Paper SM03

Validating outsourced programming tasks by comparing SAS datasets without using PROC COMPARE.

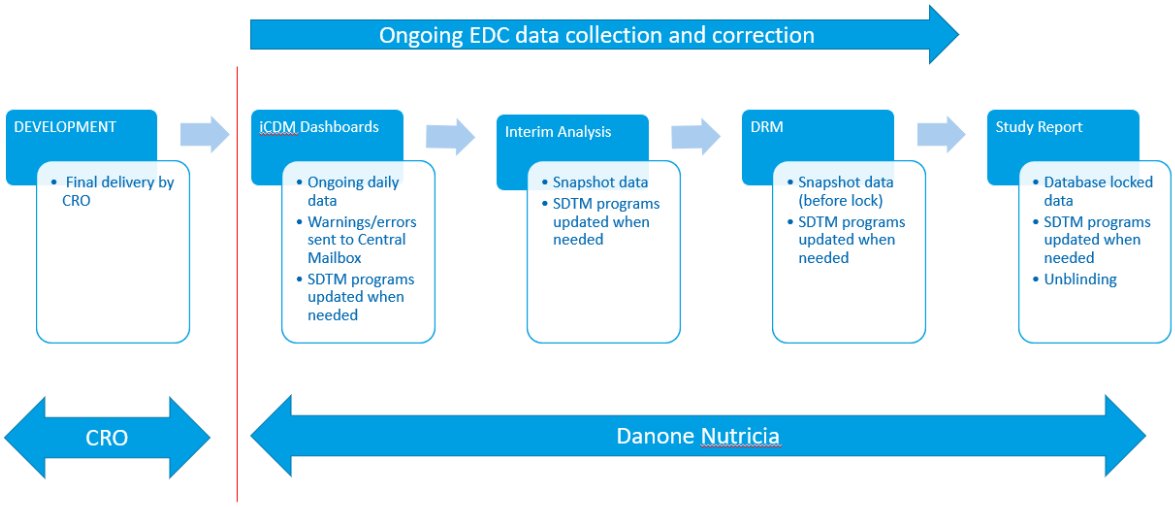
Herman Ament, Danone Nutricia Research, Utrecht, NL

ABSTRACT

Sponsor companies often outsource programming tasks to a CRO, whereby the SAS® programs will be executed in the SAS environment of the company itself. An important requirement is that the delivered programs should give the same results in the SAS environment of the CRO and in the company.
Danone Nutricia Research (DNR) has outsourced some SDTM tasks to a CRO. DNR has an LSAF environment, the CRO is using SAS Studio. The programming structure at the CRO mimics that of DNR, including study specific metadata and global SAS macros.
Deliverables shared with DNR consist of SDTM programs, study specific macros and SDTM SAS datasets. A special component of the deliverables is a hash dataset. This dataset is used to verify that the SDTM SAS datasets created in both environments by the delivered programs and study specific macros are equal.

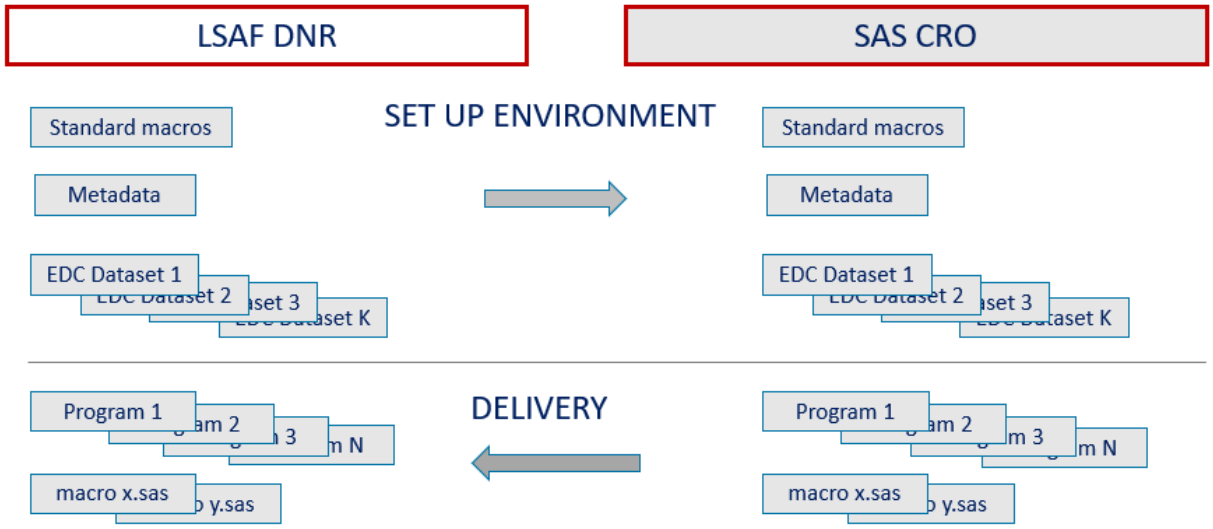
INTRODUCTION

Because of the standard structure of SDTM SAS datasets, DNR is implementing the usage of SDTM data at several milestones of a study. At first, from the beginning of a study, SDTM programs are used for the so-called interactive Clinical Data Monitoring [1)] dashboards, until the end of a study for creating the final study reports. This means that the programs are created already very early of in the lifecycle of a study – or even before the start of a study based on real or dummy EDC data. Due to several deliverables of the milestone of a study, mid-study changes of e-CRFs in the EDC systems, unforeseen data quality issues, updating programs might be needed at several stages.
Figure 1: Lifecycle of SDTM programs of a study



For several studies Danone Nutricia Research (DNR) has outsourced the creation of SDTM programs to a CRO. Standard macros and standard (versioned) metadata are shared with the CRO. SAS Datasets, often with dummy data, are provided by DNR to the CRO.
Next step is that CRO will develop the SDTM programs in 2 or more iterations. During the development, mapping sheets are updated. QC documents keep track of the findings and solutions provided by the developer. Each iteration ends with a delivery to DNR. DNR will check the results of the programs. After 2 or more iterations the development by the CRO is completed and the delivery is considered final. During the lifecycle of the study, the SDTM programs are updated and validated in the DNR LSAF environment.
For delivery by the CRO, all programs plus additional study specific macros are shared with DNR. DNR will upload the programs plus additional study specific macros in LSAF. In LSAF all programs are executed and all SDTM datasets are created.

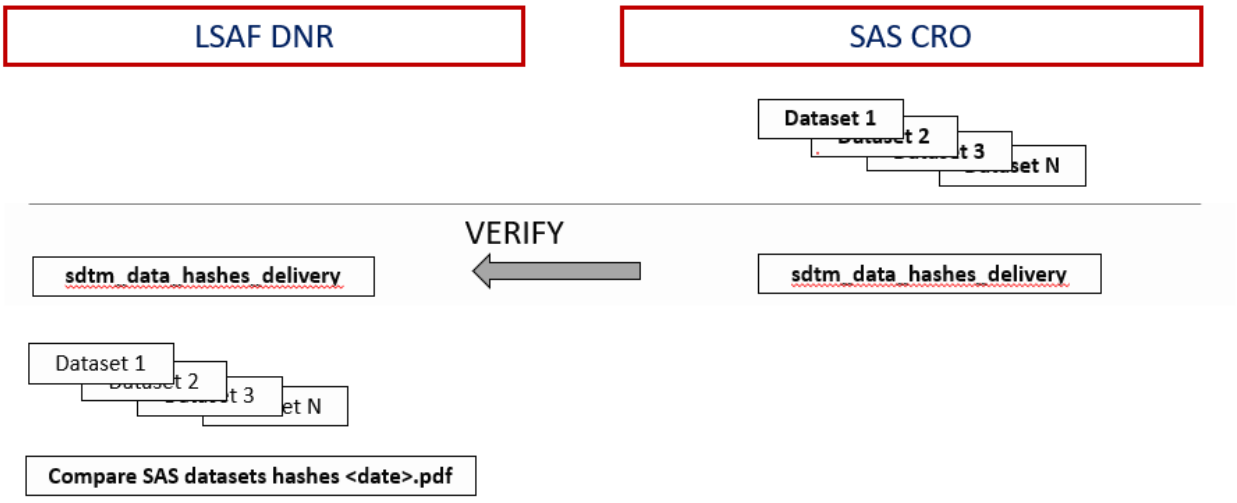
Figure 2: Transfers between DNR and CRO per delivery



One step to verify the delivery is to compare the SDTM SAS datasets at the CRO with the SDTM SAS datasets created by the delivered SAS programs in the DNR environment.

To avoid duplicate SDTM SAS datasets in LSAF that never will be used, a comparison of the SDTM SAS datasets in both environments is done via a hash dataset. If the hashes in the hash datasets of the SDTM SAS datasets on both sides are the same, it can be concluded that the datasets are equal on both sides. Hashes (or checksums) are created from SDTM SAS datasets.

Figure 3: Verify SDTM datasets via a hash dataset.



HASH BACKGROUND

From SAS Help [2)]: "Hashing is the transformation of data (known as a message) into a short fixed-length value (known as a digest) that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the hashed key than using the original value. Hashing functions are useful when you are developing shell scripts for software installation, file comparison, and detection of file corruption and tampering."

SAS supports the following hash algorithms: CRC-32, MD5, SHA1, SHA256, SHA384, SHA512

Examples of hashing the text 'Hi'

Method	Hash (checksum) in hexadecimal presentation (Digest)	Display Length
CRC-32	4D170E0E	8
MD5	C1A5298F939E87E8F962A5EDFC206918	32
SHA1	94DD9E08C129C785F7F256E82FBE0A30E6D1AE40	40
SHA256	3639EFCD08ABB273B1619E82E78C29A7DF02C1051B1820E99FC395DCAA3326B8	64

In general, the larger the hash, the less chance the same hash is created from different original strings. The display length of method CRC-32 is 8, a number of 32 bits. With method MD5 there are approximately 3.4×10^{38} (number with 128 bits), different hashes, while SHA256 has approximately 1.16×10^{77} (256 bits) and CRC-32 has approximately 4.3×10^{9}.

At DNR it was chosen to use the CRC-32 method. The hash string is short, several hashes can be displayed easily next to each other, and the chance on a collision is very low (10^{9}). This is very acceptable for comparing SAS datasets for our purpose.

HASHING SAS DATASETS

Each SAS dataset contains metadate like date modified, date created, SAS version used, running operating system, and so on. Therefore, creating a hash from the original SAS dataset itself is not useful and another method has to be used.

If we look at PROC COMPARE, we can see that results of PROC COMPARE are split up into comparing the data itself and comparing the metadata of SAS dataset [3]. Data is compared on the values of variables in both datasets, and the number of observations Compared metadata are label, length, type, format and informat of the variables, and variables not in both SAS datasets.

For our purpose, hashes are created on the contents, on the attributes of the variable names, and on the SAS dataset label. For each part, relevant data is written to a temporary file. After this, the hash is calculated with function HASHING_FILE.

First an example of the metadata. In this case, the contents of the temporary file for the attributes label (all missing), name and type (1 or 2) of the variables of SAS dataset SASHELP.CLASS.

```
1# #Age#1## 2# #Height#1## 3# #Name#2## 4# #Sex#2## 5# #Weight#1###
```

Second example is for the first record from SASHELP.CLASS: Age, Height, Name, Sex and Weight for Alfred.
In this example, the stored numeric values in the SAS dataset are written as 64 bits using IEEE standard via format BINARY64. [4] For the comparison of the hashes, the exact numeric values are used.

```
 1#0100000000101100000000000000000000000000000000000000000000000000
000#0100000001010001010000000000000000000000000000000000000000000000
0000#Alfred#M#0100000001011100001000000000000000000000000000000000
00000000000000## 2#0100000000101010000000000000000000000000000000
```

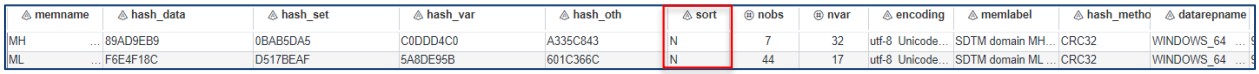
CREATING A HASH DATASET

To create a hash dataset, hashes are created for/from:

- all data in the SAS dataset
- the label of the dataset
- variable attributes name, type and label
- variable attributes format, informat, sorted and varnum

Together with other metadata like the number of observations, the number of variables, hash method, SAS version, and the system environment, one row is created for each SAS dataset.

Below a screenshot of the hash datasets from SDTM SAS datasets MH and ML is shown



memname	hash_data	hash_set	hash_var	hash_oth	sort	nobs	nvar	encoding	memlabel	hash_metho	datarepname
MH	89AD9EB9	0BAB5DA5	C0DDD4C0	A335C843	N	7	32	utf-8 Unicode...	SDTM domain MH...	CRC32	WINDOWS_64
ML	F6E4F18C	D517BEAF	5A8DE95B	601C366C	N	44	17	utf-8 Unicode...	SDTM domain ML ...	CRC32	WINDOWS_64

In case the sorting of the SAS datasets might be a reason for a mismatch, the SDTM SAS dataset can be sorted by all variables before creating the hashes. In that case the sort is set to Y and the result of hash data is different.

MACROS

We have created the following SAS macros for this purpose.

`%hash_set`	creates a hash dataset of one SAS dataset
`%hash_lib`	creates a hash dataset of SAS datasets in one library
`%hash_readback`	creates a SAS dataset by reading back the temporary written flat files
`%hash_verify`	applies macro `%hash_set` for one SAS dataset, and then applies macro `%hash_readback` and compares the original SAS dataset with the SAS dataset created by macro `%hash_readback`. Used to validate macro `%hash_set`

COMPARING A FINAL DELIVERY

Each delivery is accompanied by a hash dataset. After uploading and executing all programs that are delivered, the final step is running a program that creates a PDF file which demonstrates that the SDTM SAS datasets are equal to the SDTM SAS datasets at the CRO. The PDF file contains the results of the comparison of the hashes. Via traffic light colors the reviewer can easily see whether all SDTM SAS datasets are equal or not. If a cell is not green, there is a mismatch.

An example of a part of the output with no issues

Compare hashes result of delivery 2023-05-23

Result	Sort before creating hash data	Source	Member Name	Hash of data	Hash of memlabel	Hash of attributes name, label, type	Hash of other attributes (format, informat, sorted, length, varnum)
All Match	N	LSAF	MH	89AD9EB9	0BAB5DA5	C0DDD4C0	A335C843
	N	Delivered	MH	89AD9EB9	0BAB5DA5	C0DDD4C0	A335C843
All Match	Y	LSAF	MH	6E1A5C8C	0BAB5DA5	C0DDD4C0	A335C843
	Y	Delivered	MH	6E1A5C8C	0BAB5DA5	C0DDD4C0	A335C843
All Match	N	LSAF	ML	F6E4F18C	D517BEAF	5A8DE95B	601C366C
	N	Delivered	ML	F6E4F18C	D517BEAF	5A8DE95B	601C366C
All Match	Y	LSAF	ML	C344131E	D517BEAF	5A8DE95B	601C366C
	Y	Delivered	ML	C344131E	D517BEAF	5A8DE95B	601C366C

An example of part of the output with issues.

Data differ	N	LSAF	EC	AD61EA3C	C38DFA2C	FBB031A5	832D8A10
	N	Delivered	EC	879AC552	C38DFA2C	FBB031A5	DEF8EE48
Data differ	Y	LSAF	EC	2AB1D1DF	C38DFA2C	FBB031A5	832D8A10
	Y	Delivered	EC	E0325429	C38DFA2C	FBB031A5	DEF8EE48

REASON OF MISMATCH

When there is a mismatch, the SDTM SAS datasets are not exactly the same on both sides. If a dataset is only present on one side, there is also a mismatch.

In case of a mismatch of SDTM SAS datasets on both sides, further investigation is needed. Further investigation may depend on the number of SDTM SAS datasets that differ.

For example, the content of STUDYID, is specified in a set-up program to be maintained on both sides. A difference in STUDYID will result in mismatch for all SDTM SAS datasets.

Other possible reasons of a mismatch:

- Different EDC source data, resulting in different numbers of observations and thus differences in hashes of the data. Many datasets are unequal
- Metadata not applied in a program, e.g. sorting is not applied. Differences in hashes of other attributes
- Different version of the DNR metadata: (a few or many differences)
    - Differences in the labels of variables
    - Differences in the numbers of variables
- Different SAS system options, for example OPTIONS MISSING =

CONCLUSION

This approach offers a very convenient way to ensure that programs create exactly the same SDTM SAS datasets on both sides. Only one small dataset is needed to verify that all created SAS SDTM datasets are the same.

I would like to conclude with a message to SAS. Could you please consider adding hash values as metadata to SAS Datasets? If those hashes can be retrieved with PROC CONTENTS, or from DICTIONARY.TABLES, they can be used to compare SAS datasets without PROC COMPARE.

REFERENCES

1) iCDM - interactive dashboards that innovate central data monitoring at Danone Nutricia Research Paul Vervuren, Danone Nutricia Research, Utrecht, The Netherlands Dennis Menkveld, PW Consulting, Culemborg, The Netherlands, Phuse EU 2022, Belfast (IRL)
   PAP_AR05.pdf (phuse.s3.eu-central-1.amazonaws.com)
2) SAS Help Center, Hashing Functions and Hash-Based Message Authentication Code
   SAS Help Center: Hashing Functions and Hash-Based Message Authentication Code
3) SAS Help Center, Results: COMPARE Procedure
   SAS Help Center: Results: PROC COMPARE
4) SAS Help Center, How SAS Stores Numeric Values
   SAS Help Center: Numeric Precision

CONTACT INFORMATION

Herman Ament
Danone Nutricia Research
Uppsalalaan 12
3584 CT Utrecht
Email: herman.ament@danone.com