Paper 117-28

A Simplified and Efficient Way to Map Variable Attributes of a Clinical Data Warehouse

Yanyun Shen, Genentech, Inc., South San Francisco

ABSTRACT

In the pharmaceutical industry, pooling a number of studies together to analyze the safety and efficacy data is a routine task to supplement FDA filing. In order to be able to accomplish this task, data sets of the same domain from different studies need to be standardized to the extent that they have the same structure and set of variables. Corresponding variables from different studies share the same attributes. While there are numerous ways to perform the task, this paper presents a simplified and efficient approach that automates the variable attributes mapping process based on the data set specifications documented in a Microsoft Excel spreadsheet. By converting the spreadsheet information into a SAS® data set, all the variable attributes included in the SAS data set can be applied to a data set of an individual study through a macro call at the end of the program. Any later changes in the data set specifications, which are annoying but inevitable in reality, will not be that difficult to implement.
This paper is intended for the intermediate level SAS programmers who may need to work on data warehouses in the future. The author hopes that the reader can walk way with a simplified and efficient way to build a clinical data warehouse.

INTRODUCTION

In order to evaluate the safety of a study drug before FDA submission, several clinical studies of different phases (I to III) need to be pooled together for an integrated safety summary (ISS). Since some studies may be pooled together for certain aspects of the analysis while others may not, we decided to build a virtual ISS data warehouse. For each study, we built ten data sets of different domains related to safety. For example, adverse event (AE), medical history (HX) and study drug administration (TX) are three of the ten data sets. All the data sets of the same domain from different studies are standardized in structure and variable attributes. Figure 1 illustrates a sample ISS data warehouse structure. For example, data sets of adverse event should have the same structure, which is one adverse event per record per subject. They should also include the same set of variables, and each variable should have the same attributes (label, length and format). While there are lots of different ways to apply standardized variable attributes to a SAS data set, this paper will present an automated variable attributes mapping method which greatly simplifies the mapping task.

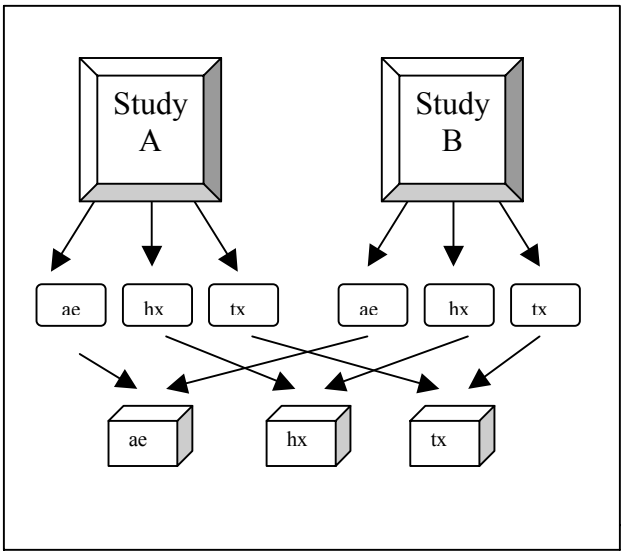


Figure 1. Sample ISS Data Warehouse Structure

Why Do We Need to Automate Variable Attributes Mapping

There are three major reasons that drive us to consider automating variable attributes mapping.

1. Different studies may design case report forms (CRF) differently. Taking adverse event CRF as an example, in one study, variable AERAW (adverse events raw term) may be defined with length 80 characters and labeled as 'Primary AE – Raw Term', while another study may define this variable length with 100 characters and label it as 'AE Raw Term (from CRF)'. In order for us to be able to easily pool these two CRF data sets together, we would ideally need to standardize the variable AERAW with exactly the same length, type, and label. If we didn't want to use either set of AERAW's attributes from the data sets, we could simply make this change in the program that generates the AE data set with the following code:

```
data ae;
set rawdata.ae;
length aeraw $120;
label aeraw='Adverse Event Raw Term' ;

other SAS codes ….……

run;
```

Imagine if we had to modify the variable attributes for every variable in AE data sets for all the studies. Each AE data set contains over 100 variables. If, for example, we decided that 120 was not a large enough length and we wanted to change it to 150, we would have to go back to the program, find the code buried in thousands of lines, and manually make the changes.

2. There are often times when different studies may collect different sets of variables. Take AE CRF as an example again. One study might need to collect hypertension and associated symptoms and signs, i.e., Did the event require a clinic or ER visit? What is the systolic blood pressure and diastolic blood pressure? These variables may not have been collected in other studies. When we build the integrated ISS data warehouse, we need to mock up those variables in the AE data sets for those studies that didn't collect the information. We could generate this in each individual program. But if we had to mock up 30 of those variables, would it be painful?

3. Last but not least, once you have built the data warehouse, you don't have to cross check the attributes of the same variables among all the studies to make sure they are the same.

It seems like there are sufficient reasons for us to choose a simplified and efficient way to accomplish this task. Now I will show you the process that we used step by step and the associated SAS code.

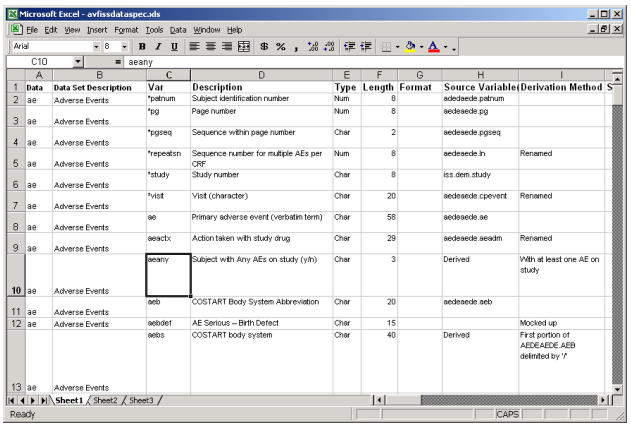
Automatic Mapping Process

Start with Microsoft Excel Spreadsheet

We start by building an MS Excel spreadsheet. There are a number of good reasons we chose Excel spreadsheet as our documentation tool. First, it is a very popular data management tool. Most users have access to it and are familiar with it. Second, it is easy to convert the file into a SAS data set. Third, it meets most of the electronic submission documentation requirements. This means when an e-submission task is due, we already have the bulk of the documentation done. Therefore this Excel spreadsheet is the only data description document we need to maintain for the whole project.
Our Excel spreadsheet basically includes the following columns:

Data set Name
Data Set Description
Variable Name
Variable Description
Variable Type
Variable Length
Variable Format
Source Variable
Derivation Method

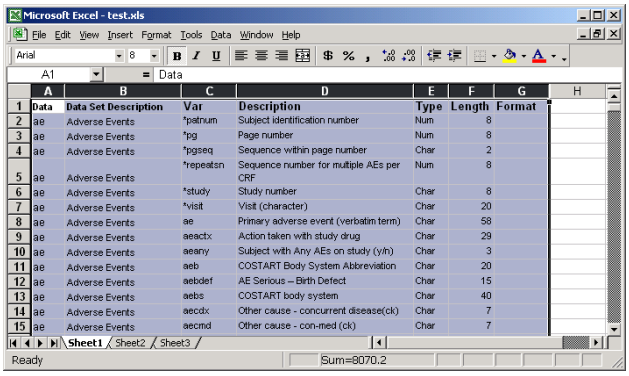
The following is a snap shot of the actual spreadsheet.



All the data sets are listed alphabetically. Within each data set, all the variables are listed alphabetically except the key variables, which are listed at the top based on their sorting hierarchy. The key variables are marked with '*' before their name.

Convert the Excel Spreadsheet to a SAS Data Set

Once the spreadsheet, which is named issdataspec.xls, is built, we use PC SAS import to convert the spreadsheet into a SAS data set called DATASPEC. However our data set of variable attributes only needs the first seven columns. Therefore, we need to remove the rest of the columns, (source variable and derivation method). The process of removing columns and saving the newly created file can be recorded in a macro in the Excel spreadsheet and mapped to a keyboard shortcut. (Ctrl c). In practice, we just need to open the Excel spreadsheet (issdataspec.xls), and hit Ctrl-c. The last two columns are removed and a new Excel spreadsheet is created (test.xls). We then save the new file. A snap shot of the file test.xls is shown below:



When the new spreadsheet is created, we use SAS import to convert it into a SAS data set, DATASPEC. The easiest way to accomplish this is to open SAS, invoke the import procedure once, and then save the import process into a SAS program (import.sas) and run it in batch later on.

Upload the SAS Data Set to Unix System

After DATASPEC is generated, it is transferred to the UNIX

system where our data warehouse is located via FTP Client in a transport file.

Generate a SAS Data Set of Variable Attributes

Once DATASPEC is uploaded to UNIX, the transport file is converted back to a SAS data set. This data set is then converted to a data set called ATTRIBUT by a program named ATTRIBUT. Basically the program translates attributes information into SAS syntax, which can be applied later in the SAS programs. Appendix I contains the program ATTRIBUT, but the major portion of the code is shown below:

```
data one.attribut(rename=(data=tablenm
                          data_set=tblabl
                          var=varnm
                          descript=vardscr
                          lengthc=varlen
                          format=fmtname)
                  drop=type length);
set one.dataspec;

** Remove '*' from the variable name **;
if substr(var, 1, 1)='*'
  then var=substr(var, 2);

** Build variable attributes **;
length lengthc $8;
if type='Num' then lengthc=compress(length);
else lengthc='$'||compress(length);
run;
```

One Step Further

The process of converting the Excel spreadsheet to a SAS data set, transferring the SAS data set to Unix, and generating a SAS data set of variable attributes in Unix can also be combined in one program call if SAS/CONNECT® is available. See the following program for detail code:

```
** Convert the Spreadsheet into a SAS data
on PC **;
%include 'C:\Documents and
Settings\yshen.GNE\Desktop\sugi\import.sas'
;

** Using TCP/IP to connect to UNIX **;
options comamid=TCP remote=SUMO;

** Sign on to Unix **;
** User will be prompted for Unix login ID
and Password **;
signon
'!sasroot\connect\saslink\tcpunix8.scr';

** Location of the SAS data set converted
by SAS IMPORT **;
libname pclib 'C:\Docs\My SAS Files\V8';

rsubmit;

options nofmterr;

** Location of SAS data set on Unix **;
libname unixlib
'/onco/avf/iss00/clinos_tools';

** Upload the SAS data to Unix **;
proc upload data=pclib.dataspec
out=unixlib.dataspec;
run;

** Move to Unix specific directory **;
x 'cd /onco/avf/iss00/clinos_tools';

** Generate ATTRIBUT data set **;
x 'ssub attribut';

endrsubmit;
```

Build Up a Macro Program to Map Variable Attributes

Finally, all we need to do is to come up with a macro program to map the variable attributes. At the end of the program generating the ISS data set of each domain for each study, by calling the macro %ATTRIMAP, all the variables are mapped with the standard attributes defined in the spreadsheet. For those variables originally defined with different lengths, labels or formats, they are all changed to the standard attributes. For those variables not collected in a study, they will be automatically added with the standard attributes. We are no longer overwhelmed by any request that any variable attributes need to be changed or any number of new variables be added. All we need to do is to change the spreadsheet and repeat the process - it is that easy!
For example, ae.sas is a program that generates an adverse event analysis file for every study. At the end of the program, we can simply add the following code:

```
%attrimap(attrlib=sugi,dataset=ae,outlib=sugi
);
```

All we need to do is to pass values to the three macro parameters. ATTRLIB specifies the libname where the attribut data set is located; DATASET denotes the data set name that needs to be applied standard variable attributes; OUTLIB specifies the libname where the output data set is located.
See Appendix II for the whole ATTRIMAP program.

CONCLUSION

Building a clinical data warehouse is a very time consuming task, yet a must for submission. With increasing numbers of studies to be built into a clinical data warehouse, and constant modifications to the attributes being inevitable, automating the variable attributes mapping becomes the least time consuming part in building a clinical data warehouse.

ACKNOWLEDGMENTS

I would like to thank Ai-Yu Wu, who kindly allowed me to publish her macro program as part of this paper. I would also like to thank Raoul Bernal for his valuable advice and Ray Pass for editing my paper. Last but not least, I would

like to thank Lauren Haworth for giving me the opportunity to work on this project and encouraging our participation at SAS conferences.

CONTACT INFORMATION

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

Yanyun Shen
Genentech, Inc.
1 DNA Way, South San Francisco, CA 94080
(650) 225-8385
yshen@gene.com

APPENDIX I

```
*************************************************************************
PROGRAM:
        attribut.sas
DESCRIPTION:
        To generate a data set called ATTRIBUT that includes
        all the iss data set attributes as described below:
        tablenm - Data Set Name
        tblabl - Data Set Label
        varnm - Variable Name
        vardscr - Variable Label
        varlen - Variable Length
        fmtname - Variable Format Name
        The data set attribut will be used by macro %attrimap
        to  apply the attributes for all the ISS data set
        variables.
        The input data DATASPEC is a data set converted
        from an Excel spreadsheet that has the following
        columns with their corresponding converted SAS
        variable names in parentheses:
        Data  ( tablenm)
        Data Set Description (tblabl)
        Var (varnm)
        Description (vardscr)
        Type
        Length (varlen)
        Format (fmtname)
        This program does the following:
        1) Rrenames the variable name as mentioned above.
        2) Combines the type and length variables so that the
           value for attribute can be set up.
        3) Remove the * before some of the key variables.
ASSUMPTIONS:
        None
INPUT:
        dataspec
OUTPUT:
        attribut
MACROS:
        None
AUTHOR:
        Yanyun Shen
DATE WRITTEN:
        6/7/2002

*****************************************************;

** Set up the libname  so that the output data set can be     **;
** saved as a permanent data set . Note  that DATA SPEC **;
** has been  converted to version 6 data                      **;
libname one '.';

data one.attribut(rename=(data=tablenm
                          data_set=tblabl
                          var=varnm
                          descript=vardscr
                          lengthc=varlen
                          format=fmtname)
                                  drop=type length);
set one.dataspec;
  if substr(var, 1, 1)='*' then var=substr(var, 2);
  length lengthc $8;
  if type='Num' then lengthc=compress(length);
  else lengthc='$'||compress(length);
run;
```

APPENDIX II

```
*****************************************************
PROGRAM:
        attrimap.sas
DESCRIPTION:
        A macro that uses SAS data ATTRIBUT to map
        standard attributes to the variables in a SAS data set
REQUIRED INPUTS:
        dataset – the data set that contains all the standard
        variable attributes
OUTPUT CREATED:
        The dataset with standard variable attributes
DESCRIPTION OF MACRO PARAMETERS:
        attrlib  - Location of data set ATTRIBUT
        dataset -  Name of the data set to be applied standard
                   attributes
        outlib – Location of output data set with standard
                   variable attributes
AUTHOR:
        Ai-Yu Wu
DATE WRITTEN:
        8/14/97

*****************************************************;
%macro attrimap(attrlib=,dataset=,outlib=);

%if &outlib =   %then %do;
   %let outlib = rawdata;
%end;

data _tmp;
  set &attrlib..attribut;
  if left(upcase(tablenm)) eq "%upcase(&dataset)";
run;

data _null_;
  set _tmp end=last;

  if tblabl ne '' then call symput('dsetlbl',trim(tblabl));
  call symput('vlen'||left(_n_),varlen);
  call symput('var'||left(_n_),varnm);
  call symput('lbl'||left(_n_),trim(vardscr));
  call symput('fmt'||left(_n_),trim(fmtname));

  if last then call symput('_numvar',left(_n_));
run;

** change attributes of variable length **;
data &outlib..&dataset;
  attrib
  %do i = 1 %to &_numvar;
     &&var&i length = &&vlen&i
  %end;
  ;
  set &outlib..&dataset;
run;

proc datasets lib=&outlib memtype=data nolist;
  modify &dataset (label="&dsetlbl");


  label
    %do _tt=1 %to &_numvar;
        &&var&_tt = "&&lbl&_tt"
    %end;
      ;
   format
    %do _tt=1 %to &_numvar;
        %if &&fmt&_tt ne
          %then &&var&_tt &&fmt&_tt;
    %end;
    %do _tt=1 %to &_numvar;
        %if &&fmt&_tt eq
          %then &&var&_tt &&fmt&_tt;
    %end;
      ;
  informat
  /** erase all informats **/
    %do _tt=1 %to &_numvar;
        &&var&_tt
    %end;
      ;
run;

%mend attrimap;
```