PhUSE 2017

Paper PP16

Automatic creation of define.xml for ADaM: a fast way approach starting from ADaM Metadata

Dmitri Petratchenko, Valos, Genova, Italy
Andrea Parodi, Valos, Genova, Italy
Anna Romanova, Valos, Belarus

ABSTRACT

CDISC standards have become a must in submission of the documentation for clinical trials.
The define.xml is part of these standards and its creation can be very time-consuming.
This paper will show how we speed up the process of creation of define.xml for ADaM that will be automatically based on ADaM Metadata. With some additional instructions put on ADaM Metadata the user will save time in the creation of define.xml.

INTRODUCTION

The purpose of the define.xml is informing the reviewer (i.e. authority, CRO, external users) on what datasets, variables, controlled terms, and other specified metadata were used for building ADaM datasets.
Pinnacle 21 Community® creates define.xml starting from a detailed xls input file that is really time-consuming to be created. Valos created a SAS® macro that exploits the ADaM Metadata for speeding up the process of creation of this file (we will call it Excel Spec throughout this paper) through some instructions to be put in ADaM Metadata during its compilation.

ADAM METADATA

Below the list of variables needed in ADaM Metadata for letting it be successfully processed by our macro for creating Excel Spec in an easy and fast way.

- Dataset Name
- Parameter identifier
  Can take values 'Req' / 'Perm' / 'Cond'
- Variable Name
- Variable label
- Variable Type
  Can take values 'text' / 'integer'
- Display format
  a '$' followed by a number for text variables, a number or a data format for integers
- Controlled Term
  The list of values the variable can assume, separated by a comma
- Source/Derivation
  It is the algorithm describing the derivation of the variable (it will be deepen in the next subsections)
- Comments
  Filled with any type of information the user need to annotate on the variable.
- Codelist
- Origin

These last two columns deserve a detailed explanation, because they hold the key instructions for the macro.

In the "Codelist" column the user must specify just the name of the codelist for the variable. This will be processed then by the macro as detailed then in this paper.
The "origin" column with origin type information, can be filled with four different definitions:

- Predecessor: for variables pre-existing in an SDTM domain or in another ADaM dataset

- Assigned: for variables that are a classification of others like for example (from ADaM Metadata)

FIGURE 1

Dataset Name	Parameter identifier	Variable Name	Variable label	Variable Type	Display format	Codelist/Controlled Term	Source/Derivation	Comments	Codelist	Origin
ADSL	Perm	COUNTRY	Country	text	$3		DM.COUNTRY			Predecessor
ADSL	Perm	COUNTRYL	Country (Decode)	text	$50		One-to-one map to COUNTRY (according to C66786)		COUNTRY	Assigned
ADSL	Perm	COUNTRYN	Country (N)	integer	8.		Uniquely linked to COUNTRYL, numeric. Coded in alphabetical order		COUNTRYN	Assigned

- Derived: for variables derived from pre-existing variables by means of a specific algorithm
- Composite: the origin column is filled using the below structure

@<order number>@<origin>@<variable>@<comparator>@<condition>;

Where <order number> is a provisional index number for condition identification;
<origin> is a origin type for this condition;
<variable> is a variable which should has a condition;
<comparator> is a comparison operator (EQ, NE, GT, LT, GE, LE, IN, NOTIN);
<condition> is a condition for variable.

If condition is composite then it is separated to parts and for each of them assigned one common order number.

It is necessary each record has all 5 parts of this structure. If <variable> is a predecessor without any condition than NOTIN comparator value is used.

Below an example from ADaM Metadata:

FIGURE 2

Dataset Name	Parameter identifier	Variable Name	Variable label	Variable Type	Display format	Codelist/Controlled Term	Source/Derivation	Comments	Codelist	Origin
ADVS	Req	USUBJID	Unique Subject Identifier	text	$30		VS.USUBJID			Predecessor
ADVS	Req	PARAM	Parameter	text	$200	Diastolic Blood Pressure (mmHg), Systolic Blood Pressure (mmHg), Pulse Rate (BEATS/MIN), Height (cm),	For records with a corresponding record in VS, populate with the value of VS.VSTEST||(VS.VSSTRESU) For records created to contain the 'Body Mass Index-derived ' result, populate with PARAM= "Body Mass Index-derived (kg/m2)"		PARAM_ADVS	Derived
ADVS	Req	PARAMCD	Parameter Code	text	$8	DIABP, SYSBP, PULSE, HEIGHT, WEIGHT, BMID	@1@VS.VSTESTCD where One-to-one correspondence with PARAM For records with a corresponding record in VS ; @2@If PARAM= 'Body Mass Index-derived' result, then PARAMCD= "BMID";		PARAMCD_ADVS	@1@Predecessor@PARAM@ NOTIN@'Body Mass Index-derived',"Heart Rate Recovery"; @2@Derived@PARAM@EQ@ 'Body Mass Index-derived';
ADVS	Perm	PARAMTYP	Parameter Type	text	$200	DERIVED, null	Populated with "DERIVED" for records created to parameters 'Heart Rate Recovery' or 'Body Mass Index-derived', blank otherwise			Assigned

EDIT SOURCE/DERIVATION (STDM) INFORMATION

For variables with composite origin type, the structure must be:

@<order number>@<full condition description>;

Where <order number> is an index number that corresponds to <order number> in Origin Column and < full condition description> is a text with source/derivation information about each condition.

If in full condition description there is 'then' word, in Derivation column in define.xml file only the part of condition description after this word will be reported. This way the information presentation will be more readable.
In the example above on the variable PARAMCD, we can see the two conditions connected to Origin column:

@1@VS.VSTESTCD where One-to-one correspondence with PARAM
For records with a corresponding record in VS ;

@2@If PARAM="Body Mass Index-derived" result, then PARAMCD= "BMID";

AUTOMATIC CREATION OF INPUT FILE FOR PINNACLE 21 COMMUNITY®

Starting with the import of ADaM Metadata file the macro proceed step-by-step in the creation of each sheet that must be included in Excel Spec. Import file should be in xslx format.

STUDY

This sheet will contain the following information that must be put in input into the macro:

FIGURE 3

Attribute	Value
StudyName	CDISC-Sample
StudyDescription	CDISC-Sample Data Definition
ProtocolName	CDISC-Sample
StandardName	ADaM-IG
StandardVersion	1.0
Language	en

DATASETS

The first sheet in ADaM Metadata called "Analysis Dataset Metadata" contains this information on ADaM datasets:

- Dataset Name
- Dataset Description
- Dataset Location
- Dataset Structure
- Key Variables of Dataset
- Class of Dataset
- Source Data
- Comments
- Derive Order

This example below shows how two datasets (ADSL and ADAE) are presented in Excel Spec.

FIGURE 4

Dataset	Description	Class	Structure	Purpose	Key Variables	Repeating	Reference Data	Comment
ADSL	Subject-Level Analysis Dataset	ADSL	One record per USUBJID per STUDYID	Analysis	STUDYID, USUBJID	No	No	ADSL
ADAE	Adverse Event Analysis Dataset	ADAE	One record per each AE, per USUBJID, per STUDYID	Analysis	STUDYID, USUBJID, AEDECOD, AESEQ	No	No	

Repeating and Reference Data columns are set as "No" by default, while Comment column is filled with the ID of the comment in Comments sheet. This means that in Comments sheet the user can find:

FIGURE 5

ID	Description	Document	Pages
ADSL	Screen Failures are excluded since they are not needed for this analysis		

VARIABLES

The Variables sheet contains some information exactly as they were stored in ADaM Metadata and other derived. The information from ADaM Metadata are:

- Dataset
- Variable
- Label
- Data Type
- Format
- Codelist

Variable Origin is equal to Origin specified in ADaM Metadata when this is equal to 'Predecessor', 'Derived' or 'Assigned', otherwise it is left blank and the define.xml will get the information on the Origin from ValueLevel and WhereClauses sheets.

Variable Method contains the ID of the record the user have to check in sheet Methods to find the derivation method of a variable with Origin = 'Derived'.

Variable Predecessor contains the predecessor variable when Origin = 'Predecessor'.
Variable Comment contains the ID of the record the user have to check in sheet Comments to find the comment linked to the variable.

As example the figure below shows only the derived variables in this sheet (data are the same of FIGURE 2):

FIGURE 6

Dataset	Variable	Label	Origin	Method	Predecessor	Comment
ADVS	USUBJID	Unique Subject Identifier	Predecessor		VS.USUBJID	
ADVS	PARAM	Parameter	Derived	ADVS.PARAM		
ADVS	PARAMCD	Parameter Code				
ADVS	PARAMTYP	Parameter Type	Assigned			ADVS.PARAMTYP

VALUE LEVEL
Explains the origin of the variables with composite origin type.
Where clause variable is used as ID in Where Clauses sheet.

The variable PARAMCD from dataset ADVS is considered in the following example:

FIGURE 7

Dataset	Variable	Where Clause	Origin	Method	Predecessor	Value Level Comment	Join Comment
ADVS	PARAMCD	ADVS.PARAM.NOTIN.'BodyMassIndex-derived',"HeartRateRecovery"	Predecessor	ADVS.PARAMCD.ADVS.PARAM.NOTIN.'BodyMassIndex-derived',"HeartRateRecovery"	VS.VSTESTCD		
ADVS	PARAMCD	ADVS.PARAM.EQ.'BodyMassIndex-derived'	Derived	ADVS.PARAMCD.ADVS.PARAM.EQ.'BodyMassIndex-derived'			

WHERE CLAUSES
Explains how the variables with composite origin type are derived.

The variable PARAMCD from dataset ADVS is considered in the following example:

FIGURE 8

ID	Dataset	Variable	Comparator	Value
ADVS.PARAM.EQ.'BodyMassIndex-derived'	ADVS	PARAM	EQ	'Body Mass Index-derived'
ADVS.PARAM.NOTIN.'BodyMassIndex-derived',"HeartRateRecovery"	ADVS	PARAM	NOTIN	'Body Mass Index-derived',"Heart Rate Recovery"

CODELISTS
This sheet explains in detail the codelists used.
The macro processes the codelist column in ADaM_Metadata checking if the codelist specified belongs to:

- CDISC Controlled Terminology extensible codelist
- CDISC Controlled Terminology not extensible codelist
- Non-standard CDISC codelist

and comparing the values found in the datasets with CDISC Terminology, notifying the user if some value is misspelled.
To create this sheet the macro reads ADaM Terminology.xls and SDTM Terminology.xls files (previously stored in the same folder of ADaM Metadata) together with Terms and Decoded Values taken from datasets.
The codelists with the same prefix are processed together to a code/number decoded value.
For example the non-standard codelists ARM, ARMN and ARMCD are automatically processed for obtaining in Excel Spec:

FIGURE 9

ID	Name	NCI Codelist Code	Data Type	Order	Term	NCI Term Code	Decoded Value
ARM	ARM		text	1	Active Treatment		Active Treatment
ARM	ARM		text	2	Not Randomized		Not Randomized
ARM	ARM		text	3	Placebo		Placebo
ARMCD	ARMCD		text	1	A		Active Treatment
ARMCD	ARMCD		text	2	NOTASSGN		Not Randomized
ARMCD	ARMCD		text	3	PBO		Placebo
ARMN	ARMN		text	1	1		Active Treatment
ARMN	ARMN		text	2	2		Placebo
ARMN	ARMN		text	3	3		Not Randomized

DICTIONARIES

This sheet will contain the following information that must be put into the macro:
ID is a coded term of variables taking values from the dictionary: these values end with 'DICT' and they are present in Codelists sheet.

FIGURE 10

ID	Name	Data Type	Dictionary	Version
AEDICT	Adverse Event Dictionary	text	MedDRA	8

METHODS

When a variable has Derived origin, the derivation method is described in this sheet. In figure 6 ADVS.PARAM has been chosen as example and here below we show how the derivation method of this variable appears in this sheet:

FIGURE 11

ID	Name	Type	Description
ADVS.PARAM	CM.ADVS.PARAM	Computation	For records with a corresponding record in VS, populate with the value of VS.VSTEST||(VS.VSSTRESU) VSSTRESU are in standart format: lb -> kg, in -> cm For records created to contain the 'Body Mass Index-derived ' result, populate with PARAM= "Body Mass Index-derived (kg/m2)"

COMMENTS

With the same logic of Methods sheet, the Comments sheet ID represents the identifier of the comment specified both for datasets and variables. When a dataset or a variable has a comment, this is described in this sheet.
In figures 4 and 6, two IDs for comments are reported: in figure 4 ADSL dataset and in figure 6 ADVS.PARAMTYP variable have been chosen as example and here below we show how the comments referred to these appear in this sheet:

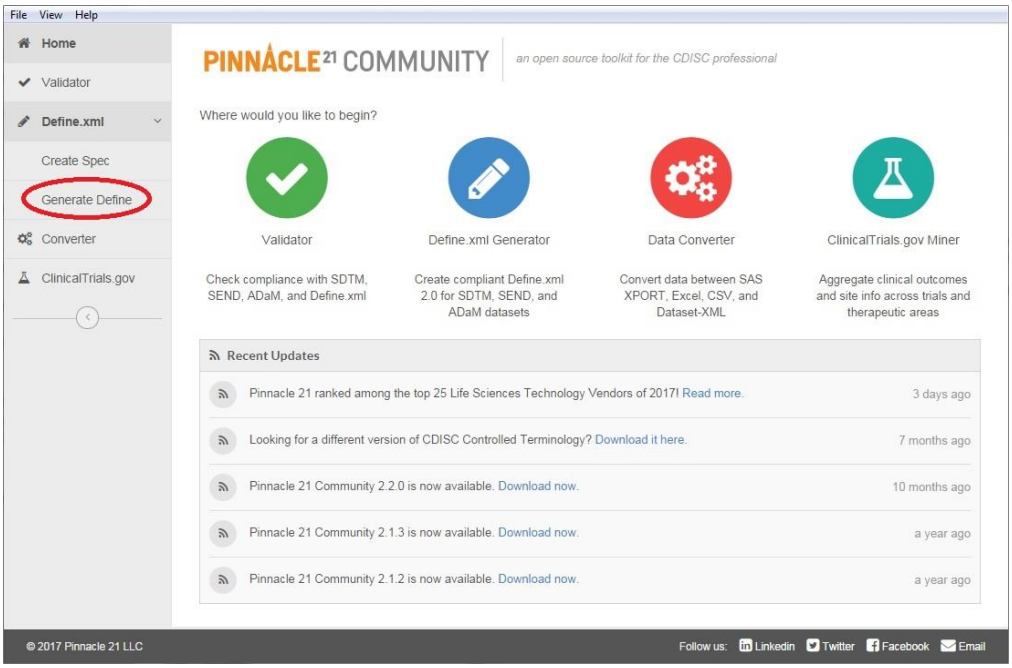
FIGURE 12

ID	Description
ADSL	Screen Failures are excluded since they are not needed for this study analysis
ADVS.PARAMTYP	Populated with “DERIVED” for records created to paramerers 'Body Mass Index-derived', blank otherwise

CREATION OF DEFINE.XML USING PINNACLE 21 COMMUNITY®:

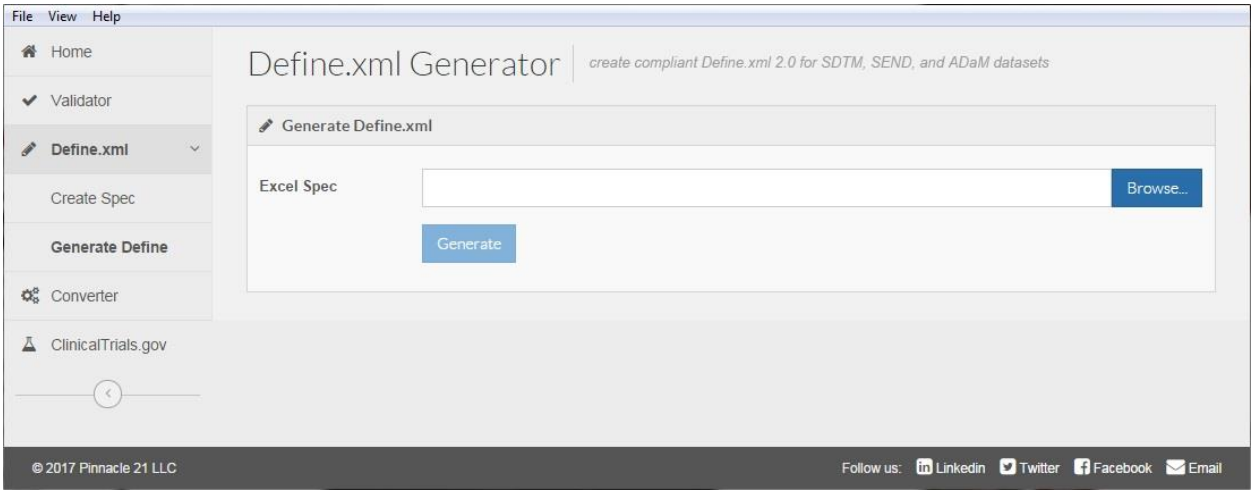
Once the ADaM Metadata has been compiled following the instructions and the macro has been run obtaining the Excel Spec, this file must be processed by Pinnacle 21 Community®:

FIGURE 13



Through the "Generate Define" button on the left column, the user is brought to the following screen where the Excel Spec must just be uploaded:

FIGURE 14

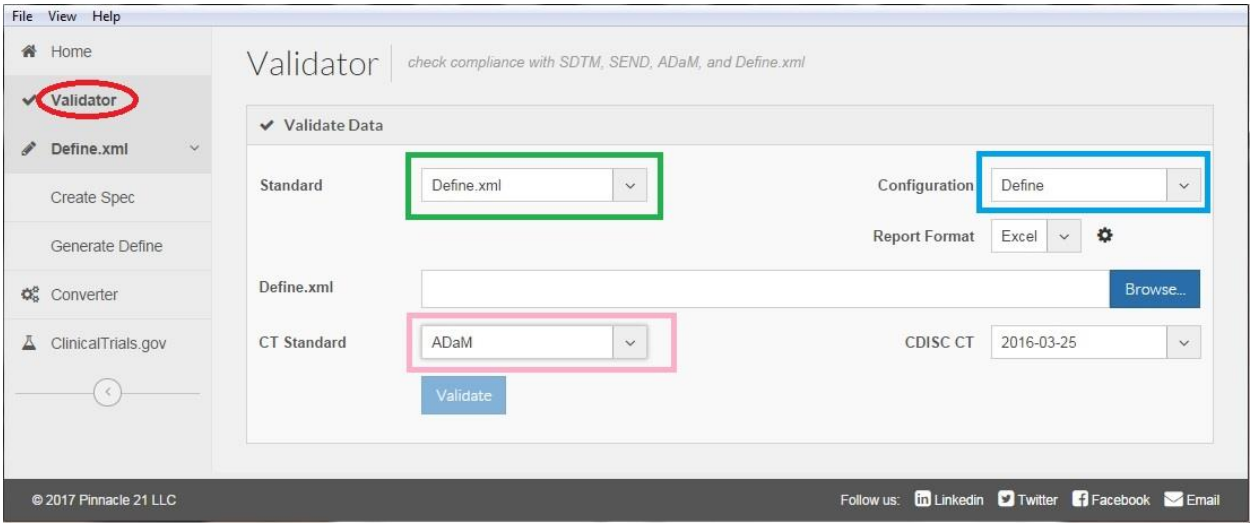


VALIDATION OF DEFINE.XML USING PINNACLE 21 COMMUNITY®:

The last step the user have to do for completing the process of creation of define.xml is to validate it using Pinnacle 21 Community®. Define.xml should be placed in the same folder with xpt files of datasets.

As shown in the example below, clicking on the button marked in red, the validator screen will appear. After selecting what validation (green), configuration (turquoise) and CT Standard (pink) to perform, the user can validate define.xml, understanding than if all previous steps described in this paper were done well.

FIGURE 15



CONTACT INFORMATION

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

Author Name	Dmitri Petratchenko
Company	Valos
Address	Via Ceccardi 4/31
City / Postcode	16121
Work Phone:	+39 010 407 7182
Fax:	+39 010 253 4160
Email:	dmitri.petratchenko@valos.it
Web:	www.valos.it