
How to

- ... cooperate with Statisticians in an effective way
- ... accelerate the production of report objects
- ... create technical specifications out of programs

PhUSE 2009, Basel

Dipl. Dok. Christoph Ziegler, F. Hoffmann – La Roche AG, Basel, Switzerland



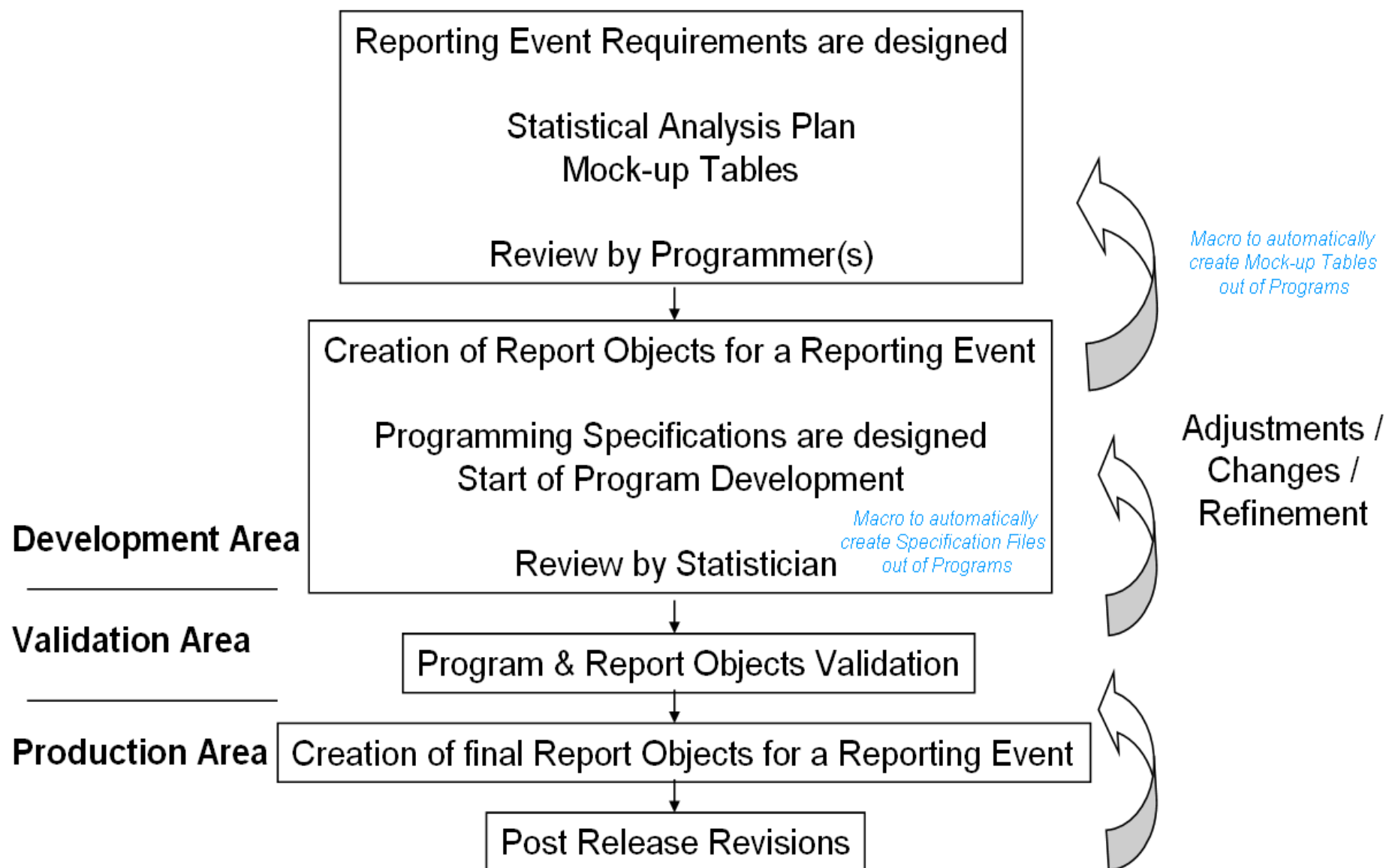
- Introduction
- How to accelerate the Production of Report Objects?
- The Specification and Programming Workflow
- User Requirements
 - What do good User Requirements look like?
 - How to automatically create Mock-Up Tables?
- Programming Specifications
 - How to create Technical Specifications out of Programs?
 - How to meet QA-related expectations?
- How to cooperate with Statisticians in an effective way?
- Conclusions

- User requirements are the basis of all statistical programming activities
- The problem: they usually keep changing and/or get updated on a regular basis
- Final user requirements are usually only available just before the reporting event
- Programming specifications and user requirements should go hand in hand
- This is a time-consuming and resource-intensive process
- Programs are usually created on specifications which are based on non-final data and on non-final user requirements
- Appropriate versioning and tracking of requirements, specifications and programs also need to be considered from a regulatory point of view

How to accelerate the Production of Report Objects?

- In order to address the aforementioned challenges and to accelerate the production of programs and report objects, several options are available
- Improvements should be made on the requirements-side and on the specifications-side, i.e. both Statisticians and Programmers are affected
- The communication paths and the way of working together are also of importance
- A macro to automatically create mock-up tables and a macro to automatically create a program specification document out of a program can be used to support the acceleration of the production of programs and report objects

The Specification and Programming Workflow



User Requirements

What do good User Requirements look like?

- The statistical analysis plan (SAP) should contain a description of the statistical methods (detailed if possible), derivations and calculations used for the efficacy and the safety analysis of a clinical trial and should be in harmony with the trial protocol
 - The SAP is written by the statistician in collaboration with clinical science
 - The description of the statistical methods should be accompanied with mock-up tables (table shells) for each unique data display in order to be able to produce the requested report objects
 - Mock-up tables should always include annotations (text-form), i.e. the statistician should define all related calculations, derivations, exception rules etc. together with the mock-up tables.
- A lot of data problems, obstacles and possible difficulties can be identified beforehand
- Annotations done by the statistician lower the need for later questions by the programmers
- The statistician is able to identify possible layout problems and is able to see possible difficulties in derivation rules

User Requirements



A typical Mock-Up Table

xxx Summary Of xxxx

Protocol(s): xxxxxx

Analysis: Full Analysis Set

	Drug A			Drug B		
	Score	No.	(%)	Score	No.	(%)
Summary of the Means (Total Score)						
Mean (Baseline)	xxx	xxx	(xx.x)	xxx	xxx	(xx.x)
Mean (Treatment Timepoint 1)	xxx	xxx	(xx.x)	xxx	xxx	(xx.x)
Mean (Treatment Timepoint 2)	xxx	xxx	(xx.x)	xxx	xxx	(xx.x)
Mean (Treatment Timepoint 3)	xxx	xxx	(xx.x)	xxx	xxx	(xx.x)
Mean (Follow-Up Timepoint 4)	xxx	xxx	(xx.x)	xxx	xxx	(xx.x)
Change from Baseline to Timepoint 1	xx.x			xx.x		
Change from Baseline to Timepoint 2	xx.x			xx.x		
Change from Baseline to Timepoint 3	xx.x			xx.x		
Change from Baseline to Timepoint 4	xx.x			xx.x		

User Requirements

A typical Mock-Up Table

xxx Summary Of xxxx

Protocol(s): xxxxxx

- Get the data from the XY dataset (parameter XY ='ab',..., 'ah' and parameter XY is in

- ('BASELINE', 'TIMEPOINT 1', 'TIMEPOINT 2', 'TIMEPOINT 3', 'TIMEPOINT 4')

- The score computation should be done per patient (remove duplicates by using the sort key xy).%)

- Compute the number of scores being present per time-point and per patient as follows:

- a=number of assessments present, if a>2 then compute the score as follows: ...

- Sub-scores should not be considered

- Split total score by treatment and compute the means and the change from baseline ...

- The following footnote should be added: total score derived according to ...

Change from Baseline to Timepoint 1 xx.x xx.x

Change from Baseline to Timepoint 2 xx.x xx.x

Change from Baseline to Timepoint 3 xx.x xx.x

Change from Baseline to Timepoint 4 xx.x xx.x

User Requirements

A typical Mock-Up Table

A good mock-up table should include the following elements / information:

- clear and understandable structure of the layout
- detailed wording of the report object title and possible footnotes
- information concerning the population and/or sub-group analysis which should be applied
- information concerning decimal places
- annotations in written form in order to clarify all possible questions and vagueness beforehand
- a report object should be self-explaining, i.e. all important aspects of a report object should be present in the title / footnotes
- possible references to standard programs / macros which should be used in order to produce the report object

User Requirements

How to automatically create Mock-Up Tables?

- Mock-up tables need to be specified to a sufficient level of detail in order to be able to create the programs and to produce the report objects.
- Reality shows that problems might only be seen at the programming (specification) stage
- In order to avoid discrepancies between the actual programs (report objects) and the defined mock-up tables a macro to automatically create a mock-up (dummy) table for each report object can be used.
- It is only executed in the development area
- In the dummy report objects, all numbers are replaced with “x” (or any other character).
- The created dummy outputs are also helpful to confirm the layout with the initial requestor
- Programs are usually developed before database closure and before the actual SAP is signed
- The automatically created mock-up report objects are showing the detailed number of decimal places for numeric values, all footnotes, the title(s) etc. one to one.
- It is important that the dummy report object is only created in the development area

User Requirements

How to automatically create Mock-Up Tables?

- The program to automatically create mock-up tables is called at the very end of a reporting program, i.e. after the actual report object was produced.
- The macro reads in the physical report object produced and creates a so called “dummy” output in the same directory where the report object is stored.
- Dummy in this sense means that all digits (0-9) are replaced with “x” or any other character in the “body part” of the report object.
- The suffix “_dummy” is automatically added to the original report object file name.
- The macro features four parameters:
- **PAGEONEONLY / REPLACESTRING / COLUMN1 / BODYPART**
- In case no report object is produced, e.g. due to a program error, the macro recognizes that the output was not produced and skips the creating of the dummy-report object.

User Requirements

How to automatically create Mock-Up Tables?

1 Title Block

----- <- standard line

2 Column Header(s)

----- <- standard line

3 Body

xxxx

----- <- standard line

4 Footnote Block

1 Title Block

----- <- standard line

2 Spanning Header

----- <- spanning header line

3 Column Header(s)

----- <- standard line

4 Body

xxxx

----- <- standard line

5 Footnote Block

User Requirements



How to automatically create Mock-Up Tables?

Summary of xxxxx

Protocol(s): AAAAAA

Analysis Population: SAP (N=AAA)

Subgroup: Female			

	Drug A	Drug B	Total
	N=..	N=..	N=..
	No. (%)	No. (%)	No. (%)

Observation Time (Months)			
Mean	xx.x	xx.x	xx.x
SD	xx.xx	xx.xx	xx.xx
Median	xx.x	xx.x	xx.x
Min	x	x	x
Max	xx	xx	xx
n	xx	xx	xx
Number Of Patients Observed			
> x Months	xx (xx%)	xx (xx%)	xxx (xx%)
> x Months	xx (xx%)	xx (xx%)	xxx (xx%)
> x Months	xx (xx%)	xx (xx%)	xxx (xx%)
> xx Months	xx (xx%)	xx (xx%)	xxx (xx%)

Footnote aabb

Programming Specifications

How to create Technical Specifications out of Programs?

- Programming specifications should ideally be written before any coding starts and should reflect the user requirements.
- Specifications are usually written in external documents.
- It is often difficult to link versions of specifications to versions of documents, especially in case the company-specific development environment does not support this functionality.
- A macro to automatically create a specification document out of a program can be used in order to have specifications and code closer together.
- This macro would create a specification document by reading program comments (which are marked in a specific way) for a reporting program and for additional called macros (if applicable).
- The macro is only executed in the development area before the respective programs gets loaded to the company specific version control system and before any validation starts.

Programming Specifications

How to create Technical Specifications out of Programs?

- Later on, the created specification documents could easily be combined to a single programming specification document.
- The specification file created for a program can be sent to the statistician for review.
- As soon as the requirements are met and as soon as the specifications are confirmed (by e.g. another programmer), the program can be loaded to the version control system.
- Whenever a program gets updated and re-run, the specification file would automatically be updated.
- The advantage of this automated specification approach is the fact, that specifications and programs are written together, i.e. within the development system, and that the link between programs and specifications is easier.

Programming Specifications

How to create Technical Specifications out of Programs?

- The macro is able to gather specially marked comments in reporting programs (*.sas® - files) and to create two specification documents with identical content (a TEXT- and a PDF®-file) out of the comments gathered.
- The code inside the macro itself is only executed if the program where the macro is called is run in the development area.
- Comments should be marked using "__START_SPECS" and "__END_SPECS"
- All specifications can either be written in one "__START_SPECS" and "__END_SPECS" block or in more than one block.
- The naming convention is as follows:

Report objects folder/<program_name_in_which_the_macro_is_called>_specs.txt

Report objects folder/<program_name_in_which_the_macro_is_called>_specs.pdf®

Programming Specifications

How to create Technical Specifications out of Programs?

- The layout of the *.txt / *.pdf® files looks as follows:

```

Programming Specification For XXXXXXXXXXXX.SAS®                                1
-----

Program Name : /path/xxxxxxx.sas®

Date/Time : DDMMYY/HH:MM

Specs xxxxxx
  
```

- The specifications between the _START_SPECS and _END_SPECS tags are printed one to one, i.e. formatting / layout etc. can be done within the comments.
- In case the macro is called within a program which contains no marked comments, no *.txt or *.pdf® files are produced within the respective folder.
- Some reporting programs include other dependent macros. e.g. test.sas® calls test_macro.sas® which is stored in another folder of e.g. the respective directory.

Programming Specifications

How to create Technical Specifications out of Programs?

- The macro features a parameter (ADDITIONAL_FILES) where dependent / additional programs / macros can be listed (separated by blanks and without the suffix *.sas®).
- In case additional files are specified, the macro also reads the specially marked comments of those programs.
- The macro uses the company specific autocall- search order.
- In case the program is found in one of the directories defined in the autocall order, files with the same name which are higher in the hierarchy are ignored.
- A specially marked comment / specification could look as follows:

```
/*_START_SPECS
```

```
1. Read in data where xxx and yyy
```

```
2. Derive zzz as follows
```

```
- xxxxxxxx
```

```
- yyyyyyyyyy
```

```
...
```

```
_END_SPECS*/
```

Programming Specifications

How to meet QA related expectations?

- Reality shows that it is very difficult and error-prone to write program specifications without writing a single line of code.
- In order to meet regulatory expectations, the macro to create a specification file out of specially marked program comments gets only executed in the development area, e.g. the specification files are available before the actual programs get loaded to the version control system.
- The macro to automatically create a dummy report object should have the same behaviour.
- This ensures that the requirements document (SAP) is updated before the program creating the report object gets loaded to the version control system.

Programming Specifications

How to cooperate with Statisticians in an effective way?

- Automatically created specification files should be reviewed by the statistician and can also include questions, comments and notes.
- The statistician should directly comment and answer within the specification file. This ensures traceability, documentation and consistency.
- If necessary, open questions can be discussed face-to-face, but the outcome should be documented within the specification file.
- The programmer will then implement any possible changes or updates and a new specification file is automatically created.
- By annotating mock-up tables in the SAP, a lot of frontloading is done by the statistician and (data) problems can be detected at an early stage.

- Well annotated table shells help to accelerate the production of programs and report objects for a reporting event.
- By using a macro to automatically create mock-up (dummy) report objects, possible late changes in the layout of report objects can easily be harmonized with the statistical analysis plan.
- The macro to create automated specification files helps to bring technical specifications and programming closer together.
- Programs and technical specifications go hand in hand, i.e. code and specifications get updated at the same place.
- Due to the fact that that both macros are only executed in the development area (before programs get loaded to the version control system) it is ensured, that the mock-up tables and specifications are present before the program starts the development life-cycle.

Thank You



Contact the author at:

Christoph Ziegler

F. Hoffmann – La Roche AG

Malzgasse 30, Bldg. 670

4070 Basel

+41 61 68 89200

christoph.ziegler@roche.com

www.roche.com

Brand and product names are trademarks of the respective companies.