**PharmaSUG 2021 - Paper AD-100**

# A metadata driven approach to mock TFL generation

Tobias Kröpelin, Anne Petersen, and Cristina Boschini, Novo Nordisk A/S

## ABSTRACT

At Novo Nordisk A/S we developed a tool to create specifications for tables, figures, and listings (mock TFL) utilizing the same tools and metadata as when the actual tables, figures, and listings (TFL) are produced. The mock TFL generator consists of a repository of table, figure or listing templates in .txt or .png format (called unique shells) as well as output metadata, including instructions on content and layout for TFL programming. The mock TFL are created by combining the unique shells with output metadata, such as titles, footnotes and programming instructions through a SAS® script. The final mock TFL in .docx format are generated by the same tool used to generate the actual trial TFL. The main advantage of this tool is the single point of definition for the mock TFL and the actual trial TFL. This assures that the programmer only needs to maintain a single repository containing output and programming metadata. Additionally, the simple approach eases alignment across trials and constitutes a good starting point for the set-up of new trials. Finally, this ensures an easier review process, thereby increasing output quality and saving time during trial conduct.

## INTRODUCTION

In Novo Nordisk A/S we use specifications for tables, figures, and listings as a tool to agree with our stakeholders on how trial results should be reported. Furthermore the specifications for TFL function as programming guidance and thus allow several programmers to work independently on the same trial. Having mock TFL is therefore valuable for both stakeholders and programmers. A downside, however, has been the considerable amount of time required to plan and produce the mock TFL, especially for new programmers. For each trial, the dedicated trial programmer creates the mock TFL documents. In these every planned output is represented through a template figure, table, or listing, with specific titles, footnotes and programming instructions. Despite the fact that this is a common task performed for most projects and trials, different approaches on how to create the mock TFL are used. Based on ideas from the different approaches we developed a tool that generates mock TFL in an easy and flexible way.

The main benefit of this approach is the utilisation of the same tools and metadata, which are used when the actual TFL are produced for the final clinical trial report (CTR). This is enabled by the full integration with SECO, a collection of tools used in Novo Nordisk A/S to plan tasks, control programming review and to define output metadata including titles and footnotes. The latter are specifically controlled by an interface tool called PLACE.

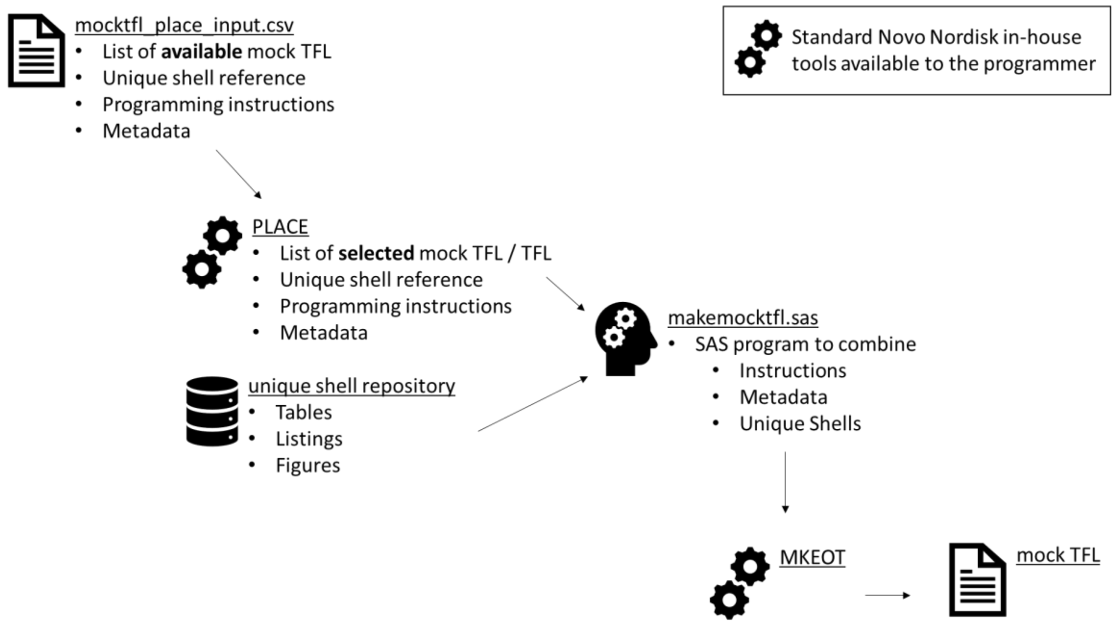
The mock TFL generator consists of three components; a repository of unique shells, a file containing metadata and programming instructions, a SAS® program which creates mock shells. Each component and the process of the mock TFL generator are described in more detail in the following section.

## METHODS

### OVERVIEW OF THE MOCK TFL GENERATOR

The mock TFL generator consists of three components and integrates with two existing tools available to the statistical programmer in Novo Nordisk A/S and routinely utilised during planning and creation of the statistical input to the CTR. The first mock TFL generator component, the mocktfl_place_input.csv file, provides guidance on which mock shells to include, related metadata and programming instructions. The next component is a unique shell repository containing templates for tables, listings and figures each indexed with a unique name. The third component, the SAS® program makemocktfl.sas, is the heart of the mock TFL generator. It combines output metadata with unique shells to create mock shells for every single output planned in a trial.

The interplay of these tools is depicted below (Display 1). First the unique shell references and programming instructions are copied into the in-house tool, PLACE. Afterwards the makemocktfl.sas program is run, where output metadata and programming instructions are read and attached to specific mock tables, figures or listings. Finally the in-house tool, MKEOT, is utilized to concatenate all the individual tables, figures and listings to create the final human readable mock TFL documents.



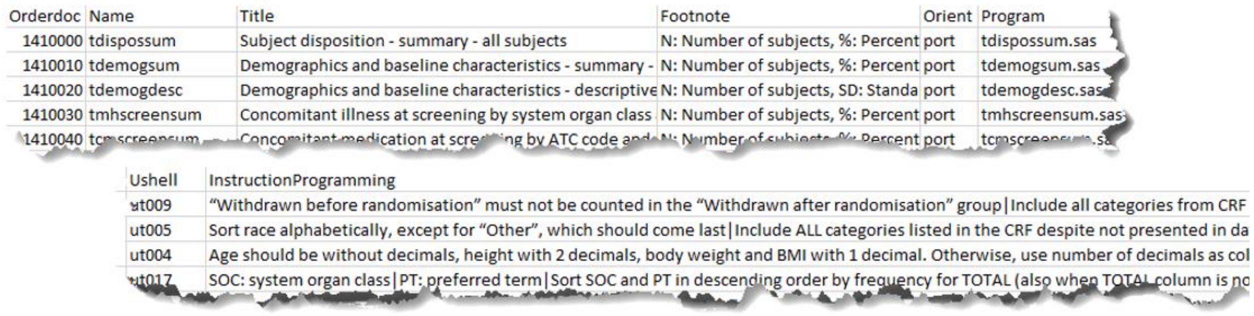
Display 1: Process of mock TFL generator

## INTEGRATION WITH EXISTING TOOLS

PLACE is a planning tool for output creation built to specify the TFL required for the CTR and to organize programming reviews. Within this tool output metadata such as titles and footnotes are defined. MKEOT is a tool that assembles all single TFL to a concise document in .docx format.

## INPUT FOR THE MOCK TFL GENERATOR

The mocktfl_place_input.csv contains the same columns (Display 2) as required for PLACE and is prefilled with output metadata and programming instructions. These instructions can be tailored to every planned table, figure or listing. This enables the trial team to select only the relevant TFL and populate PLACE with their respective metadata and programming instruction. After the initial selection of TFL from the input data file PLACE becomes the single point of definition for all output metadata and programming instructions. In that way PLACE is an efficient tool to define and change output metadata while at the same time ensuring these metadata are applied to mock shells as well as the final TFL deliverables.

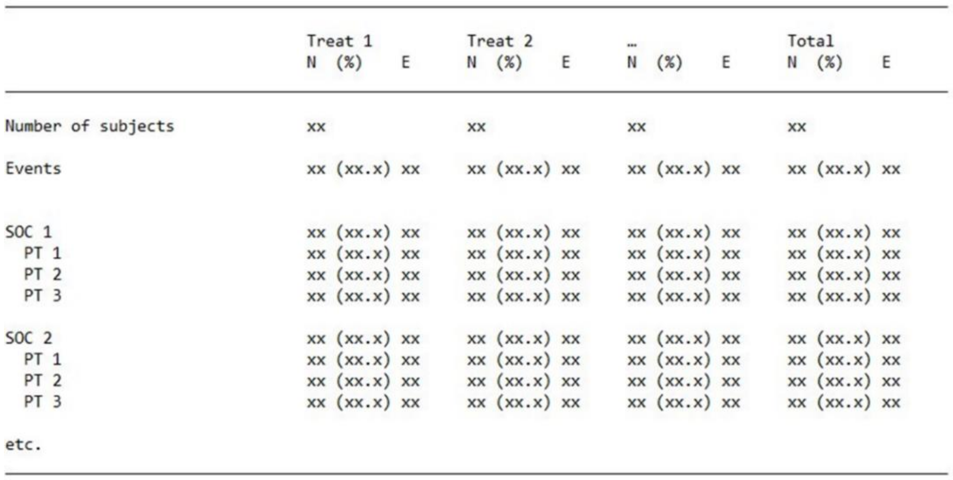


| Orderdoc | Name | Title | Footnote | Orient | Program |
|---|---|---|---|---|---|
| 1410000 | tdispossum | Subject disposition - summary - all subjects | N: Number of subjects, %: Percent | port | tdispossum.sas |
| 1410010 | tdemogsum | Demographics and baseline characteristics - summary - | N: Number of subjects, %: Percent | port | tdemogsum.sas |
| 1410020 | tdemogdesc | Demographics and baseline characteristics - descriptive | N: Number of subjects, SD: Standa | port | tdemogdesc.sas |
| 1410030 | tmhscreensum | Concomitant illness at screening by system organ class | N: Number of subjects, %: Percent | port | tmhscreensum.sas |
| 1410040 | tcmscreensum | Concomitant medication at screening by ATC code and | N: Number of subjects, %: Percent | port | tcmscreensum.sas |

| Ushell | InstructionProgramming |
|---|---|
| ut009 | “Withdrawn before randomisation” must not be counted in the “Withdrawn after randomisation” group\|Include all categories from CRF |
| ut005 | Sort race alphabetically, except for “Other”, which should come last\|Include ALL categories listed in the CRF despite not presented in da |
| ut004 | Age should be without decimals, height with 2 decimals, body weight and BMI with 1 decimal. Otherwise, use number of decimals as col |
| ut017 | SOC: system organ class\|PT: preferred term\|Sort SOC and PT in descending order by frequency for TOTAL (also when TOTAL column is no |

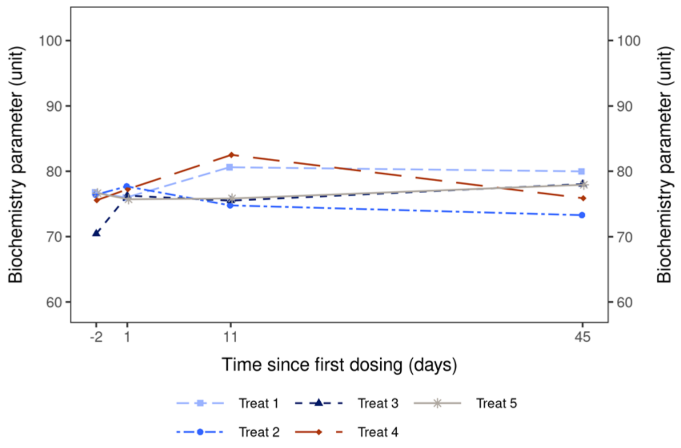
Display 2: mocktfl_place_input.csv example of five outputs (the lower part is an extension to the right of the upper part)

The unique shell repository is a collection of template tables (Display 3), figures (Display 4) and listings (Display 5) to present in the CTR. All unique shells are either in .txt or .png format. Having this repository enables the trial team to easily select and adjust templates according to the trial design e.g. by changing the number of treatment arms or the presented parameters.

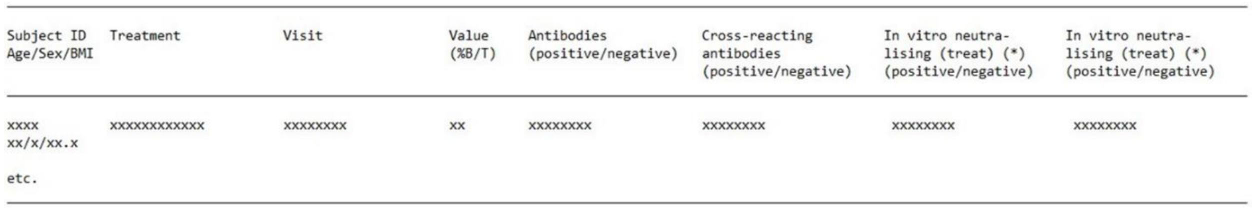


| | Treat 1<br>N (%) E | Treat 2<br>N (%) E | …<br>N (%) E | Total<br>N (%) E |
|---|---|---|---|---|
| Number of subjects | xx | xx | xx | xx |
| Events | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| SOC 1 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| PT 1 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| PT 2 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| PT 3 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| SOC 2 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| PT 1 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| PT 2 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| PT 3 | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx | xx (xx.x) xx |
| etc. | | | | |

Display 3: Example of a unique shell for a table



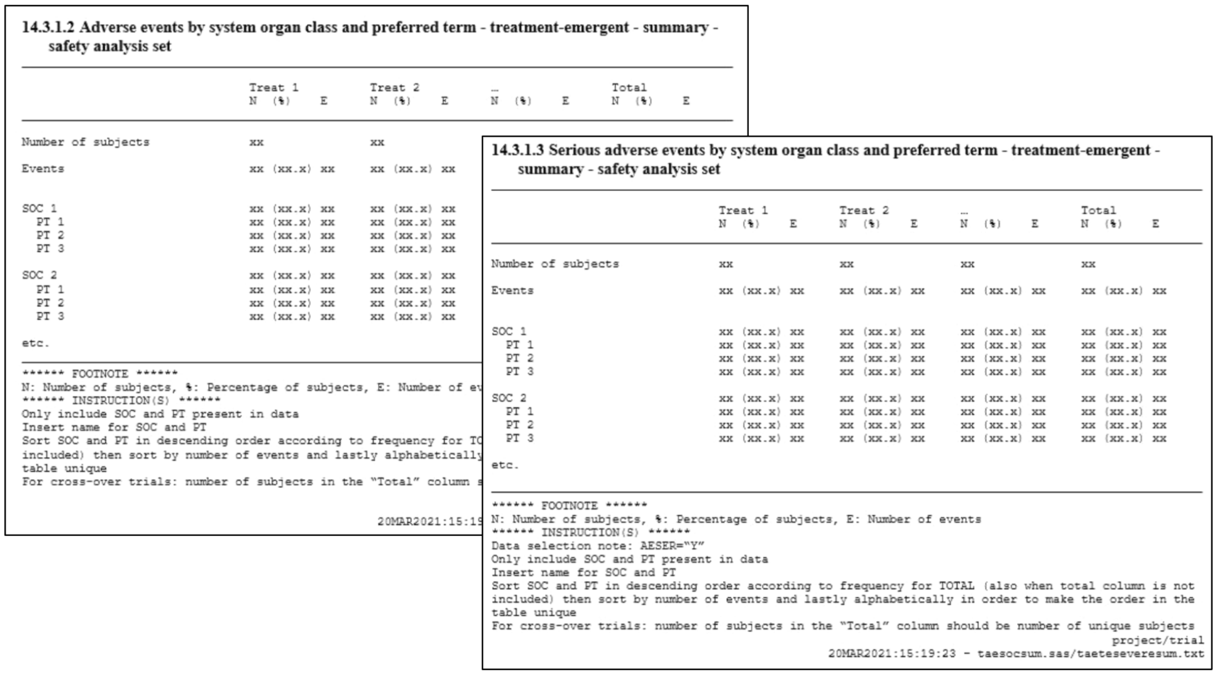
Display 4: Example of a unique shell for a figure



| Subject ID<br>Age/Sex/BMI | Treatment | Visit | Value<br>(%B/T) | Antibodies<br>(positive/negative) | Cross-reacting<br>antibodies<br>(positive/negative) | In vitro neutra-<br>lising (treat) (*)<br>(positive/negative) | In vitro neutra-<br>lising (treat) (*)<br>(positive/negative) |
|---|---|---|---|---|---|---|---|
| xxxx<br>xx/x/xx.x | xxxxxxxxxxxx | xxxxxxxx | xx | xxxxxxxx | xxxxxxxx | xxxxxxxx | xxxxxxxx |
| etc. | | | | | | | |

Display 5: Example of a unique shell for a listing

A unique shell combined with associated metadata and programming instructions constitutes a single mock shell. Detaching the unique shells from metadata and programming instructions makes them reusable within and across trials. For example the unique shell depicted above (Display 3) can be used for several outputs of similar content and layout. The utilisation of the same unique shell for different mock shells is shown below (Display 6).



**14.3.1.2 Adverse events by system organ class and preferred term - treatment-emergent - summary - safety analysis set**

```
                              Treat 1          Treat 2          …                Total
                              N   (%)     E    N   (%)     E    N   (%)     E    N   (%)     E

Number of subjects            xx               xx

Events                        xx (xx.x) xx     xx (xx.x) xx


SOC 1                         xx (xx.x) xx     xx (xx.x) xx
  PT 1                        xx (xx.x) xx     xx (xx.x) xx
  PT 2                        xx (xx.x) xx     xx (xx.x) xx
  PT 3                        xx (xx.x) xx     xx (xx.x) xx

SOC 2                         xx (xx.x) xx     xx (xx.x) xx
  PT 1                        xx (xx.x) xx     xx (xx.x) xx
  PT 2                        xx (xx.x) xx     xx (xx.x) xx
  PT 3                        xx (xx.x) xx     xx (xx.x) xx

etc.

****** FOOTNOTE ******
N: Number of subjects, %: Percentage of subjects, E: Number of ev
****** INSTRUCTION(S) ******
Only include SOC and PT present in data
Insert name for SOC and PT
Sort SOC and PT in descending order according to frequency for TO
included) then sort by number of events and lastly alphabetically
table unique
For cross-over trials: number of subjects in the "Total" column s

                                                  20MAR2021:15:19
```

**14.3.1.3 Serious adverse events by system organ class and preferred term - treatment-emergent - summary - safety analysis set**

```
                              Treat 1          Treat 2          …                Total
                              N   (%)     E    N   (%)     E    N   (%)     E    N   (%)     E

Number of subjects            xx               xx               xx               xx

Events                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx


SOC 1                         xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx
  PT 1                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx
  PT 2                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx
  PT 3                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx

SOC 2                         xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx
  PT 1                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx
  PT 2                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx
  PT 3                        xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx     xx (xx.x) xx

etc.

****** FOOTNOTE ******
N: Number of subjects, %: Percentage of subjects, E: Number of events
****** INSTRUCTION(S) ******
Data selection note: AESER="Y"
Only include SOC and PT present in data
Insert name for SOC and PT
Sort SOC and PT in descending order according to frequency for TOTAL (also when total column is not
included) then sort by number of events and lastly alphabetically in order to make the order in the
table unique
For cross-over trials: number of subjects in the "Total" column should be number of unique subjects
                                                                                        project/trial
                                                  20MAR2021:15:19:23 - taesocsum.sas/taeteseveresum.txt
```

Display 6: Example of two different mock shells using the same unique shell combined with different metadata and programming instructions

The program, makemocktfl.sas, is the core of the mock TFL generator. It combines output metadata, programming instructions and unique shells to produce every single mock shell. In the next step an in-house tool assembles these single mock shells into a concise document presented like the actual TFL appendices for the CTR.

If needed the computed mock TFL can be easily updated by modifying the programming instructions or unique shells and rerunning the makemocktfl.sas program. When creating and updating the trial mock TFL it is important to keep the audience of the mock TFL in mind and accordingly balance the level of detail of the mock shells with the time spent on the process.

## CONCLUSION

The functional integration with existing tools used to produce the actual TFL allows for a single point of definition for the mock TFL and the actual TFL for the CTR. This ensures that programmers only need to maintain a single document containing output and programming metadata. The flexible way the unique shells and programming instructions are handled makes it easy to align across projects and to efficiently get started on new trials. The resulting, highly detailed mock TFL both improve the output quality and enable an efficient collaboration with stakeholders.

Despite the obvious benefits of the mock TFL generator, we have identified some areas for improvement. Implementing updates or corrections to the mock TFL can be time-consuming, because modifications to the unique shell or the metadata requires re-running the SAS® program and re-creating the mock TFL documents. In the future we aim to expand the unique shell repository to include templates for more trial phases, designs or investigational products. Ideally having a predefined list of outputs and unique shells based on the trial design could further speed up the process of creating trial specific mock TFL.

## ACKNOWLEDGMENTS

We would like to acknowledge the work of present and former colleagues who before us worked on streamlining the creation of mock TFL and aligning across projects and trials. This tool was created considering their input and ideas.

## CONTACT INFORMATION

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

Tobias Kröpelin
Novo Nordisk A/S
+45 3079 1142
tkp@novonordisk.com

Anne Petersen
Novo Nordisk A/S
+45 3075 0846
zapt@novonordisk.com

Cristina Boschini
Novo Nordisk A/S
+45 3075 0070
cbcn@novonordisk.com