PhUSE 2016

Paper DH13

From SDTM to displays, through ADaM & Analyses Results Metadata, a flight on board METADATA Airlines

Omar SEFIANI, Boehringer Ingelheim, Reims, France
Stéphane BOUGET, Boehringer Ingelheim, Reims, France

ABSTRACT

The goal of a submission is to report patient data based on specific analyses. Each step of this journey uses the appropriate set of standards, but in order to keep efficiency and flexibility, generic programming is a valuable approach, to avoid duplicated or custom code.

During the conduct of a project, many changes can occur due to standard updates leading to structural changes, new scientific approaches, or new regulatory requests that lead to various new specifications for different analyses packages.

Everyone knows how lazy programmers can be … so this presentation will highlight the full process from SDTM to report, based on the experience of implementing a metadata-driven system across a multi-trial respiratory project. It will show that metadata gives more transparency; it emphasizes the global picture to automatically detect and manage specific differences within a trial, but also across trials within a project.

INTRODUCTION

Metadata is basically data information about other data.

A typical example of metadata is the metadata about CDISC datasets (structure and format) that is stored in the define.xml (variable names, labels, formats …). This can be used to automatically generate the SAS ® datasets structure.

Metadata could also be used for the automatic generation of executable SAS programs in a generic way (SDTM conversion macro calls, display macro calls …).

Additionally, metadata could be used to implement any BDS endpoint via a generic SAS program, without any modification.

This paper will describe how metadata-driven programming was implemented in a respiratory project. It has been used in BI legacy data conversion to SDTM and also ADaM transformation. This is currently under development for the reporting part, the goal is to generate macro calls for the data building, analysis and reporting.

METADATA DRIVEN PROGRAMMING

This approach of metadata driven programming is based on the identification of algorithms by meaningful KEYWORDS that describe the functionality instead of directly embedded executable SAS code.

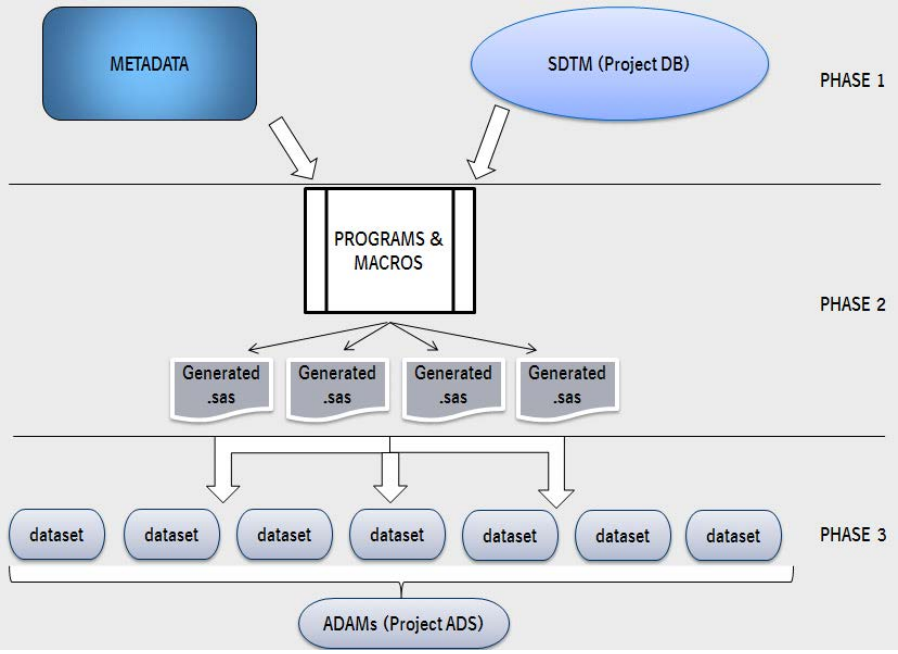
The keyword approach gives the flexibility to change the implementation of the functionality without changing the entire program or other existing metadata information.

This also promotes modular programming in order to reduce maintenance effort and side effects due to updates.

Each piece of metadata used by the generic program is an extracted TABLE from a database.

The goal of this technique is to centralize and automate similar processes as much as possible.

This approach increases the traceability and gives overview on differences and similarities across trials and endpoints.



An optimized system should be able to work correctly by updating the metadata repository without any modification of core programs.
Some techniques are very powerful in metadata retrieval (Hashcode, DoSubl, Macros …).

As an example, we will show the setup definition of the multiple exposure rules for some trial variables. Due to the fact that one patient can participate in several trials within a project, we need to define an aggregated rule to retrieve one single value for the pooling of several trials together.

In the below metadata sheet, each RULECBxx flag modelizes a way to combine trials together:

Name	*ADAM*	*SRC_DS*	*VARNAME*	*RULECB01*	*RULECB02*	*RULECB03*
Label	*Target ADaM*	*Source Dataset*	*Predecessor variable name*	*Rule to apply for COMB01FL*	*Rule to apply for COMB02FL*	*Rule to apply for COMB03FL*
Type	*text*	*text*	*text*	*text*	*text*	*text*
Format	*$8.*	*$30.*	*$200.*	*$40.*	*$40.*	*$40.*
Case	*UPPER*	*UPPER*	*UPPER*	*UPPER*	*UPPER*	*UPPER*
Context (Where condition)						
	ADSL	DM	BRTHDTC	NONE	FIRST	FIRST
	ADSL	DM	AGE	NONE	FIRST	FIRST
	ADSL	DM	AGEU	NONE	FIRST	FIRST
	ADSL	DM	SEX	NONE	FIRST	FIRST
	ADSL	DM	RACE	NONE	FIRST	FIRST
	ADSL	DM	DTHDTC	NONE	LAST	LAST

For instance, the AGE variable from SDTM DM used to create the ADaM ADSL can have multiple rows per patient (USUBJID) if the patient participated to several trials.
Currently, there is a draft paper about ADSL integration (ADaM Data Structures for Integration: General Considerations and Model for Integrated ADSL (IADSL) Version 1.0).
In the integrated iADSL, we will need to select only one value for each patient.
In this example, we select the AGE of the FIRST trial for the two rules RULECB02/RULECB03.

Here is the illustration of the metadata usage consequence, showing the dynamic definition of the variables and the hashcode initialization plugged into the data from one side, and into the metadata in the other side.

Dynamic assignment of variables format and label + hashcode tables initialization:

```
format %mxutil_setup_format(mxds=setupcomb_&mxcombvar.,mxlist=ADAM SRC_DS VARNAME);

format %mxutil_setup_format(mxds=&mxindata.,mxlist=&listvar_prop.);

format &mxcombvar. %VarNameFormat(&mxincombds.,&mxcombvar.);

label %mxutil_setup_label(mxds=&mxindata.,mxlist=&listvar_prop.);

label &mxcombvar.="%VarNameLabel(&mxincombds.,&mxcombvar.)";
```

```
%do __i=1 %to &__nb_prop.;
    %let __varname = %upcase(%scan(&listvar_prop.,&__i.));
    * dataset used to store the list of values for each variable in original order;
    declare hash hhlistvalr_&__varname.(MULTIDATA: "y", ORDERED: "n");
    hhlistvalr_&__varname..defineKey(%mxutil_add_quote(mxlist=&listvar_key.,mxseparator=%str(,),mxquote=Y));
    hhlistvalr_&__varname..defineData("r_&__varname.");
    hhlistvalr_&__varname..defineDone();

     * dataset used to store the list of unique values for each variable in asc order;
    declare hash hhunivalr_&__varname.(MULTIDATA: "y", ORDERED: "a");
    hhunivalr_&__varname..defineKey("r_&__varname.");
    hhunivalr_&__varname..defineData("r_&__varname.");
    hhunivalr_&__varname..defineDone();
%end;
* to get the SPMULTEX rules for the current ADAM for the current domain for all variables;
declare hash hhmultex(dataset: "setupcomb_&mxcombvar." , multidata: "y");
        hhmultex.definekey("adam", "src_ds", "varname");
        hhmultex.definedata("rulecb");
        hhmultex.definedone();
        call missing(rulecb);
```

Example of implementation of RULECBxx = MIN

In the example above, the purpose of the hashcode is to retrieve the correct keyword, and for each keyword, a corresponding algorithm is defined.

```
* All The Rules for Numerical variables only;
else if RULECB = 'MIN' then do;
    %if &__vartype. = N %then %do;
        if not missing(r_&__varname.) then do;
            if missing(&__varname.) then &__varname.=r_&__varname.;
            else if &__varname. > r_&__varname. then &__varname.=r_&__varname.;
        end;
    %end;
    %else %do;
        erratumtype='RULE NOT FOR CHARACTER VARIABLE';
        rc7 = hhmultex_erratum.check();
        if rc7 > 0 then rc7 = hhmultex_erratum.add();
    %end;
end;
```

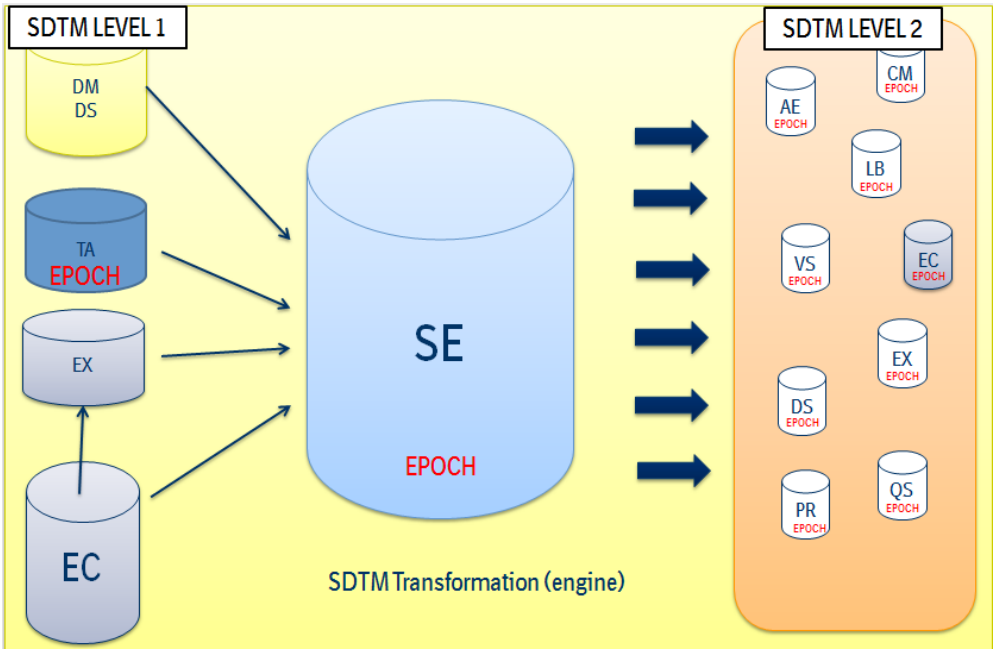
This technique allows querying a database from any place in the SAS executed code, without affecting the data handling involuntarily. This leads to dynamic and conditional programming in a very efficient way.

METADATA IN SDTM

A SDTM transformation engine has been developed in order to have all legacy patient data converted to SDTM at the beginning of a clinical trial. This approach consists of mapping all CRF fields (or external data) to SDTM format, including the non-required information by SDTM. The rationale is to be able to consider SDTM data as a unique source for any further task (ADaMs, adhoc output packages, data quality review …). This guarantees traceability, but it produces more data in SUPP and FA domains. All collected data is present in SDTM, restriction as per regulatory agencies requirements in terms of submission is still possible during the export process.

In this approach, the SDTM conversion is split in two steps:

1- SDTM mapping, i.e. data with a one-to-one relation between a raw data value in the CRF and a SDTM value
2- SDTM derivations once all domains are created, facilitating the interaction between different domains (like the EPOCH variable that is derived first in SE and then populated in other domains)

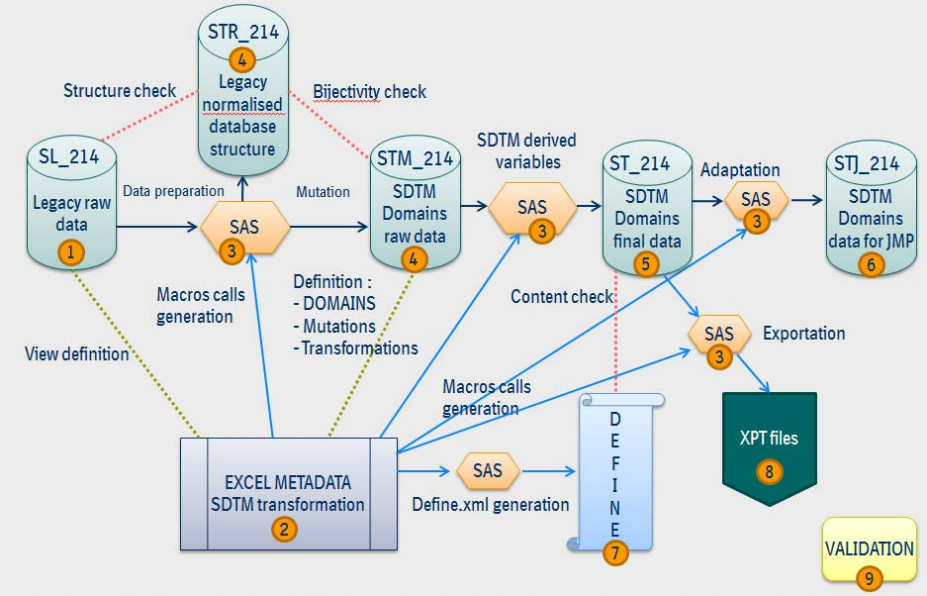


METADATA REPOSITORY (MDR)

Currently the tool "SDTM Plan" used to store the metadata and write the specifications for the SDTM datasets is in MS EXCEL ®. It could be adapted in the future to any MDR system.

The concept is to have one sheet per SDTM domain that will contain the domain structure and make the distinction between mapped variables and derived variables.
In these sheets, the derived variables are directly linked to derivation methods keywords. In terms of SDTM derivations, all available methods are centralized in one sheet. This sheet establishes the link between the keyword and the SAS macro which performs the actual derivation.

For each family of domains (SUPP, Findings, CO, Findings About …), a sheet holds the definition of the domain variables.
The tool is able to manage the Controlled Terminologies and their different versions.



Once the SDTM plan is correctly filled, a SAS macro generates all the needed programs to perform the SDTM conversion. As a result, two generated SAS programs are created, one program per domain for the mapping, and one global program for all derivations. These programs contain a successive list of macro calls.

Extraction of the DM domain sheet:

Domain Name	Variable	Label	Type	Length	Format	Class	Conversion Rule
DM	STUDYID	Study Identifier	Text	20		Source	
DM	DOMAIN	Domain Abbreviation	Text	8		Derived Level 0	RULE_DOMAIN
DM	USUBJID	Unique Subject Identifier	Text	20		Source	
DM	SUBJID	Subject Identifier for the Study	Text	20		Source	
DM	RFSTDTC	Subject Reference Start Date/Time	Datetime	20	ISO 8601	Derived Level 1	RULE_FIRST_DRUG_INTAKE
DM	RFENDTC	Subject Reference End Date/Time	Datetime	200	ISO 8601	Derived Level 1	RULE_TRIAL_COMPLETION
DM	RFXSTDTC	Date/Time of First Study Treatment	Datetime	20	ISO 8601	Derived Level 1	RULE_FIRST_DRUG_INTAKE
DM	RFXENDTC	Date/Time of Last Study Treatment	Datetime	20	ISO 8601	Derived Level 2	RULE_LAST_DRUG_INTAKE

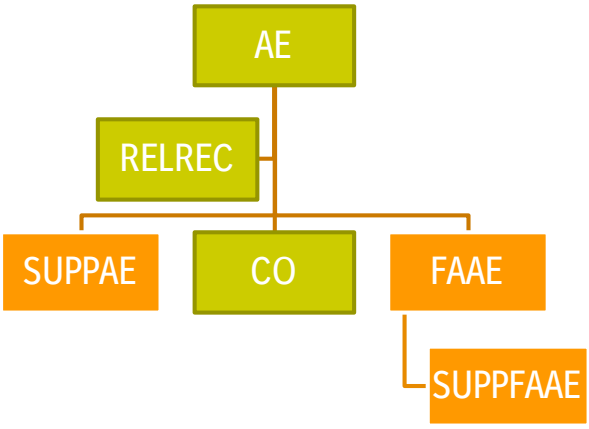
MAPPING FROM SOURCE DATA TO SDTM

As a pre-requisite, there is a sheet that makes the preparation of legacy datasets to be used by the different actions (transposition, merge, rename, drop …). The purpose is to adapt the structure of the legacy datasets in order to have a one-to-one relationship between one prepared legacy variable and one SDTM variable; the goal is to obtain a bijection between one raw value and one SDTM value. The bijection means that we have only one unique path between one point to another, here from one legacy raw value to a SDTM value.

The basic approach to map any source data to SDTM data is named an ACTION. One domain creation becomes a succession of several independent actions for the mapping part, independently from the order of execution.

The condition to use such an approach is to put in place a mechanism which conserves the validity of the correct keys in each domain with their respective sub-domains in order to be able to link the correct patient data together.

Example of AE domain dependencies:



DERIVATION OF SDTM VARIABLES AND DOMAINS

Once all domains have been created, the derivation macros are executed in order to finalize the SDTM creation. All required derivations are specified in the SDTM plan.

Due to the interdependency of the derivations, an execution order is needed in order to allow a robust implementation and avoid duplications.

All macros responsible for the implementation of the various methods are defined on a standardized way in the excel sheet (macro name, location and parameters). Another sheet contains the link between the derivation method defined by a keyword, and the corresponding macro call to perform the derivation.

This approach allows updating the two metadata sheets without modifying the programs that interpret this metadata. New methods are implemented as additive rows in the metadata repository, and the creation of the corresponding macro. If a new SDTM variable uses an existing method, only the assignment of that method in the domain sheet is needed.

DOCUMENTATION AND EXPORTATION

The define.xml is generated automatically at each conversion.
The .XPTs are also generated at the same time and the Pinnacle 21 checks are performed.

This allows frontloading for future submissions and adaptions of the data cleaning as per CDISC requirements.
Some automatic checks are also put in place to make sure that the SDTM conversion is done correctly and no data is lost during the conversion, this is the bijectivity check.

METADATA IN ADAM

Currently the tool “Analysis Dataset (ADS) Plan” that is used to store the metadata and write the specifications for the ADaM datasets is in MS EXCEL. It can be adapted in the future to any MDR system.

The concept is to have one sheet per ADaM dataset that will contain the domain structure and derivation rules for the derived variables.

METADATA REPOSITORY

The define.xml is generated automatically out of the ADS plan and becomes the source for the creation of any ADaM dataset. This is to ensure that the specifications are always in line with the content.
The ADS plan is also used to store the metadata information used to implement any endpoint in the BDS datasets.
The SAS programs used to do so are also automatically generated.
Let’s start with a basic example: the mapping of some parameter values stored in the SDTM domain TS to a SAS variable defined in the ADaM dataset ADTRIALS. This is similar to a Proc Transpose, the difference being that each parameter can have a different grouping and algorithm of transformation.

The goal in the following example is to retrieve one value per trial for each parameter.
This specification can go directly in an automated way to the Analysis Data Reviewer’s Guide.
SPTRIALS metadata sheet used to make the link between SDTM domain TS and ADaM ADTRIALS:

Name	*ADSVAR*	*TSPARMCD*	*TSGRPID*	*VALTYPE*
Label	*Variable of ADTRIALS*	*Parameter Code of TS*	*Group ID*	*Conversion*
Type	*text*	*text*	*text*	*text*
Format	*$8.*	*$8.*	*$40.*	*$20.*
Case	*UPPER*	*UPPER*	*UPPER*	*UPPER*
Context (Where condition)				
	DBLDT	DCUTDTC	DBLOCK	DATE
	SNAPMED	MEDVER		CHAR4
	SNAPWHO	WHOVER		CHAR6
	TPHASELG	TPHASE		CHAR50
	CNTRIES	FCNTRY		CHAR200
	TITLE	TITLE		CHAR200
	INTMODEL	INTMODEL		CHAR200
	TBLIND	TBLIND		CHAR200
	TCNTRL	TCNTRL		CHAR200
	RANDOM	RANDOM		CHAR10
	SEXPOP	SEXPOP		CHAR10
	AGEMIN	AGEMIN		YEARS
	AGEMAX	AGEMAX		YEARS
	PLANENR	PLANSUB		INTEGER
	FVFPATDT	SSTDTC		DATE
	TSTOPDT	CTARDTC		DATE
	EXTFL	EXTTRL		CHAR1
	PROJID	PROJNUM		CHAR10
	WOTIME	REPDUR		DAYS

```
data ADTRIALS;
    * variables from ads plan;
    set adtrials_struct studies;
    * variables from metadata sheet;
    %let tmpfmt = %VarNameFormat(&mxsetup.,ADSVAR);
    format ADSVAR &tmpfmt;
    %let tmpfmt = %VarNameFormat(&mxsetup.,VALTYPE);
    format VALTYPE &tmpfmt;
    %let tmpfmt = %VarNameFormat(&mxlibsrc..ts,TSGRPID);
    format TSGRPID &tmpfmt;
    %let tmpfmt = %VarNameFormat(&mxlibsrc..ts,TSPARMCD);
    format TSPARMCD &tmpfmt;
    * TSVALy variables;
    %let tmpfmt     = %VarNameFormat(&mxlibsrc..ts,TSVAL);
    %let list_tsval = %GetListVarPattern(&mxlibsrc..ts,TSVAL99,,Y);
    format  &list_tsval. &tmpfmt;
    array tabtsval[*] &list_tsval.;

    if _n_=1 then do;
        * HASHCODE HHTS linked to SDTM domain TS
          ........;
        * HASHCODE HHSPTRIALS linked to SPTRIALS metadata sheet
          ........;
    end;

    * For each ADSVAR;
    %do i=1 %to &end.;
        %let variable = %scan(&varfromts.,&i.,%str( ));

        ADSVAR="&variable.";
        * Retrieve TSPARMCD/TSGRPID/VALTYPE for the requested ADSVAR using HHSPTRIALS
          ........;
        * Retrieve TSVAL for the requested TSPARMCD using HHTS
          ........;
        * Apply the correct conversion using VALTYPE;
        if valtype='DATE' then do;
            call is8601_convert('dt','dt',tsvalresp,tmpdatetime);
            &variable.=datepart(tmpdatetime);
        end;
        * ........;
    %end;

run;
```

METADATA IMPLEMENTATION

Most of the programs are developed in a generic manner, so the metadata is a way to keep flexibility in various analyses and centralize similar processes as much as possible.

Some metadata datasets are stored as permanent ADaMs, when others are just created temporarily during the generation of other ADaM datasets.
There are two categories of metadata datasets:

A) Metadata ADaMs : global datasets that contains attributes (study or patient level)
These datasets can be used by all subsequent ADaMs; they are defined in MTxxxx sheets, and automatically replicated to the main sheet as 'Assigned' variables, in order to be extracted from the define.xml during the creation of the ADS.

	Context (Where condition)	GARMCD
1199_0214	ARMCD EQ "E"	B300T
1199_0214	ARMCD EQ "F"	P300T
1199_0214	ARMCD EQ "DBLUE"	DBLUE
1199_0214	ARMCD EQ "DRED"	DRED
1199_0214	ARMCD EQ "SCRNFAIL"	SCRNFAIL
1199_0214	ARMCD EQ "SCRNONG"	SCRNONG
1199_0214	ARMCD EQ "NOTASSGN"	NOTASSGN

Dataset Name	Context (Where condition)	Variable Name	Variable Label	Origin	Source / Derivation
ADTARM	*ALL*	STUDYID	Study Identifier	Predecessor	TS.STUDYID
ADTARM	*ALL*	SNAPSHOT	Snapshot Name	Predecessor	TS.TSVAL [TSPARMCD = 'SNAPSHOT']
ADTARM	*ALL*	SNAPDT	Snapshot Date	Predecessor	TS.TSVAL [TSPARMCD = 'SNAPDT']
ADTARM	*ALL*	ARMCD	Planned Arm Code	Predecessor	TA.ARMCD
ADTARM	*ALL*	ARM	Description of Planned Arm	Predecessor	TA.ARM
ADTARM	*ALL*	GARMCD	Global Arm Code	Assigned	
ADTARM	ARMCD EQ "E"	GARMCD	Global Arm Code	Assigned	B300T
ADTARM	ARMCD EQ "F"	GARMCD	Global Arm Code	Assigned	P300T
ADTARM	ARMCD EQ "DBLUE"	GARMCD	Global Arm Code	Assigned	DBLUE
ADTARM	ARMCD EQ "DRED"	GARMCD	Global Arm Code	Assigned	DRED
ADTARM	ARMCD EQ "SCRNFAIL"	GARMCD	Global Arm Code	Assigned	SCRNFAIL
ADTARM	ARMCD EQ "SCRNONG"	GARMCD	Global Arm Code	Assigned	SCRNONG
ADTARM	ARMCD EQ "NOTASSGN"	GARMCD	Global Arm Code	Assigned	NOTASSGN

B) Setup metadata datasets: contains algorithm specifications
These datasets are defined in SPxxxx sheets, and read via a generic external macro, in order to generate a SAS program responsible of the creation of the dataset version of this metadata information.
They are used as setup datasets during the creation of ADaMs.

```
data SETUP_SPTRIALS;
   attrib     ADSVAR         label="Variable of ADTRIALS"   format=$8.
              TSPARMCD       label="Parameter Code of TS"   format=$8.
              TSGRPID        label="Group ID"               format=$40.
              VALTYPE        label="Conversion"             format=$20.
   ;
   call missing(ADSVAR,TSPARMCD,TSGRPID,VALTYPE);
   delete;
run;

proc sql;
   insert into SETUP_SPTRIALS
   values (
   'DBLDT',
   'DCUTDTC',
   'DBLOCK',
   'DATE'
   )
```

DOCUMENTATION AND EXPORTATION

As already said the define.xml is systematically created right from the start and becomes the unique source of metadata for the ADaMs creation. As a consequence, this document is created before the ADaMs themselves rather than after.
The Analysis Data Reviewer's Guide contains partially derived information like SPxxxx sheets based on the ADS plan. The .XPTs are generated whenever it is needed in accordance with Pinnacle 21's checks.

METADATA IN DISPLAYS

According to the latest CDISC standards, the Analyses Results Metadata (ARM) will become more and more important in future submissions.
Our goal is to have one ARM per display. The ARM covers the regulatory agencies requirements in terms of traceability and also some useful metadata to automatically generate the SAS programs.

METADATA REPOSITORY

Currently a tool “Table of Contents (TOC) generator” used to store the metadata and manage titles and output attributes (linked with the SAS program) is in MS EXCEL. It could be adapted in the future to any MDR system.

There are three types of metadata information related to:

- unique display template
- actual output (a unique display template linked to a title and a set of ADSL variables / BDS endpoints) which consists of defining program name and output number
- statistics (descriptive or inferential calculation)

Here is the link between the title, the display template and the program responsible of generating the output.

Appendix	Number	Section	Type	Title 1	Program	Outputno	Display Template
15							
15	1	TRIAL PATIENTS					
15	1.1		Table	Disposition of patients	dispo.sas	1	EOT-DISP1
15	2	EFFICACY EVALUATION					
15	3	SAFETY EVALUATION					
15	3.1	Adverse events					
15	3.1.1		Table	Frequency [N (%)] of patients with adverse events - TS	ae.sas	1	XAE-1
15	3.2	Vital signs - physical findings and other relevant observations related to safety					
15	3.2.1		Table	Descriptive statistics and change from baseline over time - TS	vs.sas	1	EOT-DESC1

ARM VS DISPLAYS

ARM will be used for two main topics:

1- Automatically create the ARM section in the define.xml V2 for ADaMs
2- Dynamically generate a part of SAS macro calls for displays

The key elements need to be entered that link the actual display metadata information to the corresponding part of the program generator metadata information.

Progname	Outputno	Intermediate Display template	Display template
fvc.sas	1	RATE#MAIN	EOT-RATE1

The main information needed to automatically generate the macro call is:

Multiple exposure flag	Highest population flag	Treatment Analysis Number	Class	Optionnal ADSL replacement name	ADSL variables used	BDS endpoints	BDS where restriction
01	TRTFL	1000			NONE	ADVS.SYSBP[AVAL] / ADVS.SYSBP[CHG]	AVISIT = 52

The first part defines the ADSL variables needed; the second part specifies which restriction of the BDS will be used when applicable.

With this metadata information, this enables the automatic generation of the different macro calls that are responsible of doing:

1) Data building part
2) Analysis & reporting part

The data building part can be shared by multiple output creations, which means that the output generator macro is able to detect this automatically in order to optimize the program execution time.

DOCUMENTATION AND EXPORTATION

The ARMs will be present in the define.xml and will be synchronized with the displays due to the fact that the SAS programs are based on them.

Standardized results datasets are also exported from the analysis/reporting macros in order to be used for validation.

CONCLUSION

Metadata-driven programming is a powerful approach to minimize risks of inconsistencies across different packages and studies. It fits perfectly with a project centric approach dealing with multiple trials within the same project.

As an extension, an evaluation will be done on an approach that handles multiple therapeutic areas project within the same environment.

The consequence is that most of the programs should be developed in a generic way; a high level of control needs to be applied (USER ERRORs and WARNINGs in the log). Custom code or hardcoding should be avoided.

The development of generic SAS programs implies a good level of algorithmic and technical solutions like hashcode, extended attributes, arrays …

Once the system is in place and robust, metadata setup can be easily dispatched to several programmers. It leads to a fast and robust implementation of new BDS endpoints, easy process automation, natural and simpler oversight.

For SDTM conversion, this gives flexibility in terms of diversity in raw data without having different SAS programs.

For ADaMs creation, this reduces the time of implementation and facilitates the maintenance.

For displays, once the setup is done, there is a real improvement in productivity for the creation of new outputs, especially around the submission time when time pressure is high, or even later for post-marketing activities.

The metadata implementation creates somehow a synergy between the standardization and the flexibility which are two antagonist aspects in programming.

RECOMMENDED READING

http://www.cdisc.org/standards/foundational/sdtm

http://www.cdisc.org/standards/foundational/adam

http://www.cdisc.org/standards/foundational/define-xml

CONTACT INFORMATION

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

Omar SEFIANI / Stéphane BOUGET
Boehringer Ingelheim
12 rue André HUET
REIMS / 51100
Email: omar.sefiani@boehringer-ingelheim.com / stephane.bouget@boehringer-ingelheim.com

Brand and product names are trademarks of their respective companies.