PharmaSUG 2016 - Paper 218

Up-Versioning Existing Define.xml from 1.0 to 2.0

Jeff Xia, Merck & Co., Rahway, NJ
Lugang Xie (Larry), Merck & Co., Rahway, NJ

ABSTRACT

As per the latest Standards Catalog released by the FDA, both define.xml 1.0 and 2.0 are acceptable for data submissions like NDA. However, because of the technical insufficiency of define.xml 1.0, the agency specifically encourages the industry to submit the define.xml of version 2.0. It is highly possible that multiple studies are included in a data submission. Some of them may have been done in early years with an older version of define.xml, while other studies may have implemented a newer version. To keep the define xml version consistent in a single submission, there is a need for generating newer version of define.xml for these old studies. Considering that significant amount of work had been invested to check the validity and compliance of these define files in older version, it is more efficient and cost effective to convert these define.xml from version 1.0 to 2.0 without losing information.

This paper briefly discusses the difference of define.xml between version 1.0 and 2.0, then introduces a simple approach to perform the define.xml up-versioning.

INTRODUCTION

The data definition file describes the metadata of the submitted electronic datasets, which is considered as the most important part of the electronic dataset submission for regulatory review. As per the most recent Standards Catalog (Version 4.4) released in August 2015, the agency accepts both the define.xml version 1.0 and 2.0 (hereafter called define.xml 1.0 and 2.0 respectively). However, there are some technical insufficiencies associated with the define.xml 1.0. For example, it is very difficult to print out a nicely formatted hard copy of define.xml 1.0. Therefore, the agency specifically requests the sponsor to submit a test version of the define.xml version 1.0 prior to application submission to ensure that it can be printed within the agency IT environment. Alternatively, a printable copy of define.pdf has to be submitted along with the define.xml 1.0. The Agency also clearly mentioned that “If a define.xml version 2.0 or later version is submitted, then a define.pdf does not need to be included in the submission”.

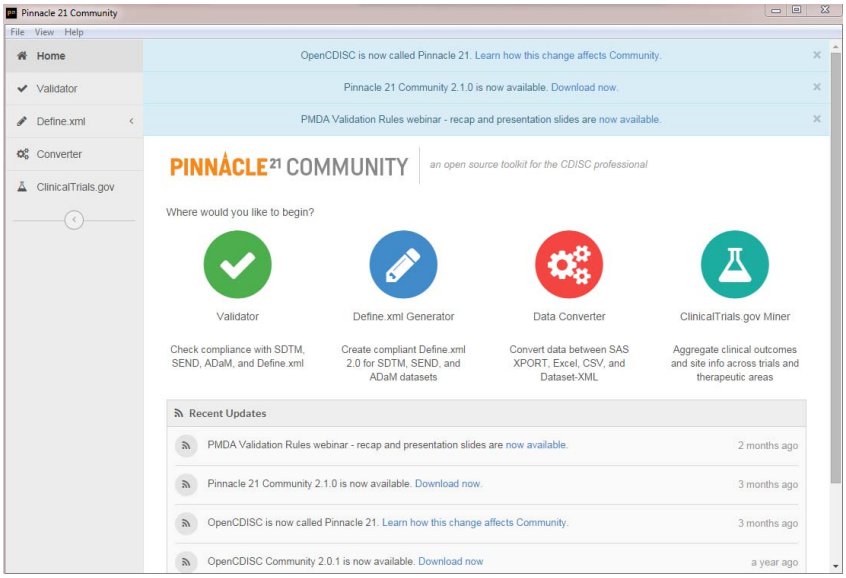
There are many new features and advantages of define.xml 2.0, such as it supports result metadata, removes redundant information from the Controlled Terminology (hereafter called CT) section by allowing omission of decode if it is the same as coded value, provide C-code for CDISC CT for a quicker reference, provides where clause for value level/parameter level metadata, etc. More and more sponsors/reviewers are in favor of this newer version of define.xml for the study data for recent or ongoing studies. On the other hand, many earlier studies might have an older version of define.xml, which had been carefully reviewed/validated. If they will be included in the same application along with the newer studies, the agency expects a consistent version of define.xml.

MAIN DISCUSSION

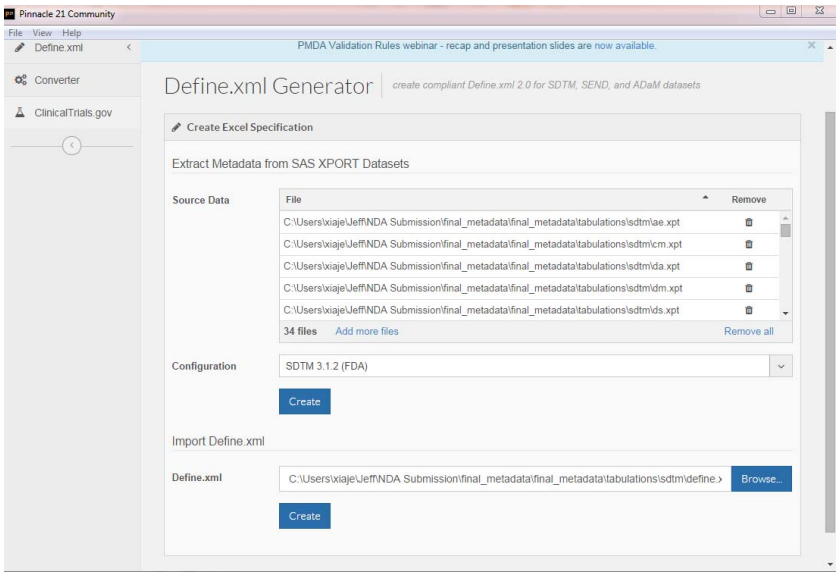
Pinnacle 21 (previously well-known as OpenCDISC) has been widely utilized in the pharmaceutical industry to validate clinical trial data. The most recent release is Pinnacle 21 Community version 2.1.0, which provides 3 valuable components: Validator, Define.xml Generator and Converter. The Validator could be used to validate ADaM, SDTM, SEND datasets as per data checking rules published by the FDA and validate define.xml as per XML schema developed by CDISC. Meanwhile, the Define.xml Generator can be used to create data spec in the format of MS Excel spreadsheet and generate define.xml 2.0 based on the updated spreadsheet. The following steps briefly describe how to convert an existing define.xml from version 1.0 to 2.0, as well as the way to validate and enhance it.

STEP 1. CONVERT EXISTING DEFINE.XML VERSION 1.0 TO DATA SPEC

1. Start Pinnacle and select “Define.xml” in the main menu on the left. “Create Spec” and “Generate Define” menu items will appear under “Define.xml”.
2. Select “Create Spec”, the interface “Define.xml Generator” appears on the right pane.
3. Select “Browse” in the “Source Data” section and import the study data in the format of SAS transport file.
4. Select the appropriate data model version from the drop-down menu next to “Configuration”. Available values include SDTM 3.1.1, 3.1.2, 3.1.3, 3.2, ADaM 1.0, SEND 3.0, etc.
5. Select “Browse” under “Import Define.xml” and import the existing define.xml in version 1.0

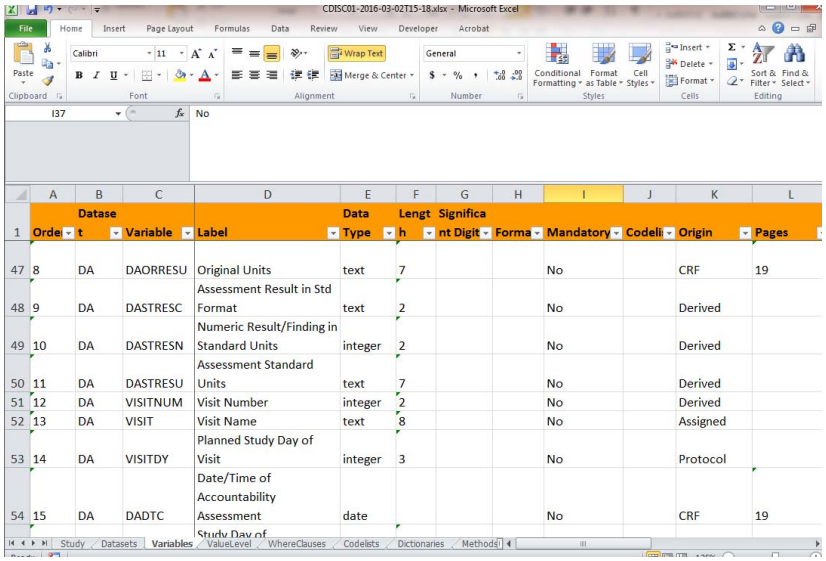


Display 1. Main User Interface of Pinnacle 21



Display 2. User Interface of Creating Data Spec

6. Select “Create” button to create the data spec. Please note that there are two “Create” buttons on the screen. The upper one is to create the data spec by only importing the metadata of the SAS transport files, whereas the lower one will create the spec by importing the metadata from data as well as the contents in the existing define.xml. In our case, select the lower “Create” button to read the contents in existing define.xml version 1.0.
7. A pop-up window “Generating Define.xml Excel Spec” appears. When the generation is done, select the “Open Spec” button to review the data spec.
8. The data spec in Excel contains the following tabs: Study, Datasets, Variables, ValueLevel, WhereClause, Codelists, Dictionaries, Methods, Comments and Documents.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Order	Dataset	Variable	Label	Data Type	Length	Significant Digits	Format	Mandatory	Codelist	Origin	Pages
47	8	DA	DAORRESU	Original Units	text	7			No		CRF	19
48	9	DA	DASTRESC	Assessment Result in Std Format	text	2			No		Derived	
49	10	DA	DASTRESN	Numeric Result/Finding in Standard Units	integer	2			No		Derived	
50	11	DA	DASTRESU	Assessment Standard Units	text	7			No		Derived	
51	12	DA	VISITNUM	Visit Number	integer	2			No		Derived	
52	13	DA	VISIT	Visit Name	text	8			No		Assigned	
53	14	DA	VISITDY	Planned Study Day of Visit	integer	3			No		Protocol	
54	15	DA	DADTC	Date/Time of Accountability Assessment	date				No		CRF	19

Display 3. Sample output of data spec

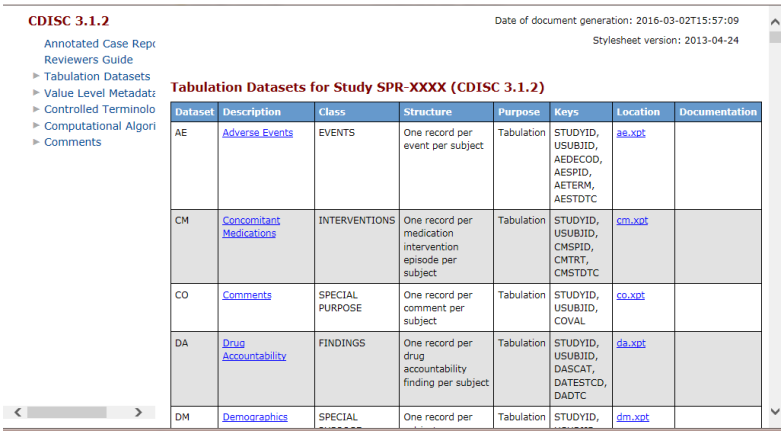
STEP 2. UPDATE THE DATA SPEC FOR DEFINE.XML 2.0

The application does a very good job in retrieving the meta data from each individual dataset, and additional information from the existing define.xml, such as the comments for each variable in the datasets, names of the code list, values of Origin (i.e., CRF Page number), etc. However extra steps have to be performed in order to convert it to a well-defined define.xml 2.0.

1. In the Codelists tab, the “NCI Codelist Code” and “NCI Term Code” columns have to be updated. A small SAS utility program can be developed to retrieve the code from the Controlled Terminology (CT) text file, which can be downloaded from the website of National Cancer Institute. Depending on the version of the CT used in the trial, the program should be able to retrieve the corresponding Term code and Codelist code if there is a match.
2. Save the updated data spec in a location at your preference.

STEP 3. CONVERT THE DATA SPEC TO DEFINE.XML 2.0

1. Select “Generate Define” under Define.xml in the main menu. “Define.xml Generator” window appears on the right.
2. Select “Browse” to import the updated data spec in Excel spreadsheet format.
3. Select “Generate” to create define.xml 2.0. When it is done, Select “Open Define.xml” to browse the output.



CDISC 3.1.2

Date of document generation: 2016-03-02T15:57:09

Stylesheet version: 2013-04-24

Annotated Case Repo
Reviewers Guide
Tabulation Datasets
Value Level Metadata
Controlled Terminolo
Computational Algori
Comments

Tabulation Datasets for Study SPR-XXXX (CDISC 3.1.2)

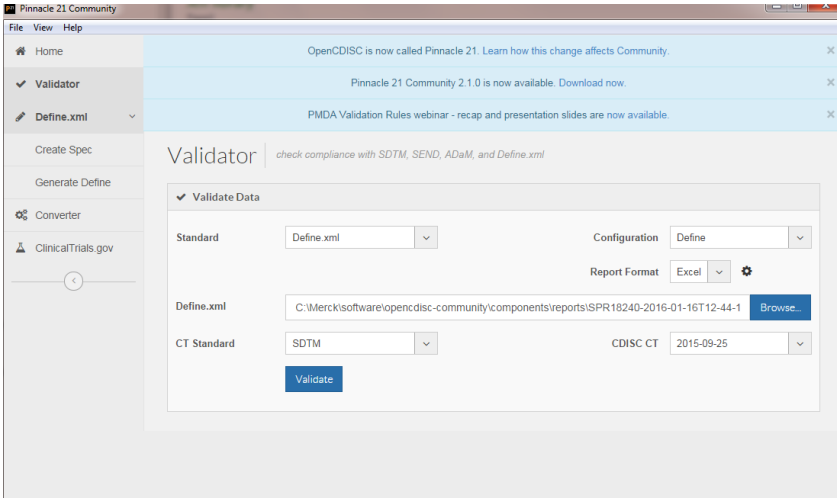
Dataset	Description	Class	Structure	Purpose	Keys	Location	Documentation
AE	Adverse Events	EVENTS	One record per event per subject	Tabulation	STUDYID, USUBJID, AEDECOD, AESPID, AETERM, AESTDTC	ae.xpt	
CM	Concomitant Medications	INTERVENTIONS	One record per medication intervention episode per subject	Tabulation	STUDYID, USUBJID, CMSPID, CMTRT, CMSTDTC	cm.xpt	
CO	Comments	SPECIAL PURPOSE	One record per comment per subject	Tabulation	STUDYID, USUBJID, COVAL	co.xpt	
DA	Drug Accountability	FINDINGS	One record per drug accountability finding per subject	Tabulation	STUDYID, USUBJID, DASCAT, DATESTCD, DADTC	da.xpt	
DM	Demographics	SPECIAL	One record per	Tabulation	STUDYID,	dm.xpt	

Display 4. Sample output of define.xml 2.0 generated using Pinnacle 21

STEP 4. VALIDATING DEFINE.XML

It is necessary to validate the generated define.xml 2.0 to ensure it is compliant with CDSIC published define.xml schema.

11. Select “Validator” in the main menu, and select “Define.xml” from the Standard drop-down menu.
12. Select “Browse” to locate the generated define.xml version 2.0. By default it is located in the subfolder “Reports” where the Pinnacle 21 is installed.
13. Select “Validate”.
14. Select “Open Report” to review findings if any.



Display 5. Using Pinnacle 21 to validate the output of define.xml 2.0

DISCUSSION

The define.xml 2.0 generated by this application is great except one thing: the domains displayed in the TOC section are sorted in alphabetical order, which is inconsistent with the example define.xml provided by CDISC and does not meet the expectation of agency reviewers. The order of domains in TOC recommended by SDTM Metadata Submission Guidelines (SDTM-MSG) is Trial Design, Special Purpose, Intervention, Events, Findings and Special Purpose. Even if the order in the input data spec is changed to the order recommended by SDTM-MSG, the datasets will still be displayed in alphabetical order in the generated define.xml, which means that the end user of the application has no way to change the display order of domains in the TOC.

To work around this issue, a small SAS utility program has been developed to re-arrange the order of the domains displayed in the TOC and DDT sections of define.xml. The basic idea is to read in the output define.xml to a SAS dataset, re-arrange the order of the XML definitions of tabulation datasets (ItemGroupDefinition) based on their Class first and then by alphabetical order of the domain names within each class. See appendix for the complete SAS code.

CONCLUSION

The Pinnacle 21 has been a great tool to evaluate the CDISC compliance for both datasets and data definition documents. Furthermore the newly released version of the application provides the capability to generate data spec based on metadata of the datasets as well as to create define.xml 2.0 based on the updated data spec. By combining these two great capabilities, this tool can be used to convert the existing define.xml from 1.0 to 2.0 with minimum programming effort involved.

RECOMMENDED READING

1. Study Data Technical Conformance Guide
http://www.fda.gov/downloads/ForIndustry/DataStandards/StudyDataStandards/UCM384744.pdf

2. Sergiy Sirichenko, Michael DiGiantomasso, Travis Collopy. 2015. Usage of OpenCDISC Community Toolset 2.0 for Clinical Programmers. PharmaSUG 2015.
3. XML Schema Validation for Define.xml. http://www.cdisc.org/system/files/members/standard/definereport_v1_0.pdf

ACKNOWLEDGMENTS

The authors would like to thank Cynthia He for her great support and valuable input of this paper.

CONTACT INFORMATION

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

Name: Jeff Xia
Enterprise: Merck
Address: 126 E. Lincoln Avenue
City, State ZIP: Rahway, NJ 07065-4607
Work Phone: 732-594-6439
Fax:
E-mail: jeff.xia@merck.com
Web: www.merck.com
Twitter:

Name: Lugang Xie (Larry)
Enterprise: Merck
Address: 126 E. Lincoln Avenue
City, State ZIP: Rahway, NJ 07065-4607
Work Phone: 732-594-1527
Fax:
E-mail: lugang.xie@merck.com
Web: www.merck.com
Twitter:

APPENDIX

```
%macro sortxml(infile=,outfile=,debug=) / minoperator;

    data xml(keep=xmlcode sect rec_ord) domains(keep=class_ord sect domain);
        length domain $20 class $50 xmlcode $8196;
        infile "&infile" lrecl=8196;
        input;
        xmlcode= infile_;
        rec_ord+1;

        retain sect 1 igdef_st igdef_en domain_found class_found 0 domain;

        ** Each tabulation dataset is enclosed with
           <ItemGroupDef> and </ItemGroupDef> **;
        if substr(left(xmlcode),1,14)='<ItemGroupDef ' then do;

            ** sect=1 for everything before the first <ItemGroupDef> **;
            ** sect=2,3... for tabulation datasets **;
            ** sect=sect of the last dataset + 1 for
               everything after tabulation datasets **;
            sect+1;

            igdef_st=1;
            if igdef_en=1 then igdef_en=0;
        end;
        else if substr(left(xmlcode),1,15)='</ItemGroupDef>' then igdef_en=1;
        else if igdef_en=1 then do;
            sect+1;
            igdef_en=0;
        end;
        else if igdef_st=1 and find(trim(xmlcode),'>')>0 then do;
          igdef_st=0;

          ** igdef_tag_en=1 if the closing > for <ItemGroupDef> is hit. **;
          igdef_tag_en=1;
        end;

        ** Retrieve domain names **;
        if (igdef_st=1 or igdef_tag_en=1) and domain_found=0 then do;
            domain_pos=find(trim(xmlcode),'Name=');
            if domain_pos then domain= substr(xmlcode, domain_pos+6,
find(upcase(trim(xmlcode)), ' ',domain_pos) - domain_pos-7);
            if domain_pos then domain_found=1;
        end;

        ** Retrieve class names and code classes **;
        if (igdef_st=1 or igdef_tag_en=1) and class_found=0 then do;
            class_pos=find(trim(xmlcode),'def:Class=');
            if class_pos then do;
                  class=dequote(substr(xmlcode,class_pos+10));
                  class_found=1;

                ** Code classes in the order recommended in the SDTM-MSG **;
                  if class='TRIAL DESIGN'    then class_ord=1;
                  else if class='SPECIAL PURPOSE' then class_ord=2;
                  else if class='INTERVENTIONS'   then class_ord=3;
```

```
                else if class='EVENTS'          then class_ord=4;
                else if class='FINDINGS'        then class_ord=5;
                else if class='FINDINGS ABOUT'  then class_ord=6;
                else if class='RELATIONSHIP'    then class_ord=7;

                else if class='SUBJECT LEVEL ANALYSIS DATASET'
                            then class_ord=1;
                else if class='ADVERSE EVENTS ANALYSIS DATASET'
                            then class_ord=2;
                else if class='BASIC DATA STRUCTURE'
                            then class_ord=3;
                else if class='ADAM OTHER'
                            then class_ord=4;

                output domains;
            end;
        end;

          ** Reset domain_found and class_found to 0 for the next dataset **;
          if igdef_tag_en=1 then do;
              domain_found=0;
              class_found=0;
          end;

          output xml;
    run;

    ** Sort classes in the order recommended in the SDTM-MSG **;
    ** Sort tabulation datasets within a class in alphabetic order **;
    ** Everything else remains unchanged **;
    data xml1;
        merge xml(in=a) domains(in=b);
        by sect;
        if sect=1 then ord=1;
        else if b then ord=2;
        else ord=3;
    run;

    proc sort data=xml1 out=xml2;
        by ord class_ord domain rec_ord;
    run;

    ** Output to XML **;
    data  null ;
        file "&outfile" lrecl=8196 nopad;
        set xml2(keep=xmlcode);
        put xmlcode;
    run;

    ** Remove temporary datasets **;
    %if %upcase(&debug) in N NO 0 FALSE %then %do;
        proc datasets lib=work nolist;
            delete xml xml1 xml2 domains;
        quit;
    %end;
%mend sortxml;
```