Paper ###-2019

Automating ADSL Programming Using Pinnacle 21 Specifications

Tracy Sherman, Ephicacy Consulting Group Inc.;
Aakar Shah, Nektar Therapeutics.

ABSTRACT

As the use of Pinnacle 21 ADaM define specifications increases, so does the opportunity for automation. ADaM programming can be automated by using the Pinnacle 21 specifications to determine the necessary variable data source, assignments, codelists and derivations. As each variable has a specific origin type designated in the specifications such as predecessor, assigned, derived, etc., the specifications can be tailored and drawn upon to write the applicable SAS code to a program.

This paper will demonstrate how Pinnacle 21 specifications for ADSL can be used to generate a program for producing an ADSL data set. Specification guidelines based on origin type will be proposed to help streamline the specification writing process and strengthen the ability to automate data set programming.

INTRODUCTION

We support the idea that Pinnacle 21 STDM specifications should be the first step of data set programming, not the last (Shah and Sherman, 2018). If we start programming with the 'define in mind', this alleviates re-work when it comes to define specification writing. Other benefits of this approach include the ability to validate on an ongoing basis such as when new SDTM is received or when statisticians want to review data set specifications. If the define is available throughout, reviewers can simultaneously view the specifications with hyperlinked data and also get a sense of the data quality through the use of Pinnacle 21 validator. Too many times, CDISC validation issues in ADaM are the result of missing the CDISC validation in SDTM. If programming starts with SDTM specifications in the form that was used for the define specifications, then validation issues would be caught at the beginning and not crunch time when ADaM and TLF programming have already been started.

Not only should SDTM define specifications be written prior to SDTM programming but ADaM define specifications should also be the first step in ADaM programming. These specifications can be used to create the required define.xml for early review and validation, whilst also being available to automate ADaM programming.

This paper will demonstrate how the 'Variables' worksheet from ADaM Pinnacle 21 specifications and more specifically the 'Origin' column is used for each variable is declared. The Origin column has three main entries (Predecessor, Assigned, and Derived) along with less common entries such as eDT (electronic data transfer), Protocol and <missing>. These different origin types can be used to create generic SAS code using metadata from the 'Codelists' and 'Methods' worksheets.

Guidelines will be proposed for each origin type to streamline the specification writing process and for consistent data set automation programming.

ADSL PINNACLE 21 SPECIFICATIONS

If you do not have a copy of the Define specification spreadsheet used by Pinnacle 21, you could easily get it from the community version software by uploading any ADaM dataset. You can download the Pinnacle 21 Community software for no cost at https://www.pinnacle21.com/downloads. For more details on creating the specifications from source data please refer to https://www.pinnacle21.com/projects/using-opencdisc-community.

The Define specification workbook contains 10 worksheets: Study, Datasets, Variables, Valuelevel, Whereclauses, Codelists, Dictionaries, Methods, Comments, and Documents. We will focus on the Variables and Methods worksheets for this paper.

On the Variables worksheet (see Figure 1), the “Origin” column generally has three main entries (Predecessor, Assigned, and Derived) along with less common entries such as eDT (electronic data transfer) and <missing>.

	Order	Dataset	Variable	Label	Data Type	Length	Significant Digits	Format	Mandatory	Codelist	Origin	Pages	Method	Predecessor	Role	Comment
2	1	ADSL	STUDYID	Study Identifier	text	15			Yes		Predecessor			DM.STUDYID	Identifier	
3	2	ADSL	USUBJID	Unique Subject Identifier	text	25			Yes		Predecessor			DM.USUBJID	Identifier	
4	3	ADSL	SUBJID	Subject Identifier for the Study	text	9			Yes		Predecessor			DM.SUBJID	Identifier	
5	4	ADSL	SITEID	Study Site Identifier	text	4			Yes		Predecessor			DM.SITEID	Identifier	
6	5	ADSL	ARM	Description of Planned Arm	text	11			Yes	ARM	Predecessor			DM.ARM	Treatment	
7	6	ADSL	TRT01P	Planned Treatment for Period 01	text	11			Yes	ARM	Predecessor			DM.ARM	Treatment	
8	7	ADSL	TRT01PN	Planned Treatment for Period 01 (N)	integer	8			No	ARMN	Assigned				Treatment	ADSL.TRT01PN
9	8	ADSL	TRT01A	Actual Treatment for Period 01	text	11			No	ARM	Predecessor			DM.ACTARM	Treatment	
10	9	ADSL	TRT01AN	Actual Treatment for Period 01 (N)	integer	8			No	ARMN	Assigned				Treatment	ADSL.TRT01AN
11	10	ADSL	COHORT	Cohort Assigned	text	8			No	COHORT	Predecessor			SUPPDM.QVAL where QNAM="COHORT"	Descriptive	
12	11	ADSL	COHORTN	Cohort Assigned (N)	integer	8			No	COHORTN	Assigned				Descriptive	ADSL.COHORTN
13	12	ADSL	STAGE	Stage	text	7			No	STAGE	Predecessor			SUPPDM.QVAL where QNAM="STAGE"	Descriptive	
14	13	ADSL	STAGEN	Stage (N)	integer	8			No	STAGEN	Assigned				Descriptive	ADSL.STAGEN
15	14	ADSL	TRTSDT	Date of First Exposure to Treatment	integer	8		date9.	No		Derived		MT.ADSL.TRTSDT		Trial Date	
16	15	ADSL	TRTEDT	Date of Last Exposure to Treatment	integer	8		date9.	No		Derived		MT.ADSL.TRTEDT		Trial Date	

Figure 1. ADSL Pinnacle 21 specifications highlighting the main origin types

GUIDELINES FOR SPECIFICATIONS BASED ON ORIGIN TYPE

On the Variables worksheet, one of the key columns used to automate ADSL programming is the “Origin” column. In Table 1, each origin type is described. The first three origins in the table below are used in the macro, %M_ORIGIN, to generate data step SAS code in adsl.sas.

Origin	Description
Predecessor	Data that is copied from a variable in another dataset. For example, predecessor is used to link ADaM data back to SDTM variables or other ADaM variables to establish traceability. If selected, the Define will include text from "Predecessor" column.
Derived	Data that is not directly collected on the CRF or received via eDT, but is calculated by an algorithm or reproducible rule defined by the sponsor, which is dependent upon other data values. Use with "Method" column.
Assigned	Data that is determined by individual judgment (by an evaluator other than the subject or investigator), rather than collected as part of the CRF, eDT or derived based on an algorithm. This may include third party attributions by an adjudicator. Coded terms that are supplied as part of a coding process (as in --DECOD) are considered to have an Origin of "Assigned" - Most of the coded terms, such as PARAMCD, AVISITN, DTYPE, are “Assigned” in source. Use with "Comment" column and corresponding ID in "Comments" sheet to add more details if needed.
eDT	Data that is received via an electronic Data Transfer (eDT). Use with "Comment" column and corresponding ID in "Comments" sheet.
Protocol	Data that is define as part of the Trial Design preparation. An example would be VSPOS (Vital Signs Position), which may be specified only in the protocol and not appear on a CRF or transferred via eDT. Use with "Comment" column and corresponding ID in "Comments" sheet.
<-> or <Missing>	Missing if value level data has more than one origin

Table 1. Entries for “Origin” column on the Variables worksheet

Assigned – for formats (numeric or character based on codelist); set to a certain text; does not contain ‘when’, ‘where’, etc. These variables are not dependent on other data values.

GENERATE SAS CODE FROM PINNACLE 21 SPECIFICATIONS

The macro, %M_ORIGN, writes code to adsl.sas depending on the three main origin types ('Predecessor', 'Assigned', 'Derived). The high-level steps in the code are outlined in the steps below and as follows:

1. Bring in Pinnacle 21 ADaM specifications for ADSL
2. If Origin=Predecessor and source data is available then output specific data step SAS code to adsl.sas to bring in the source variable and rename the variable.
3. If Origin=Assigned and source data is available then output specific data step SAS code to adsl.sas to bring in the source variable and assign values.
4. If Origin=Derived then add specific data step SAS code from the Methods worksheet or add the comment to act as a guide for programmers. A new column is added to the Methods worksheet called 'SAS Code'.
5. Merge data sets together by key variables obtained from Pinnacle 21 specifications. For ADSL, the key variables are STUDYID and USUBJID.

STEP 1.

The Pinnacle 21 specifications for ADSL are brought in which the LIBNAME statement using the XLSX option:

```
libname specs xlsx "pathname/studyname_adam.xlsx";
```

STEP 2.

If Origin='Predecessor' and source data is available then output specific data step SAS code to adsl.sas to bring in the source variable and rename the variable.

```
data ds1;
   length var whr $200;
   set specs.variables (where=(lowcase(dataset)="adsl" and origin=
"Predecessor")); ❶
   ds=scan(predecessor,1,'.'); ** domain dataset name **;
   out='VAR'||'_'||strip(put(order,8.)); ❷

   if index(predecessor,'where')=0 or index(ds,'SUPP') then do;
      if index(ds,'SUPP')=0 then var=strip(scan(predecessor,2,.')); ❸
      else var=strip(scan(predecessor,2,'"')); ❹
      ds=strip(compress(tranwrd(ds,'SUPP',''))); ❺
      whr='';
   end;
   else do; ❻
      var=scan(predecessor,1,'where');
      var=strip(scan(var,2,'.'));
      type='wheres';
      whr=strip(scan(predecessor,2,'where'));
   end;
   run;
```

❶ Select ADSL specifications that have Origin='Predecessor' on the Variables worksheet from the Pinnacle 21 specifications.

❷ Create output names will be named VARx where x equals the order variable from the Variables worksheet.

❸ Variable predecessor name. For variables with direct predecessor name or SUPP variable QNAMs.

❹ For variables from SUPP take name of variable (need domains created above with SUPP already merged in);

❺ Create a dataset name

❻ For where clause variables coming from domains other than SUPP (e.g. DSDECOD where DSSPID="CRF: END OF TREATMENT")

```
** Create macro vars for looping through list of variables **;
data _null_;
   set ds1 end=eof;
   call symput("VAR"||strip(put(_n_, 8.)),strip(variable));      ** variable
name **;
   call symput("OUT"||strip(put(_n_, 8.)), strip(out));          ** output
sas program name **;
   if eof then call symput("NVAR", strip(put(_n_,8.)));          ** &nvar is
the number of variables for the do loop**;
   run;
%put &nvar;

** Outputs sas code to ADaM data set program (e.g. adsl.sas) for each
predecessor variable **;
%do z = 1 %to &nvar;
   data _null_;
      set ds1 (where=(variable="&&var&z")) end=EOF;
      file "&inpath/prod/programs/adam/&ds..sas" mod;
      dlm = byte(9);
      if _n_=1 then do;
         put " ";
         put "data &&out&z..;";
         put dlm+(-1) "set " ds';' ;
      end;
      if type='wheres' then do;
         put dlm+(-1) "where " whr';';
      end;
      if var ne variable then do;
         txt=strip(var)||'='||strip(variable);
          put dlm+(-1) "rename " txt';';
      end;
      put dlm+(-1) "keep &keyvar " var";";
      if EOF then do;
         put 'run;';
      end;
      run;
%end;

%global outnam;

*** create macro variable outnam for all datasets to be included in the step
below ***;
data _null_;
   length outnam $200;
   retain outnam ' ' ;
   set ds1 end=eof;
   outnam = trim(left(outnam))||''||left(out);
   if eof;
   call symput('outnam',strip(outnam));
```

```
   keep ds outnam type whr var variable order out;
   run;
   %put &outnam;

data _null_;
   file "&inpath/prod/programs/adam/adsl.sas" mod;
   dlm = byte(9);
   put " ";
   put "*** Merge all predecessor data sets together by key variables ***;";
   put "data predecessor;";
   put dlm+(-1) "merge DM &outnam;" ;
   put dlm+(-1) "by &keyvar;" ;
   put "run;" ;
run;
```

%M_ADSL

ADSL.SAS

CONCLUSION

Through the use of Pinnacle 21 data set specifications, it is possible to produce a SAS program, adsl.sas, from the information contained in the specifications.

REFERENCES

"Analysis Data Model (ADaM)". http://www.cdisc.org/adam. The current version is downloadable from the web page and available to CDISC members and non-members.

"Analysis Data Model (ADaM) Implementation Guide". http://www.cdisc.org/adam. The current version is downloadable from the web page and available to CDISC members and non-members.

"CDER Common Data Standards Issues Document". https://www.fda.gov/downloads/Drugs/DevelopmentApprovalProcess/FormsSubmissionRequirements/ElectronicSubmissions/UCM254113.pdf

Shah, Aakar and Tracy Sherman, 2018. Doctor's 'Prescription' to Re-engineer Process of Pinnacle 21 Community Version Friendly ADaM Development. PharmaSUG 2019 Conference Proceedings. Paper DS-15.

ACKNOWLEDGMENTS

We would like to thank Ganesh Gopal from Ephicacy Consulting Group Inc. and Susan Zhao from Nektar Therapeutics for their continued support and encouragement in conference attendance, as well as all our family, friends and colleagues.

CONTACT INFORMATION

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

Name: Tracy Sherman
Enterprise: Ephicacy Consulting Group, Inc.
E-mail: shermantracy@gmail.com

Name: Aakar Shah
Enterprise: Nektar Therapeutics
E-mail: AShah@nektar.com