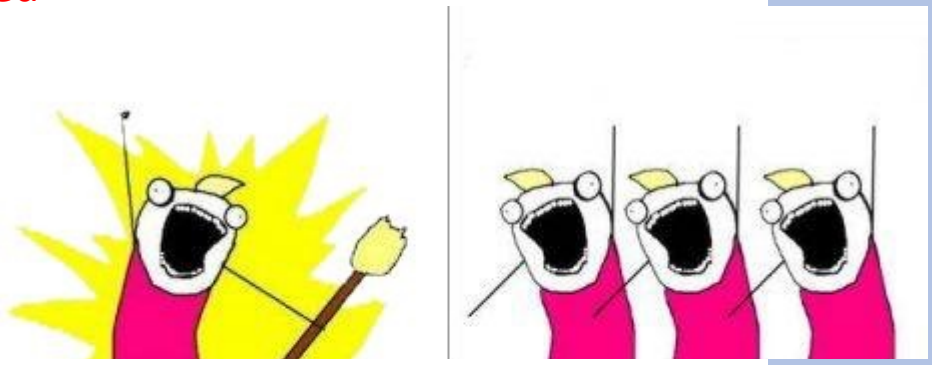
- Store all general macros into one location (read only)
- Call macros from this location

- Single Macro:
  - Name of macro is updated from Macro_V1 to Macro_V2
  - All programs using this macro need to be updated to call correct version
  - All programs need to be revalidated as change in macro name changed the program

- Suite of Macros
  - Sub macro name is updated from Smacro_V1 to Smacro_V2
  - Main macro is updated from Macro_V1 to Macro_V1_1
  - All programs using this macro need to be updated to call correct version
  - All programs need to be revalidated as change in macro name changed the program

What do we (programmers) want:

1. We want to work with macros
2. We want to not always need to copy a standard macro into every study folder
3. We want to not have multiple same versions of the macro circulating
4. We want to store standard macros in general location
5. We want to use the macro from general location
6. We want to not (accidentally) alter the standard macros
7. We want to update these standard macros when needed
8. We want to be able to view older versions of the macro
9. We want to not impact older studies with updates on standard macros
10. We want to not need to update programs when a macro is updated



Version control in LSAF

- In LSAF you can control the versioning of any file when
  - Uploading a file
  - Checking in a file
  - Enable versioning for a file
  - Checking in an output file during the execution of a job

- When a new version of the file is created, this version replaces the old version.
  - You can still call older version to be used (via job)
  - You can set a version limit and if limit is reached, the oldest version will be deleted
  - You can version a file when you have the mandatory permissions

LSAF versioning and macros

- File versioning: Single Macro
  - If update in macro, you would like to save it as a new version
    - If new file version:
      - Need or to adjust the job
      -
      - Create new job for new version
    - If new name to indicate new version
      - Need to adjust job
      -
      - Create a new job for new version

- File versioning: Suite of Macros
  - Out-of-sync sub macro versions:
    - Update 1 sub macro, this macro will get higher version number and be “ahead” in versioning
    - Add sub macro, this macro will get version 1 and will be ‘behind’ in versioning
    - Harder to see which sub macros belong to which major version of the macro

- Versioning works great for files, not for programs or macros

What do we (programmers) want:

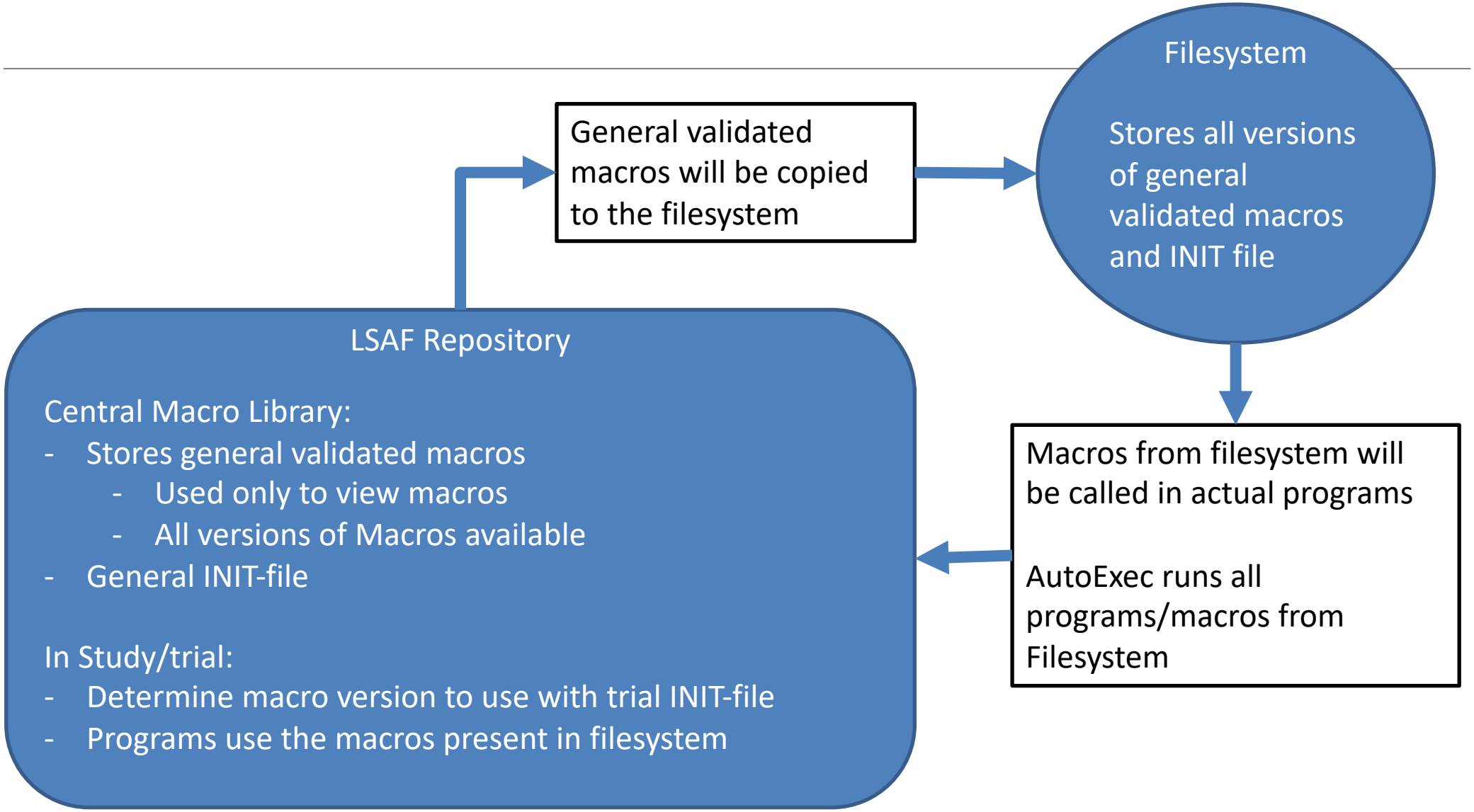
1. We want to work with macros
2. We want to not always need to copy a standard macro into every study folder
3. We want to not have multiple same versions of the macro circulating
4. We want to store standard macros in general location
5. We want to use the macro from general location
6. We want to not (accidentally) alter the standard macros
7. We want to update these standard macros when needed
8. We want to be able to view older versions of the macro
9. We want to not impact older studies with updates on standard macros
10. We want to not need to update programs when a macro is updated

Our Solution

1. General macros

2. Single configuration file – INIT FILE
   - Lists all major versions of all available global macros with every minor version referring to the most current minor version

3. Protected location to store the general macros → read only
   - Where the filename of the macro is the same as the macro name
   - LSAF Specific:
     - Use general macros:
       - Filesystem where general macros are stored (Hidden from view, outside of LSAF)
       - All macros in the filesystem will be executed by autoexec when new SAS Session starts
     - View general macros: General library



1. General macros

- Each macro has major and minor versions:
  - basic_stats_**01**_001 - major version, increment if the macro is incompatible with the previous version. For example, more parameters or different outputs.
  - basic_stats_01_**001** - minor version, compatible changes with the previous version

```
/*<basic_stats_01_001.sas>*/
  %macro basic_stats_01_001 (path=,dataset=,outpath=);
    /*<first_minor version_code>*/
  %mend basic_stats_01_001;
/*</basic_stats_01_001.sas>*/
```

```
/*<basic_stats_01_002.sas>*/
  %macro basic_stats_01_002 (path=,dataset=,outpath=);
    /*<second_minor version_code>*/
  %mend basic_stats_01_002;
/*</basic_stats_01_002.sas>*/
```

```
/*<basic_stats_02_001.sas>*/
  %macro basic_stats_02_001 (path=,dataset=,parameter=,outpath=);
    /*<second_major_version_code>*/
  %mend _baseline_02_001;
/*<basic_stats_02_001.sas>*/
```

2. Global Init-file

- Stored alongside the general macros in the Global library
- Auto executed for every SAS Session or LSAF job based on an LSAF configuration

- This Global Init file should list all major versions of all available global macros with every major version referring to the most current minor version

- Include a **%global** statement and a **%let** statement for every major version

- File updated when macro has a valid update
- Macros controlled globally and does not require any changes on trial level
  → Macro versions will be consistent across trials

```
/*************************************************************************************
  Copyright OCS Life Sciences
*************************************************************************************
  Program     : trial_init.sas
  Author      : Caro Sluijter (CS1)
  Location    :
  Date (ISO)  : 2020-12-08
  Description : Initfile to determine which version of the macro is needed
  Remarks     : None
*************************************************************************************
  Modification history:
  Date (ISO)  User                 Description
*************************************************************************************/

/*<trial init_file>*/
  %global basic_stats_01;
  %global basic_stats_02;
  %global basic_stats_trial;

  %let basic_stats_01=basic_stats_01_002;
  %let basic_stats_02=basic_stats_02_001;
  %let basic_stats_trial= basic_stats_trial;

/*</trial init_file>*/
```

3. Use in a study/trial:

Trial Init-file

- Copy Global Init-file to trial folder
- Most recent macro version at that moment will be called
  - Can be adjusted if needed

- Copy to trail folder because:
- If global init is updated due to update in general macro
  - Trial init-file makes sure that the trial still uses the correct version
  - Can create a trial specific version of the macro that can be called in this file

3. Use in a study/trial:

- Using the macro

```
/*****************************************************************************
  Copyright OCS Life Sciences
*****************************************************************************
  Program     : trial_init.sas
  Author      : Caro Sluijter (CS1)
  Location    :
  Date (ISO)  : 2020-12-08
  Description : Initfile to determine which version of the macro is needed
  Remarks     : None
*****************************************************************************
  Modification history:
  Date (ISO)  User                 Description
*****************************************************************************/

/*<trial init_file>*/
  %global basic_stats_01;
  %global basic_stats_02;
  %global basic_stats_trial;

  %let basic_stats_01=basic_stats_01_002;
  %let basic_stats_02=basic_stats_02_001;
  %let basic_stats_trial= basic_stats_trial;

/*</trial init_file>*/
```

- The init-file sets the value of the macro variable
  - %let baseline_01 = baseline_01_002

- In a SAS program in stead of just calling the normal macro,
  → use a call to a macro variable of the macro.

- Write in program:

```
%&basic_stats_01.(par=, var=);
```

Resolves to:

```
%basic_stats_01_002(par=, var=);
```

- Ensures that when minor update occurs, only INIT-file needs to be adjusted, and validate program is still valid.

Demo

- Demo of this concept in LSAF

Conclusion

OCS Life Sciences Macro version management:

- Easy to implement and maintain
- Keeps control over which version you are using
- Updates on macro do not impact previous studies

OCS Life Sciences

Caro Sluijter

caro.sluijter@ocs-consulting.com
office: +31 (0)73 523 6000

Visit
Ruwekampweg 2G
5222 AT
's-Hertogenbosch
The Netherlands

Mail
Postbus 3434
5203 DK
's-Hertogenbosch
The Netherlands

www.ocs-lifesciences.com