PhUSE 2017

Paper IS03

GUI Method of Mapping SDTM from Raw Data

Elisa Natale, CROS NT, Verona, Italy
Martina Garlet, CROS NT, Verona, Italy

ABSTRACT

The purpose of this presentation is to describe a method of improving the SDTM mapping from raw data. This approach is both time-saving and very flexible.
This method uses Excel in support of SAS programming. Within an Excel spreadsheet, the programmer can define the operation needed for the mapping such as algorithms for the definition of new variables, renaming or a change in the structure (transposing). More than one input dataset can be used in order to fulfill a variety of data situations.
A SAS macro runs using the raw data and the Excel file as input, and creates a draft of the domain that needs very few adjustments to be the SDTM domain.
This method could save up to 30% of time versus the usual programming method. This is achieved because most of the code is written into the macro without requiring changes, so a programmer just has to write a few of the most significant statements.

INTRODUCTION

The purpose of this project is to speed up and improve the programming activities in a clinical study. This work focuses on SDTM mapping from raw data. The target is to decrease the SDTM developing time thus speeding up the process from raw data to SDTM dataset.

The idea came out during an internship in CROS NT, while performing training for programming SDTM datasets. This method was developed *by* and *for* junior programmers, to make the programming easier for someone new to the industry, as it helps to focus more on the mapping than on the programming side.

At present, the most popular solution consists of a programmer who, using only SAS programming code, maps the raw datasets into SDTM datasets. Some companies have already speeded up this process by using a tool that fixes the order, the labels and length of the SDTM variables. Others have implemented GUI systems (Graphical User Interface) that, through the support of an Excel file, improve the efficiency of the mapping process. The solution developed and presented in this paper combines these two methods currently used, utilising a GUI interface accompanied by a further tool to fix the most important characteristics of the SDTM variables. It allows, through Excel file support, the programmer to considerably reduce the time normally used in programming and increases the range of actions/commands that can be done during the input dataset.

Moreover, writing algorithms into an Excel datasheet decreases the chance of making programming errors resulting in a more efficient programming experience. This is achieved because most of the code is written into the macro and does not need to be changed, so a programmer just has to write a few of the most significant statements. Moreover, this helps the review of the code, since the mapping codes can be more easily seen and understood.

The main chapter of this paper briefly introduces the overall process of the mapping. It focuses on how to compile the Excel document used as support of the mapping, and on how the SAS macro program behind it works. Before the conclusion, the limits and the solution of this approach will be fully discussed.

MAIN VIEW OF THE PROCESS

The purpose is to create a process which will speed up the mapping of SDTM dataset starting from one or more input datasets. As mentioned before, the best approach is to combine a GUI, through an Excel spreadsheet, with SAS codes.

First, the raw data structure and content must be analyzed and the programmer needs to decide the mapping algorithms, which are part of the information that needs to be completed in the excel document. These have a strict structure that will be explained in the next paragraph. This document contains mapping information about every SDTM domain.

Secondly, a SAS program for each domain has to be written.
The macro `%createSDTM` has to be executed (details on this macro will be given later on). This macro program uses both raw data and the Excel spreadsheet as input in order to create a temporary dataset that is the draft of the SDTM dataset.
At this point, the draft domain has to be checked and possible adjustments can now be done (through *ad hoc* SAS steps). When the draft is ready, the statement of the last macro (%createDATASET), which is an internal validated CROS NT macro, can be run. It fixes the length, label and order of the variables in the datasets. Then the SDTM domain is ready for validation.
Below is an example of how the program can be written:

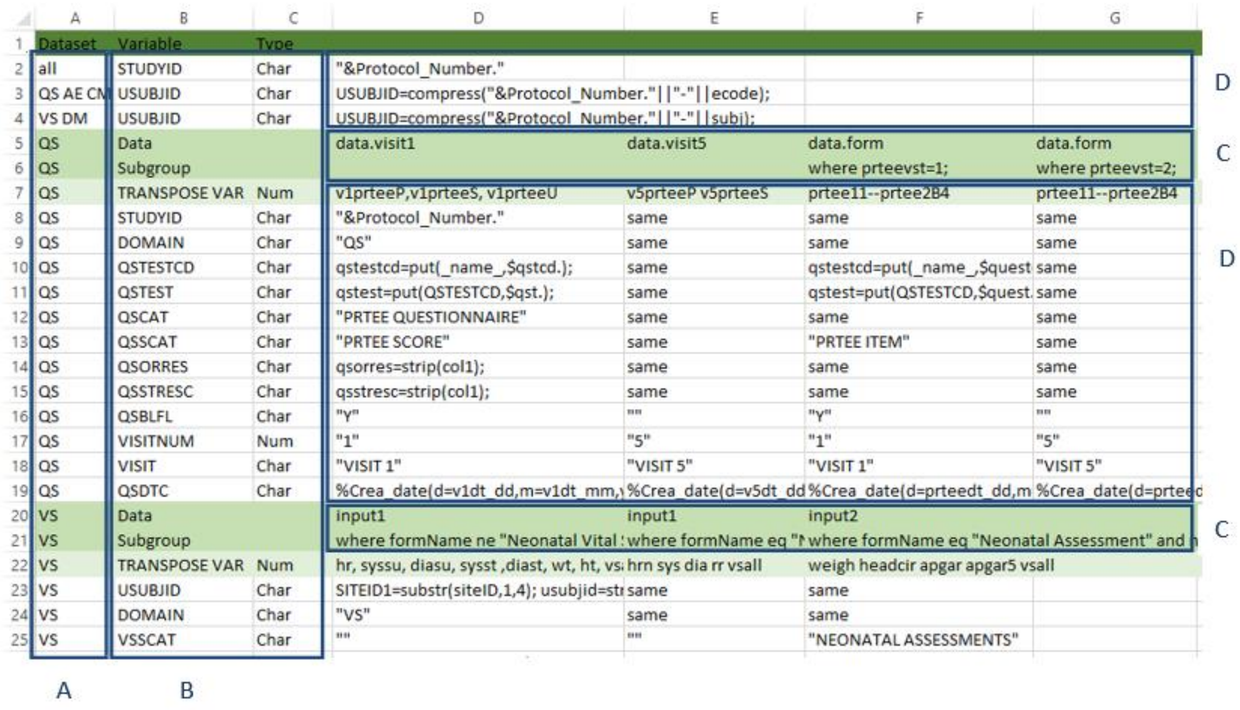
```
%createSDTM(domain=SC);
/*Optional AD HOC steps*/
%CreateDATASET (datashell=SC, dataset=Union);
```

The steps are summarized below:

1. Writing the mapping information of every SDTM domain into an Excel document
2. Writing and running a SAS code for each domain:
   2.1. Recall of macro `%createSDTM(domain=);`
   2.2. Eventually fix the temporary dataset “union”;
   2.3. Recall of the final macro `%createDATASET`.

THE EXCEL DOCUMENT

Usually when an SDTM dataset is created, the programmer has to identify the input datasets and the algorithms to map the variables into the SDTM domain. Then the SAS program with the mapping algorithms has to be written. This process can take a long time for a junior programmer, far longer than for an expert programmer. So, in order to make this process easier and faster, why not write all the algorithms into a more familiar interface?
The idea allows the visualization of the different mapping codes for each SDTM variables even from different input datasets, in order to align them and have a clearer overview of the overall situation.
It was thought that the easiest way was to create an Excel file with all the mapping algorithms.
Figure 1 shows an example of how the Excel file might look like:



	A	B	C	D	E	F	G
1	Dataset	Variable	Type				
2	all	STUDYID	Char	"&Protocol_Number."			
3	QS AE CM	USUBJID	Char	USUBJID=compress("&Protocol_Number."||"-"||ecode);			
4	VS DM	USUBJID	Char	USUBJID=compress("&Protocol_Number."||"-"||subj);			
5	QS	Data		data.visit1	data.visit5	data.form	data.form
6	QS	Subgroup				where prteevst=1;	where prteevst=2;
7	QS	TRANSPOSE VAR	Num	v1prteeP,v1prteeS, v1prteeU	v5prteeP v5prteeS	prtee11--prtee2B4	prtee11--prtee2B4
8	QS	STUDYID	Char	"&Protocol_Number."	same	same	same
9	QS	DOMAIN	Char	"QS"	same	same	same
10	QS	QSTESTCD	Char	qstestcd=put(_name_,$qstcd.);	same	qstestcd=put(_name_,$quest	same
11	QS	QSTEST	Char	qstest=put(QSTESTCD,$qst.);	same	qstest=put(QSTESTCD,$quest.	same
12	QS	QSCAT	Char	"PRTEE QUESTIONNAIRE"	same	same	same
13	QS	QSSCAT	Char	"PRTEE SCORE"	same	"PRTEE ITEM"	same
14	QS	QSORRES	Char	qsorres=strip(col1);	same	same	same
15	QS	QSSTRESC	Char	qsstresc=strip(col1);	same	same	same
16	QS	QSBLFL	Char	"Y"	""	"Y"	""
17	QS	VISITNUM	Num	"1"	"5"	"1"	"5"
18	QS	VISIT	Char	"VISIT 1"	"VISIT 5"	"VISIT 1"	"VISIT 5"
19	QS	QSDTC	Char	%Crea_date(d=v1dt_dd,m=v1dt_mm,	%Crea_date(d=v5dt_dd	%Crea_date(d=prteedt_dd,m	%Crea_date(d=prteed
20	VS	Data		input1	input1	input2	
21	VS	Subgroup		where formName ne "Neonatal Vital	where formName eq "	where formName eq "Neonatal Assessment" and	
22	VS	TRANSPOSE VAR	Num	hr, syssu, diasu, sysst ,diast, wt, ht, vs	hrn sys dia rr vsall	weigh headcir apgar apgar5 vsall	
23	VS	USUBJID	Char	SITEID1=substr(siteID,1,4); usubjid=st	same	same	
24	VS	DOMAIN	Char	"VS"	same	same	
25	VS	VSSCAT	Char	""	""	"NEONATAL ASSESSMENTS"	

FIGURE 1: Example of the Excel.

The Figure shows an example of the top part of the Excel spreadsheet, where the creation of QS and VS domains are. The algorithm used for the creation of every variable can be clearly identified. The details of how it should be compiled are listed below:

- Section A: contains the name of the domains (for example: “QS”). The key word ‘all’ indicates that the corresponding variables will be created the same way in all the domains (in the example above, the variable STUDYID will be created the same way in all the SDTM domains) from any input dataset. If there are just a few domains sharing the mapping of one or more variables, the list of the domains can be written (in the example: USUBJID is created the same way in QS, AE and CM domains, and another way in VS and DM domains.).
- Section B: contains the name and the format of the SDTM variables. It is enough to write ‘Char’ or ‘Num’ in order to define if they are character or numeric data.
- Section C: contains the library and the name of the input dataset in format “libname.dsname”. If the dataset is placed in a temporary library then the name of the dataset is enough (for example: “dsname”). In the below line, any subgroup of the input datasets can be written ending with a semicolon (for example: “Where varx=’txt’; ”). This can be useful when we create many SDTM variables differently within the same ‘filter’, instead of recalling the filter anytime in the algorithms;
- Section D: contains the mapping of the variables. The mapping could be an algorithm, a string or a variable taken as it is from the input dataset. There are some easy rules to follow (listed below) to write them down. If a mapping code is the same between more input datasets, the key word ‘same’ just indicates to use the same algorithms used for the previous dataset. In the above example, since the variable DOMAIN will be the same string for any of the raw data, it is enough to write ‘QS’ in the column of the first input raw dataset (column D) and just ‘same’ in the adjacent cells (columns E, F, G). If no values should be added to a specific variable for a specific input dataset, the cell can be left empty.
    - If there are variables which need to be transposed, the user can list the variables to transpose in the ‘TRANSPOSE VAR’ (lines #7 and #22 of the excel file) line, separated by a blank space, a comma or a hyphen (for example, the user can write “`SBP1,DBP1,HR1,WEIGHT,HEIGHT,BMI`”). When a transposition is made, then the variables ‘col1’ ‘_name_’ and ‘_label_’ are created and they can be used to create other SDTM variables.
    - If the variable is derived with an algorithm, (for example: “`USUBJID=compress ("&Protocol_Number."||"-"||ECODE);`”), the SAS code has to be written, it can be made up of more than one statement, and every statement has to end with a semicolon. The variables used in the algorithm can be either in the input dataset mentioned in section C or other SDTM variables.

In this last case, the SDTM variables used need to be placed in an excel line above the current one. For example, as in the picture, QSTEST uses QSTESTCD for its calculation. This means that QSTESTCD must be placed in a line (line #10) above QSTEST (line #11).

- If the variable has to be imputed as a string, it has to be written between quotation marks (for example for the variable STUDYID the user can write " `'&Protocol_Number'` ").
- If the variable is already in the input dataset and the user needs only to rename it, the user can write the name of the variable of the input dataset (for example if VISIT is already in the input dataset with the name of "VISITCHAR", the user can just write "`VISITCHAR`").

Problems can arise when the raw data contains variables with the same name of SDTM variables for creation, but the program overcomes it renaming those variables by adding an underscore before them, so that they can still be used.

The only variable that should be avoid being written is xxSEQ. Since the way it is computed is always the same, this should be treated differently from any other variables. The next paragraph will also focus on this.

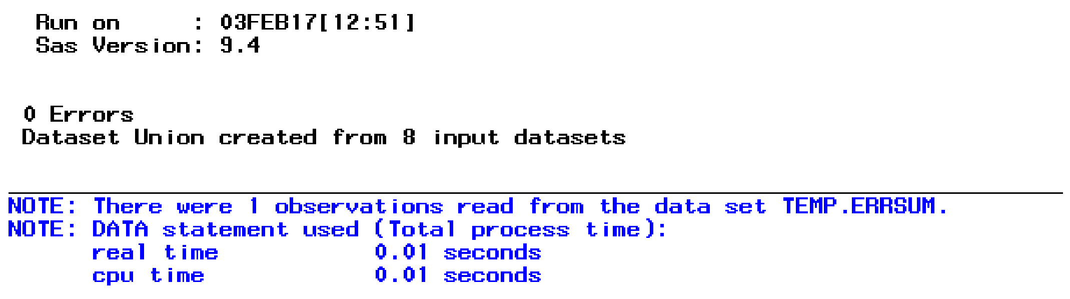
THE PROGRAM

Once the Excel is ready, a SAS code for each domain has to be prepared. As we already pointed out in the first paragraph, there are a few of the usual statements that needs to be settled every time. What we focus on here are the option in the macro `%createSDTM` which uses Excel as an input.

Firstly, the macro needs a parameter to be settled. This is 'Domain'. It has to be settled after the domain name.

The macro has one more optional parameter, SEQ, defaulted to 'Y'. Since many SDTM domains need the variable xxSEQ, the macro normally computes it. Anyway there are cases where xxSEQ is not necessary, or has to be computed later with specific rules. In this case it can just be settled to 'N' (for example: `%createSDTM(domain=DM,SEQ=N))` and then the programmer derives it in the output dataset of the macro.

After the macro has been executed, an SDTM draft placed in a temporary library is created, and a warning message like the one in Figure 2 appears in the log, telling you if there were errors and the number of input datasets used.



```
 Run on      : 03FEB17[12:51]
 Sas Version: 9.4


0 Errors
Dataset Union created from 8 input datasets

NOTE: There were 1 observations read from the data set TEMP.ERRSUM.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

FIGURE 2: Example of a warning message in the log.

If there are no errors, then the SDTM draft dataset can be checked and finalized into the SDTM domain.

THE STRUCTURE OF THE MACRO '%CREATESDTM'

The only information to set, before `%createSDTM` execution, is the macro variable 'Excel' where to indicate the location of the Excel program.
This macro uses both raw data and the Excel as input. Firstly it imports the Excel and processes its information. Every variable mapping code is analyzed and treated differently, depending on the code specified into the Excel file (algorithm, variable, string or key words). Then, these information are used to work on each input raw data according to the following steps:

1. Transposing every input raw dataset, if necessary.
2. Mapping codes are applied to each transposed input dataset.
3. Every modified input dataset is set into a unique SDTM draft dataset called 'Union'.

A resume of the steps is listed below and shown in Figure 3.

1. The import of the Excel dataset;
2. The creation of support datasets;
3. Work on each of the raw data separately:
    - 3a. The import of the raw dataset;
    - 3b. The transpose of the dataset, if necessary;
    - 3c. The creation of the SDTM variables;
4. The union of all the modified input dataset into one, 'Union'.

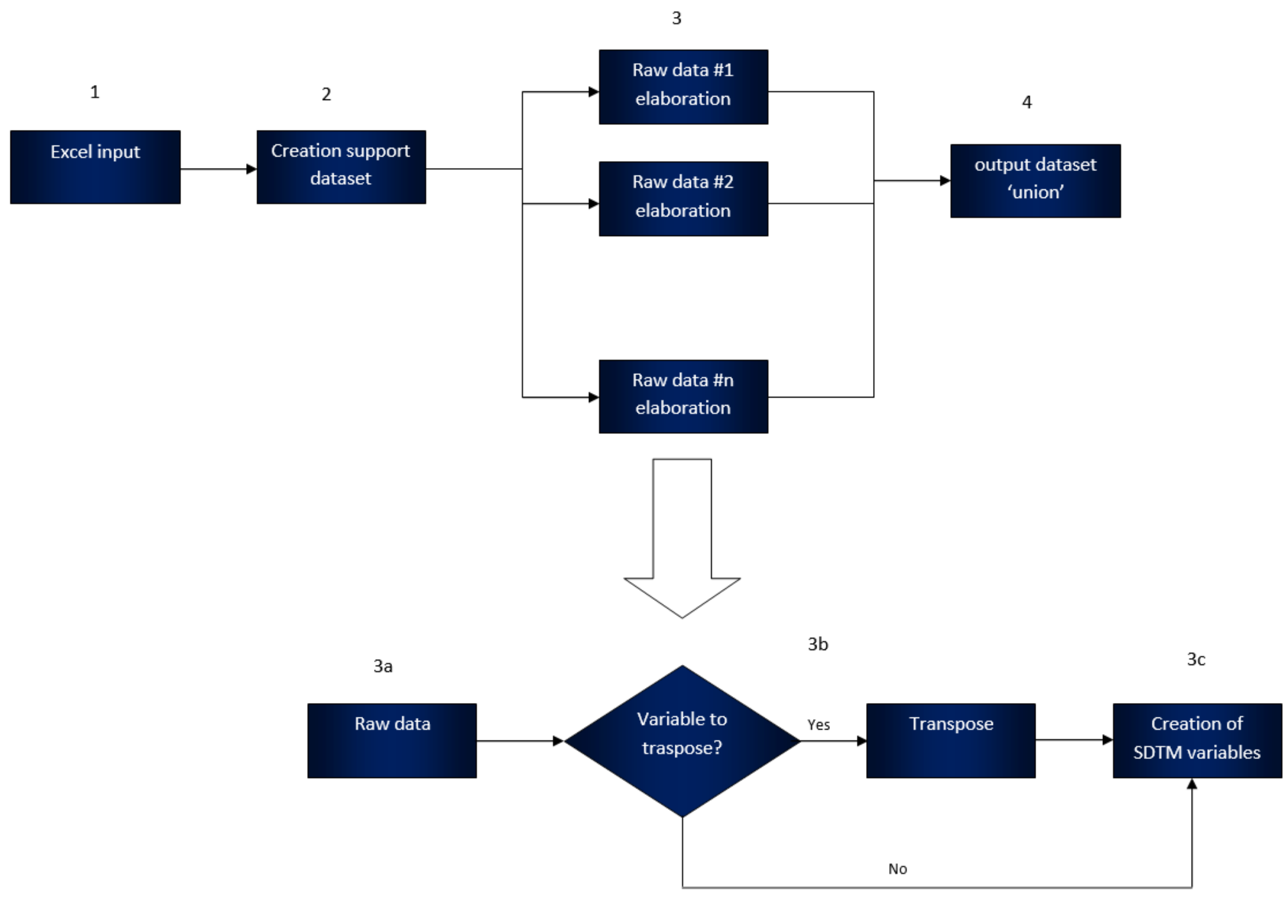


FIGURE 3: Structure of the SAS program '%createSDTM'.

LIMITS AND SOLUTIONS

This macro has been tested and works for almost every SDTM domain, exception for the Trial domains and the Supplement ones.

Anyway, this method has some limits. The first one is about the selection of the records: this program creates a dataset with the structure and the variables of the SDTM domain, but sometimes not all the records must be kept. For example, the IE domain, is called “Inclusion and exclusion criteria not met”. What the program does is to create a dataset with all the inclusion and exclusion criteria, saying if they are met or not. What we should do is to select the ones not met. Up to now, this problem can be fixed editing the dataset ‘Union’ before finalizing it into the SDTM domain.

One related problem is about the creation of new records: this situation is really uncommon for SDTM, since we usually compute new parameters only for ADAM. This macro program doesn’t cover this situation, anyway the new records can be added in the ‘ad hoc’ space. This can happen when, for example, we need to add the calculation of the BMI when it is not collected in the CRF.

CONCLUSION

This program covers many different situations: from the easy case where a SDTM variable is a string defined from the programmer, to the more tricky case where the raw dataset has to be transposed. These steps are not difficult but are time-consuming. Writing the information of more than an input dataset into just one Excel document could save up to 30% of time, especially for a junior programmer. This happens because most of the code is already written and need not be changed, so a programmer just has to write and focus on a few crucial statements. Writing less code increases the efficiency of the process and hence its quality, since it is less likely to make mistakes.

Moreover, the Excel make it easier to understand how every variable is derived, so that when the domain has to be validated, it is faster to find the issues.

REFERENCES

Clinical Data Interchange Standards Consortium, “Study Data Tabulation Model (version 1.4),” 2013

Clinical Data Interchange Standards Consortium, “CDISC SDTM Implementation Guide (version 3.2),” 2015

ACKNOWLEDGMENTS

We would like to thank:

- Paolo Morelli (CROS NT owner), for inspiring this paper.
- Silvia Faini (CROS NT, Principal statistical Programmer), for her support on SDTM standards and processes.

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Contact the authors at:

Elisa Natale
CROS NT
Via Germania 2
Verona 37135, Italy
Work Phone: +39 (0)45 82 02 666
Fax: +39 (0)45 820 58 75
Email: elisa.natale@crosnt.com
Web: www.crosnt.com

Martina Garlet
CROS NT
Via Germania 2
Verona 37135, Italy
Work Phone: +39 (0)45 82 02 666
Fax: +39 (0)45 820 58 75
Email: martina.garlet@crosnt.com
Web: www.crosnt.com