

macro MDU: a flowchart of used programs, datasets and report files based upon the Final Run

Herman Ament, Organon N.V., Oss, The Netherlands

ABSTRACT

Within the Biometrics department of Organon N.V. SAS ® programs are run accordingly to the so-called FINAL RUN procedure. The FINAL RUN is used to create to-be-released statistical output. The FINAL RUN provides a unified way of producing output of a clinical trial that can be reproduced straightforwardly. According to the FINAL RUN the statistical programmer of a trial has to start one program, the Final Run, that calls each SAS program in the appropriate order. The result of the Final Run is the creation of intermediate SAS datasets (so called NOMASTER datasets), final datasets (MASTER datasets) and report files (tables, listings and figures).

Based on the FINAL RUN macro MDU determines which SAS statements are executed per program. By determining the boundaries of PROC and DATA steps and macro calls, macro MDU can determine a) which datasets are read, created and deleted and b) which macros are called by each program. Furthermore macro MDU collects the date-time modified property of the various files found on the file system and it analyses the contents of the LOG and report files .

Based on all collected information macro MDU can make a) flowcharts for the SAS programs of a trial, b) generate the part of the program header that documents which datasets are used and which macros are called within a program and c) perform certain consistency checks within the SAS environment of the trial.

INTRODUCTION

An important part of the statistical programming of a clinical trial/study is the documentation. Documentation is time consuming and not always the most favorite job of a statistical programmer. When last minute changes have to be made in the programs just before a dead line, documentation gets into a tight corner. It would be convenient if programmers would be able to generate documentation automatically.

Part of the documentation is a list of all the SAS ® programs that are run for a trial, together with all the files (datasets and statistical report files) that are read and/or created by these programs. Macro MDU can be used to create this part of the documentation automatically based upon the contents of the programs and based upon the files that are found on the file system.

Almost every Biometrical Department has his own standards and procedures for conducting the statistical analyses of clinical trials. Within macro MDU some of the standards and procedures applicable to SAS programming within department are taken into account.

This paper is split into the following sections:

- Setup of the environment for a clinical trial within our department
- The Analyzing phase of macro MDU
- The Reporting phase of macro MDU
- How to use macro MDU
- Conclusion

SETUP OF THE ENVIRONMENT FOR A CLINICAL TRIAL

To understand the working of macro MDU a short explanation of the SAS environment for conducting the statistical analyses within our department is given. The final part of this explanation is the FINAL RUN, a way to produce statistical output straightforwardly.

FOLDER STRUCTURE

For the statistical analyses of trials in our department a special folder structure exists. All the analyses performed for one compound together are called a **project**. For each project a project folder exists. For each trial that is analyzed a trial folder is created in the relevant project folder with a dedicated and standard structure.

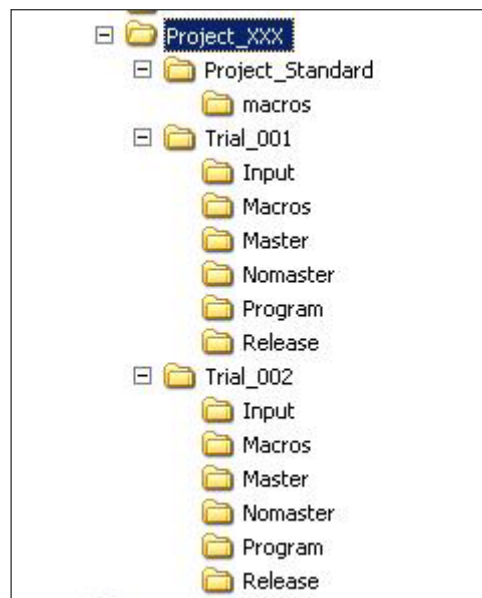
The relevant folders in this trial folder for macro MDU are (See also figure on the right side):

- **INPUT** raw data retrieved from the operational clinical database
- **NOMASTER** intermediate datasets
- **MASTER** final datasets with all raw and derived data
- **PROGRAM** SAS programs and LOG files
- **RELEASE** Output files: tables, listings and figures
- **MACROS** storage of trial specific macros

Besides the trial folders a special standard folder called **Project_standard** exists in the project folder. In this Project_Standard folder one relevant folder for MDU exists:

- **MACROS** storage of project specific macros

When the SAS system is invoked standard library names are assigned for these dedicated folders (relative to the start up folder). The same applies for the search for macros (OPTION FMTSEARCH). A search is performed in the MACROS folder of the trial as well in the MACROS folder of Project_Standard.



SAS MASTER DATA SETS

The folder MASTER contains the so-called SAS Master Data Sets (MDSs). All clinical information/data from the original database are stored in this folder. In principle, all data that are retrieved from the original data base (the data base containing the raw clinical data) and all derived data needed for statistical reports must be represented in the SAS MDSs. The contents of the SAS MDSs are standardized for all trials within a project.

The folder NOMASTER is used to store intermediate results. One of the guidelines is that derivations should be carried out only once. Derivations that have to be passed through from one program to another program are stored in datasets in the folder NOMASTER. Since programs have to be able to run in batch mode, the WORK library can not be used for this purpose.

STATISTICAL REPORTS

The RELEASE folder contains all the statistical report files created by the SAS programs. The original reports are written to text files (for tables and listings) and postscript files (for graphics). The final reports are PDF-files compiled from these text files via a standard macro.

To be able to compile these PDF-files there is some standardization within the statistical report files. The first lines of each page in the statistical reports contain always the title in a specific format. The last line of each page contains almost always the name of the SAS program that created the report. For the graphical (binary) files a separate text file exists containing the title in the specific format. The extensions of text and graphical files are predefined (See (1), Lex Jansen 2001).

FINAL RUN PROCEDURE

Within our department the so-called 'FINAL RUN' procedure exists. The FINAL RUN is used to create to-be-released statistical output. The FINAL RUN provides a unified way of producing output of trial that can be reproduced straightforwardly.

In order to be able to carry out the FINAL RUN, a text file (e.g. finalrun.txt) has to be created listing all programs in the order in which they should be run. This text file is placed in the PROGRAM folder. A corresponding SAS program (e.g. finalrun.sas) is created that calls our standard macro %FINAL01 to run all the listed programs in batch mode.

Before the FINAL RUN is carried out, the folders have to be cleaned by deleting old files, like LOG files and permanent datasets created earlier in the various folders.

The actual FINAL RUN can be split up into several FINAL RUNS. E.g. one FINAL RUN to create the permanent MDSs and one to create the report files.

According to the guidelines within our department all SAS MDSs have to be created before any statistical report is created. Thus, because all data needed for the statistical reports is in the SAS MDSs, no permanent datasets are created by the statistical report programs.

CONCEPT OF MACRO MDU

The concept of the FINAL RUN makes it possible to create a part of the documentation automatically. The FINAL RUN text file implicitly contains the flowchart of the SAS programs. Other information that is used, is the date/time property from the files stored in the various folders and the contents of the SAS and LOG files.

In brief, macro MDU does the following:

Start Phase

1. Read the FINAL RUN text file(s)

Analyzing

2. Read each SAS file found in FINAL RUN text file(s) and create the individual SAS statements by stripping off the comments and string expressions.
3. Collect information from these statements.
4. Read the LOG files found in PROGRAM folder. Collect information about created, read and deleted datasets and collect certain errors.
5. Read the files found in the RELEASE folder and collect the title and footnote information.
6. Collect the property date/time modified of the files in the various folders.

Producing reports

7. Combine all the information above.
8. Create the flowchart documentation
9. Create the program header documentation
10. Create the check reports

ANALYZING PHASE

This section contains technical information about the analyzing phase mentioned in the previous section.

READING THE SAS STATEMENTS

A difficult part for macro MDU is analyzing the SAS statements. The purpose of analyzing the SAS statements is to collect information about:

1. The use of permanent SAS datasets;
2. Whether macros are created;
3. Which macros are called;
4. Whether unwanted statements like ENDSAS in the middle of a program are programmed.

To be able to do so macro MDU tries to identify each SAS statement in a program. Within the statements comments and strings can be deleted because in a program that is well programmed the comments and strings do not contain relevant information for macro MDU. In order to be able to identify the statements, strings and comments, the reading of the SAS programs is carried out by reading each SAS program character by character. Macro MDU keep track of the status wherein the analyzed program is. This status is updated after reading each character.

Here are some examples of the status wherein the analyzed program can be:

- Normal mode
- Within a comment opened by /* or *
- Within a string opened by ' or "

So for example when macro MDU reads the end of line character ';', the following will happen depending on the status:

Current status	New status
Normal mode	Normal mode: end of SAS statement. Next character is beginning of a new statement
Within a comment opened by *	Normal mode: end of SAS statement. Next character is beginning of a new statement
Within a comment opened by /*	No change
Within a string opened by ' or "	No change

Since each statement is stored in a separate observation, after this phase macro MDU can easily determine:

- Which macros are created within the program by looking for statements that start with %MACRO
- Which macros are called within the program by looking for statements that start with % (except for the statement %MACRO).
- Whether unwanted statements are programmed.

How to determine the use of datasets is explained in the next paragraph.

DETERMINING THE USE OF DATASETS IN A PROGRAM

It is complicated to determine the use of datasets in a program. Macro MDU determines the use of datasets by analyzing the standard SAS statements and macro calls separately.

The result of these analyses is which datasets are read, created and/or deleted per SAS program.

Analyzing the standard SAS statements.

Let us look at a simple SAS statement like 'DATA xxx;'. This statement is simple. Any name after DATA determines a dataset that is created. So in this case, dataset xxx is created. However other SAS statements are not so clear. Let us look at an example of PROC SORT.

Example 1:

```
PROC SORT DATA = xxx;
BY NAME;
RUN;
```

Example 2:

```
PROC SORT DATA = xxx OUT = yyy;
BY NAME;
RUN;
```

In example 1 dataset xxx is read *and* created. However, because the phrase 'OUT = yyy' is added dataset xxx is only read while dataset yyy is created in example 2.

In order to be able to determine the use of datasets macro MDU identifies so-called blocks of PROC and DATA steps by looking for statements beginning with PROC, DATA, QUIT and RUN in each program. For each type of block, like PROC SORT, combinations of statements and keywords are defined in a table giving information about the use of datasets.

The following applies to PROC SORT:

- the statement is PROC SORT,
- the keywords are:
 - o 'DATA =' denoting the read and creation (at first) of a dataset.
 - o 'OUT =' denoting the creation of dataset. It overrules the Create from 'DATA ='.

A part of the table looks as follows:

BLOCK	STATEMENT	KEYWORD	READ/CREATE/DELETE
DATA	DATA	DATA	Create for each next word
DATA	SET	SET	Read for each next word
PROC SORT	PROC SORT	DATA =	Both: Read and Create (if no other explicit Create is specified)
PROC SORT	PROC SORT	OUT =	Create
PROC MEANS	PROC MEANS	DATA =	Read
PROC MEANS	OUTPUT	OUT =	Create
PROC	PROC	DATA =	Read (see note below)
PROC	PROC	OUT =	Create (see note below)

Etc.

Note that the PROC block defines all procedures not specified in this table

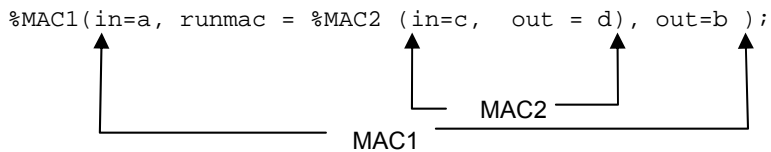
This table is included as a dataset in macro MDU.

Analyzing of the macro calls.

While reading the characters of a program macro MDU keeps track of the number of parentheses opened. In this way macro MDU can determine the start and end of macro calls. These blocks are analyzed separately.

Parameters are identified by the words appearing on the same level of opened parentheses after the delimiter ','. Named parameters are identified by a trailing equals character ('=').

Examine the following statement:



Macro %MAC1 is called on the level of zero opened parenthesis, so the parameters for macro %MAC1 appear on the level of one opened parenthesis. Macro %MAC2 is called on the level of one opened parenthesis, so the parameters for macro %MAC2 appear on the level of two opened parentheses.

In the statement the words 'in = a', 'runmac = %mac2' and 'out=b' appear on the level of one opened parenthesis within the call of macro %MAC1.

In the statement the words 'in=c' and 'out=d' appear on the level of two opened parentheses within the call of macro %MAC2. They do not apply to macro %MAC1 because they are defined on the wrong level of opened parentheses.

With the described method macro MDU identifies parameter statements for each macro call.

Let us look at two examples.

Example 3:

The definition of macro ADD_AGE1:

```
%MACRO ADD_AGE1(dataset=, outset=master.medication, ageset=master.demograp);
```

The call of macro %ADD_AGE1 within the program:

```
%ADD_AGE1(dataset=nomaster.med0, outset=master.medication,
ageset=master.demograp);
```

Example 4:

The definition of macro ADD_AGE2:

```
%MACRO %ADD_AGE2(dataset,outset);
```

Within the macro the following lines appear:

```
DATA med;
    MERGE med master.demograp;
BY SUBJNR;
```

The call of the macro %ADD_AGE2 within the program:

```
%ADD_AGE2(nomaster.med0, master.medication);
```

In both examples datasets `nomaster.med0` and `master.demograp` are read and dataset `master.medication` is created. However macro %ADD_AGE1 uses named parameters, while macro %ADD_AGE2 uses positional parameters. Also note that the use of dataset `master.demograp` is hard-coded within the macro %ADD_AGE2.

Now similar to the standard SAS statements macro MDU uses a table to determine the use of datasets for called macros. The table defines the use of datasets for each named parameter or positional parameters. For each parameter a default value can be given. This value is used at the beginning of the analyzing of a macro block. For hard-coded datasets the positional parameter 0 (=zero) can be used by filling the default value.

Macroname	Parameter	Read/Create/Delete	Default value
%ADD_AGE1	dataset	Read	''
%ADD_AGE1	outset	Create	master.medication
%ADD_AGE1	ageset	Read	master.demograp
%ADD_AGE2	1	Read	''
%ADD_AGE2	2	Create	''
%ADD_AGE2	0	Read	master.demograp

For the standard macros within our Department this table is included as a dataset in macro MDU. The user can provide trial specific macros information to macro MDU via a dataset.

There is a standard set of predefined parameters, like DSETIN and DSETOUT, that the user can use in the definition for trial specific macros. When only these predefined parameters are used within a macro, the user does not have to provide trial specific information to macro MDU via a dataset for this macro.

For each called macro, macro MDU also determines where the macro is located:

- in the program itself
- in one of trial specific MACROS folders
- in one of standard central stored macros of our department
- provided by SAS via the setting SASAUTOS in OPTIONS SET (e.g. %LOWCASE).

ANALYZING THE LOG FILES

The structure of LOG files is quite standard. E.g. the LOG files do not contain comments. Hence, the sentences in LOG files can easily be scanned for :

- use of datasets by looking for sentences like "NOTE: THERE WERE n OBSERVATIONS READ FROM THE DATA SET aaa". This information is added to the information collected from the SAS statements.
- Certain errors and/or warnings.

Please note that it is not reliable to scan the LOG files only and skip the scanning of the SAS programs. Not all information is available within the LOG files. For example PROC SQL does not give information about datasets that are read. Besides that, users can use the option NONOTES to suppress information written to the LOG files.

Also note that in case of sloppy or complex programming techniques analyzing the SAS statements only is not reliable either. Examining the following complex statements:

```
%LET sasstmnt=%STR(DATA demog; SET master.demograp;RUN;);  
&sasstmnt;
```

In this example the permanent dataset `master.demograp` is read. It is difficult to analyze this syntax during the analyzing the standard SAS statements because the contents of the macro variable `&sasstmnt` during runtime has to be determined as well. However the log perfectly states that a certain number of observations are read from dataset `master.demograp`.

ANALYZING THE STATISTICAL REPORT FILES

From the report files the title and, if available, the name of the SAS program are collected. To be able to create PDF-files the format of the titles is standardized within our Department. They appear on top of each page.

In general the name of the SAS program is mentioned in the last line of the output on each page. However the figures are binary files and it is too difficult to analyze these files. Therefore, for graphical files and for text files with no information of the corresponding SAS program, the chronological order of the LOG files and report files is used to determine which program created the report file.

COLLECTING THE PROPERTIES OF THE SYSTEM FILES

The property date/time modified for the files is collected for each individual file on the system. The extension of the file is used to determine what kind of file it is, e.g. a SAS dataset or report file etc.

REPORTING PHASE

Based upon all the data collected macro MDU is able to create the desired documentation. The documentation consists of:

- Flowcharts
- Program header documentation
- Check report

FLOWCHARTS

Three types of flowcharts are generated:

- A general overview of all programs
- A flowchart of the datasets
- A flowchart of statistical report files

These flowcharts are all integrated into one PDF-file.

General overview

The general overview shows clearly per SAS program when datasets are read, created or deleted and what kind of report files are created (see Figure 1).

Table 1-1

**Table of programs of study xxx
With indication if datasets go in or out
and an indication of what kind of report files are created**

No.	Program name	Datasets			Report files			
		Input	Output	Deleted	LIS	TAB	DOC	FIG/FIT
1	D_FORMAT.SAS							
2	N_ALL_LAB_REFE2.SAS	X	X					
3	N_BIOPENDO.SAS	X	X					
4	M_BASECHAR.SAS	X	X					
5	M_ALL_DE.SAS	X	X					
~~~~~								
69	F_61FIGURES.SAS	X						X
70	F_71TABLES.SAS	X				X		
71	F_72TABLES.SAS	X				X		
72	F_73TABLESDOC.SAS	X			X	X		
73	F_74TABLESFIGURE.SAS	X				X		X

**Figure 1. General overview**

The order of the programs is retrieved from the FINAL RUN text file(s). With the general overview the programmer can perform several visual checks like:

- Does a report file create any permanent dataset?
- Are all permanent datasets created before any report file is created?

#### Flowchart of the datasets

The flowchart of the datasets shows clearly per SAS program which particular datasets are read and/or created (see Figure 2). For each standard library name (INPUT NOMASTER and MASTER for read datasets, NOMASTER and MASTER for created datasets) a column exists. All other permanent datasets, if applicable, are mentioned in a separate column 'Other'.

**Table 1-2**

**Master data set flow chart study xxx.**

No.	Program name	Used data set				Created data set	
		Input	Nomaster	Master	Other	Nomaster	Master
14	M_LAB.SAS	LABCENTR LABCOMM LABMARK REFRANGE SAFCONV SAFPARM SAFRANG SAFSYMN SAFUNIT	LABVALUE	BASECHAR			ALL_LAB
15	M_MED.SAS	MEDICATI		BASECHAR	WDD19991.ATCCOD		ALL_CODE ALL_MED

**Figure 2. Flowchart of the datasets**

#### Flowchart of the report files

This flowchart of the report files shows clearly per SAS program which report file(s) are created and per report file which titles are found (see figure 3).

No.	Program name	Report file	Title
23	G_00ALLOC.SAS	G00.LIS	Listing 0 Subject allocation
24	G_01DEMOGRAP.SAS	G01.LIS	Listing 1 Demographic data.
25	G_02GENMEDHIST.SAS	G02.LIS	Listing 2 General medical history.
26	G_03GYNHIST.SAS	G03.LIS	Listing 3 Gynecological history.
27	G_04VS.SAS	G04.LIS	Listing 4.A Vital Signs Blood pressure
			Listing 4.B Vital Signs Heart rate and body weight

**Figure 3.** Flow chart of the report files.

### PROGRAM HEADER DOCUMENTATION

The first part of a program is a comment block that contains general documentation of the program. Items like author, purpose and remarks are entered in the header. Part of the header information states about datasets, which report files are created and which macros are called .

Macro MDU generates a text file for this header. Below an example is given.

```

Input
- Data sets : MASTER.ALL_VS

- Macros   :
  - Standard utility :%APPEND03
                    %FOOTER03
                    %_INCMACS
                    %_VSIDL
                    %TITLE03

  - Study related   :%CHAR2NUM
                    %BLOODLIS                               (Declared in this program)

Output
- Data sets   :
- Files       : G04.LIS
                Listing 4.A                               Vital Signs
                Listing 4.B                               Blood pressure
                Listing 4.B                               Vital Signs
                Listing 4.B                               Heart rate and body weight

```

The user can easily verify and update the header of his/her program by applying macro MDU to the SAS program(s) he/she is developing. Note that In the example above you can see whether a macro is declared in the program itself (macro %BLOODLIS) or stored in the MACROS folder (macro %CHAR2NUM). The macros mentioned in the section 'Standard Utility' are stored on a central version controlled place on our file server.

### CHECK REPORTS

The last part of the documentation concerns several checks. At the moment there are about 45 checks defined for MDU. Checks can be reported in the PDF-file as well as in the header documentation file.

The following table gives some examples of checks made by macro MDU.

Category	Examples	Remarks
Chronological consistency checks	Program has a modification date-time that is later than the creation date-time of one or more output data set(s) it creates.	Apparently the programmer has changed the program after the FINAL RUN. When SAS statements are changed it is not certain whether the statistical reports can be reproduced!

Category	Examples	Remarks
	The modification date-time of the log file of a program precedes the modification date-time of one of the input data sets.	Apparently data is retrieved after the FINAL RUN has been carried. The SAS MDSs and report files can be based upon old data!
	Program occurs more than once in the final run.	This should not happen!
	The program sequence of the finalrun is not consistent with the times of the logfiles. The log date-time of PROGRAM1 is xxx and the log date-time of PROGRAM2 is yyy.	When a user changes and runs a report file after the FINAL RUN, the statistical reports can most likely be reproduced. However when it concerns a program that creates a MASTER or NOMASTER dataset, it is not likely that the statistical reports can be reproduced!
Dataset Consistency checks	Data set aaa.bbb was created more than once. By Programs: xxxx.sas.	This should not happen!
	Data set aaa.bbb is created by program xxx.sas but is never used.	This can indicate that this program of the FINAL RUN is obsolete.
Guidelines	Macro %xxx in the study macro folder is not used in final run	To be sure that only relevant files are stored in trial folder, unused macro should be deleted.
	Program xxx.sas creates a data set with a name that is longer than 8 characters. The data set name is aaa.bbb.	According to the guidelines datasets are restricted to eight characters. For electronic submissions SAS MDSs are converted into SAS transport files. The name of a SAS transport files is restricted to 8 characters.
	Program is not part of the final run.	This can indicate that this program of the FINAL RUN is obsolete.
Unwanted statement	Error: ENDSAS found not at the end of program	When a program contains an ENDSAS statement somewhere in the middle of the program, the remaining part should be removed!
Errors	There was no log file found for a SAS program	This should not happen!
	ERROR: SAS ended due to errors.	This should not happen!

## HOW TO USE MACRO MDU

In this section an explanation is given how macro MDU can be used.

### DURING THE DEVELOPMENT PHASE

Macro MDU facilitates the documentation of a program during the development. Apart from specifying one or more FINALRUN text files, the use can also specify one or more individual SAS files. The generated header information can be copied straight away in the program.

The following statement can be used during the development phase.

```

%MDU(
  path      = ..\program\,          /* folder of TXTFILE and SASFILE ;
  txtfile   = finalrun.txt,        /* finalrun source ;
  parmset   = parmset              /* Dataset with info about study macros; );
```

### DURING THE FINAL RUN

Macro MDU generates automatically a full flow chart of the FINAL RUN including a check report in PDF format.

The following statement can be used during the FINAL RUN.

```
%MDU(  
  path      = ..\program\,          /* folder of TXTFILE and SASFILE ;  
  sasfile   = sasprog1.sas sasprog2.sas, /* the desired SAS files;  
  parmset   = parmset              /* Dataset with info about study  
macros; );
```

### **DURING CHECKING UNKNOWN TRIALS OR OUTSOURCED TRIALS**

Sometimes a user can be asked to have a look at a trial that is unknown to him/her. Unknown trials can be old trials from which programs have to be adapted for a new trial.

Unknown trials can also be trials where the programming is outsourced to a CRO and where SAS programs, datasets and report files are part of the deliverables of the CRO.

By running macro MDU for such a trial a programmer can get quickly insight into the programs.

### **CONCLUSION**

Macro MDU is a useful tool to document the programs of a trial. It also can be used to gain insight in an unknown trial and it can be used to check the programs of an outsourced trial.

### **REFERENCES**

(1) Lex Jansen, Creating PDF® Documents Including Links, Bookmarks and a Table of Contents with the SAS Software, Conference Proceedings, PharmaSUG 2001

### **ACKNOWLEDGMENTS**

I thank Luuk van der Veen and Hans Wijn for their contribution in developing macro MDU.

### **CONTACT INFORMATION**

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

Herman Ament  
Organon N.V.  
Biometrics Department  
Postbus 20  
5340 BH OSS  
Tel. +31 412 663857  
E-mail: [h.ament@organon.com](mailto:h.ament@organon.com)