Tracking Metadata within SAS Drug Development Using SDDPARMS

Bradford J. Danner, i3 Statprobe, Lincoln, NE
Matthew J. Wiedel, Celerion, Lincoln, NE
Katrina E. Canonizado, Celerion, Lincoln, NE

ABSTRACT

An increasingly prevalent request from our sponsors is to provide cross-referenced documentation associated with all datasets, programs, macros, and outputs contained in an electronic submission. To accommodate these requests, without modifying our standard processes too much, we found a valuable tool in the SDDPARMS dataset, an optional SAS® file capable of being produced with all programs in SAS Drug Development. We developed an easy-to-use, non-invasive macro to compile and update metadata, required for the documentation package, as a SAS data table. The macro we propose requires limited updates from users and utilizes common Base SAS procedures, macro processing, and the Data step with the Update statement. While permanent SAS datasets are not updated instantly when programs are submitted in batch as Jobs in SAS DD, the method we propose works similarly when submitted from both the Process and Job editors. Once all metadata has been finalized for a submission, the resulting permanent SAS dataset may be processed as needed and produced using an appropriate ODS destination, for example the ExcelXP tagset. Preliminary results suggest a significant reduction in the time to produce the documentation package, and increased quality relative to early, more involved approaches employed.

INTRODUCTION

The decision was made several years ago to upgrade our instance of SAS off a non-supported operating system (VMS, running 8.2 version SAS) over to the latest and greatest version of SAS Drug Development (SDD) at the time (SDD version 3.4, running 9.1.3 SAS). For those who are unaware of SDD, it is a SAS environment for the pharmaceutical and life science industries designed to gracefully meet compliance standards set forth by regulatory agencies (most specifically Item 11/12 compliance), while at the same time providing one enterprise-wide global software and storage platform, and a development and execution environment for SAS programmers, statisticians, and scientists. As one can imagine, the paradigm shift of transitioning from VMS SAS to an entirely new platform was dramatic for our organization. In fact, it is an ever-evolving process that has taken more than two years to accomplish, and admittedly, there are none of us that feel the application has yet to be used to its fullest potential.

However, there have been instances where we have found great efficiencies with the use of our new SAS system, and would like to take this opportunity to briefly discuss our experience. As a small CRO (Celerion) catering to a wide variety of pharmaceutical and biotech customers conducting primarily first in man studies, requests for cross-referenced documentation of programs related to electronic submissions has been limited, until recently, once certain memos/letters by the FDA and third part oversight vendors become increasingly prevalent. At first our standard processes were not designed to gracefully collate and present information regarding program metadata, function, and input/output without a lot of manual labor. Requests for information, besides providing actual programs and macros to sponsors, had us tracing details regarding titles, macros called by individual programs, permanent SAS datasets utilized by specific programs, and similar types of information. Our first such foray into requests of this nature conservatively took greater than 200 non-billable hours, and over a month of back-and-forth reviews and subsequent revisions.

Manual reviews by contributors to various projects were intensive, not efficient or profitable use of time, and quite often were prone to error and several cycles of re-work. Different types of reviews were required based upon expertise – specifically programmers, reporting associates, and report leaders were all necessary in delivering an adequate final product. Examples of the great potential for error included, but were not limited to, spelling (major and minor), transcribed numbers, and of greatest importance, false cross-referencing of submission components and their associated metadata and outputs.

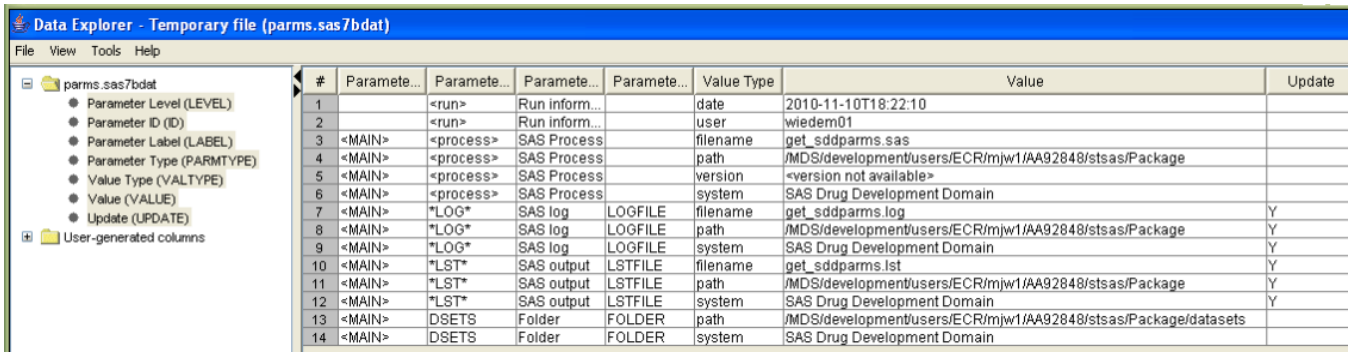
To alleviate some of the problems, we realized several tools within SAS, and specifically SDD, were at our disposal, and would efficiently provide the deliverable we had been struggling to produce. Given the necessary information required for the cross-referenced documentation package, much could be gleaned from SASHelp Views or Dictionary Tables (specifically output title names and numbering associated with a program). Much of the other information required was conveniently located in the SDDPARMS dataset (an optional SAS table associated with a program within SDD holding information about any user-defined parameters accessed within that run cycle). Furthermore, not wanting the end process to be too complicated, we limited our code to take advantage of simple Base SAS tools, specifically the Data step, macro language and processing, and the SQL procedure. And finally, we did not have the time, nor the desire, to redevelop all the standard table and figure programs required for a submission. We wanted our approach to use the programs which were already available, and be non-invasive, such that limited updates would be required by end users.

METHODS/FRAMEWORK

The most basic information required by the documentation packages was an accurate assessment of Table and Figure titles and numbers. Several of our programs store this information in an accessible repository, but, some programs do not, requiring the user to enter this type of information directly within the program itself. Therefore, we knew we needed a way to track this information as the output was being created. And since many of our applications produce multiple outputs, tracking this information had to be accomplished dynamically within a program, regardless of how many outputs were created. Thus we chose to us the SQL procedure to query the TITLES dictionary following every production of an output as follows:

```
%global mtdpnam mtdids_ mtdmcals_ mtdtit mtdts;
proc sql noprint;
select text into: mtdtit separated by '~' from dictionary.titles where(type="T");
```

In order to tie this in with the name of the program, runtime, various accessed datasets, and programs or macros invoked to produce the output, information was gathered from the SDDPARMS dataset every time a program was run. Many programmers not familiar with SAS Drug Development may wonder what this PARMS dataset is. In a nutshell, any metadata or macro parameter used in a program may be accessed through a temporary dataset every time it is run, and some of the information is exactly what we needed to compile our documentation package. The following screen print provides a very simple example of some of the dynamic information available to a programmer every time he/she executes a process in SDD.



#	Paramete...	Paramete...	Paramete...	Paramete...	Value Type	Value	Update
1		<run>	Run inform...		date	2010-11-10T18:22:10	
2		<run>	Run inform...		user	wiedem01	
3	<MAIN>	<process>	SAS Process		filename	get_sddparms.sas	
4	<MAIN>	<process>	SAS Process		path	/MDS/development/users/ECR/mjw1/AA92848/stsas/Package	
5	<MAIN>	<process>	SAS Process		version	<version not available>	
6	<MAIN>	<process>	SAS Process		system	SAS Drug Development Domain	
7	<MAIN>	*LOG*	SAS log	LOGFILE	filename	get_sddparms.log	Y
8	<MAIN>	*LOG*	SAS log	LOGFILE	path	/MDS/development/users/ECR/mjw1/AA92848/stsas/Package	Y
9	<MAIN>	*LOG*	SAS log	LOGFILE	system	SAS Drug Development Domain	Y
10	<MAIN>	*LST*	SAS output	LSTFILE	filename	get_sddparms.lst	Y
11	<MAIN>	*LST*	SAS output	LSTFILE	path	/MDS/development/users/ECR/mjw1/AA92848/stsas/Package	Y
12	<MAIN>	*LST*	SAS output	LSTFILE	system	SAS Drug Development Domain	Y
13	<MAIN>	DSETS	Folder	FOLDER	path	/MDS/development/users/ECR/mjw1/AA92848/stsas/Package/datasets	
14	<MAIN>	DSETS	Folder	FOLDER	system	SAS Drug Development Domain	

As briefly demonstrated above, the use of SQL to dynamically grab all available title information from the TITLE dictionary was just too good to pass up. Therefore, we chose to duplicate this manner of syntax when querying the SDDPARMS dataset for additional information. We chose to use a macro variable process so that every time the macro was compiled, the information could easily be re-declared and initialized to null. Once the information was gathered, it was passed into both a permanent and temporary SAS dataset, conveniently using the UPDATE function on the SAS dataset, which allows us to access a dataset, without having to run through the entire dataset line by line. The following example code illustrates this:

```
select value into :mtdpnam from &SDDPARMS where (id="<process>" and valtype="filename");
*** program name ***;
select value into :mtdids_ separated by ', '  from &SDDPARMS
where ((id="SDTM" or id="INC" or id="CDASH" or id="ADAM") and (index(value,'.sas7bdat')))
order by value; *** datasets ***;
select value into :mtdmcals_ separated by ', ' from &SDDPARMS
where (index(value,'.sas') and id ne "SETUP" and index(value,'.sas7bdat')=0 and id ne
"<process>" and id ne "MNEW") order by value; *** SAS programs or macros included in a
program ***;
select put(input(scan(value,1,'T'),yymmdd10.),date9.)||" "||scan(value,-1,'T') into :mtdts
from &SDDPARMS where upcase(valtype)="DATE";
quit;  *** run date and time ***;
```

The establishment of a permanent SAS dataset for subsequent use is obvious. Perhaps less obvious, the redundant step of creating a temporary SAS dataset in the Work library, although the name chosen for this temporary dataset was chosen such that the likelihood a user of the application would unintentionally overwrite it was minimized. Within the SDD system, programs may be submitted individually within the process editor, or in sequence using what is referred to as jobs within the job editor. Depending on which interface is used to submit code and produce output changes the functionality and operation of our metadata tracking approach.

Within SAS DD, the process editor is a Java application that allows users to build and modify process, or programs. One of the strengths of the process editor is programs can be designed to run automatically with no user intervention, or alternatively with a user interface to solicit specific parameters, for example data files or variables to use or output destinations.

The job editor within SAS DD is a Java application that allows users to run multiple processes sequentially. Unfortunately, when processes are run sequentially in a job, updates made to a permanent dataset, are not updated until the end of the job. This can often lead to misleading results, and reports generated using obsolete data.

Unfortunately, when programs are submitted in sequence using a job editor, permanent SAS datasets are not updated until the very last process within the job. Only the entry created by the last program is reflected in the permanent metadata. Given our approach, it is necessary to update our running log of metadata every time an output is produced. To alleviate this difficulty, using both a permanent and temporary (with a reserved designation) dataset was required and is illustrated in the following example code:

```
%let mtdexist=mnew.metadata;
%let wmtdexist=work.metadata_injob;

data work.metadata;
length runtime $30.     progname $40. idata $200.   macros $200.  title $200.;
runtime="&mtdts.";
progname="&mtdpnam.";
idata="&mtdids_.";
macros="&mtdmcals_.";
title="&mtdtit.";
run;

data work.metadata_injob;
%if %sysfunc(exist(&wmtdexist)) %then %do;
update work.metadata_injob work.metadata;
%end;
%else %do;
set work.metadata;
%end;
by title;
run;

data mnew.metadata;
%if %sysfunc(exist(&mtdexist)) %then %do;
update mnew.metadata work.metadata_injob;
%end;
%else %do;
set work.metadata_injob;
%end;
by title;
run;
```

Finally, all the pieces of this process are centrally located in a macro repository, aptly named **%metadata**, and called/compiled every time output is produced (Canonizado *et al.,* 2011). Therefore, the only modification required of the individual programs themselves is to invoke the macro – which was added during the last maintenance release of many standard programs.

RESULTS

The SDDPARMS dataset is a SAS table representation of the parameters window available with any SAS program or process. The dataset is composed of six columns, or variables [LEVEL, ID, LABEL, PARMTYPE, VALTYPE, VALUE, and UPDATE]. Without going into too much detail, various types of information are accessible programmatically though this table and include user id, run date and time, program name and pathway, name and pathways of all inputs, outputs, logs, and macros. While much, if not all, of this information may be available using a combination of SASHELP or DICTIONARY tables, the really wonderful thing about SDDPARMS is all the information is centrally located in one place!

To accommodate the initial needs of our documentation package, we selected to always associate the current run date and time with the program name, all input datasets accessed by the program, all macros included by the program, all linked to each individual output file identified by title and number, in our case usually LST text output, CGM, or RTF. Therefore, the metadata dataset produced would contain one row for every individually titled output produced. Given the design of the approach, it was very easy for new programmers to a project to implement the macro. The only direction required is to call the macro following the production of an output!

Once all programs for a submission have been validated, and everything has ideally been finalized, we now have one SAS dataset that has nearly all the information required most sponsors requirements for submission package documentation. The following example briefly illustrates a subset of our submission metadata in SAS data format:

#	runtime	progname	idata	macros	title
1	14JUL2010 21:18:01	SDTM-LIS-LNO.SAS	adsl.sas7bdat, co.sas7bdat, l...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.1.10.1. Clinical Laboratory Reference Ranges
2	14JUL2010 21:16:03	SDTM-LIS-DIS.SAS	adsl.sas7bdat, co.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.1. Subject Discontinuation
3	14JUL2010 21:15:53	SDTM-LIS-DEM.SAS	adsl.sas7bdat, co.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.4.1. Demographics
4	14JUL2010 21:14:10	CDASH-LIS-PHY-BES...	adsl.sas7bdat, pe.sas7bdat	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.4.2. Physical Examination
5	14JUL2010 21:19:00	SDTM-LIS-MH.SAS	adsl.sas7bdat, mh.sas7bdat,...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.4.3. Medical and Surgical History
6	14JUL2010 21:19:14	SDTM-LIS-SU.SAS	adsl.sas7bdat, su.sas7bdat	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.4.4. Substance Use
7	14JUL2010 21:16:38	SDTM-LIS-INC.SAS	adsl.sas7bdat, ieop.sas7bdat	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.1.1. Inclusion Criteria
8	14JUL2010 21:16:23	SDTM-LIS-EXC.SAS	adsl.sas7bdat, ieop.sas7bdat	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.1.2. Exclusion Criteria
9	14JUL2010 21:16:31	SDTM-LIS-IE.SAS	adsl.sas7bdat, ie.sas7bdat, i...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.2. Subject Eligibility
10	14JUL2010 21:15:21	SDTM-LIS-CHK.SAS	adsl.sas7bdat, mqop.sas7bd...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.3.1. Check-in and Return Criteria
11	14JUL2010 21:15:28	SDTM-LIS-CHK1.SAS	adsl.sas7bdat, mqop.sas7bd...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.3.2. Check-in and Return Responses
12	14JUL2010 21:18:24	SDTM-LIS-MED.SAS	adsl.sas7bdat, ex.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.4.1. Test Compound Description
13	14JUL2010 21:18:37	SDTM-LIS-MED2.SAS	adsl.sas7bdat, co.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.4.2. Test Compound Administration Times
14	14JUL2010 21:15:13	SDTM-LIS-BLD.SAS	adsl.sas7bdat, pcop.sas7bdat	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.5. Blood Draw Times
15	14JUL2010 21:18:51	SDTM-LIS-MEL.SAS	adsl.sas7bdat, ml.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.6. Meal Times
16	14JUL2010 21:15:41	SDTM-LIS-CON.SAS	adsl.sas7bdat, cm.sas7bdat,...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.5.7. Concomitant Medications
17	30JUL2010 14:21:03	SDTM-LIS-AE.SAS	adsl.sas7bdat, ae.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.7.1. Adverse Events (I of II)
18	14JUL2010 21:14:44	SDTM-LIS-AE2.SAS	adsl.sas7bdat, ae.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.7.2. Adverse Events (II of II)
19	14JUL2010 21:14:55	SDTM-LIS-AE3.SAS	adsl.sas7bdat, ae.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.7.3. Adverse Event Comments
20	14JUL2010 21:15:04	SDTM-LIS-AE4.SAS	adsl.sas7bdat, ae.sas7bdat, ...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.7.4. Adverse Event Preferred Term Classification
21	26JUL2010 15:09:19	SDTM-LIS-LAB.SAS	adsl.sas7bdat, co.sas7bdat, l...	CDATETIME.sas, MERGE_SUPP.sas, ...	Appendix 16.2.8.1.1. Clinical Laboratory Report - Serum Chemistry (I

Of course, many sponsors have their own preferences and ideas as to how the information is to be presented, however, since the information is in a SAS dataset, and we are using SAS to access it, manipulation, re-arrangement, and mining of the data to satisfy customer needs is programmatically possible with relatively little time or effort, using BASE SAS procedures, and the DATA step.

COMPONENTS OF SUBMISSION AND DOCUMENTATION PACKAGE

Data package requests can differ from client to client. The FDA suggests attaching Table and Figure code along with other supplemental documentation to tag along with the actual data and report submission. Clients are thus asking for asking an increasing amount of information to demonstrate due diligence. Below is a list of items we have routinely been asked to include in a typical Data Submission Package:

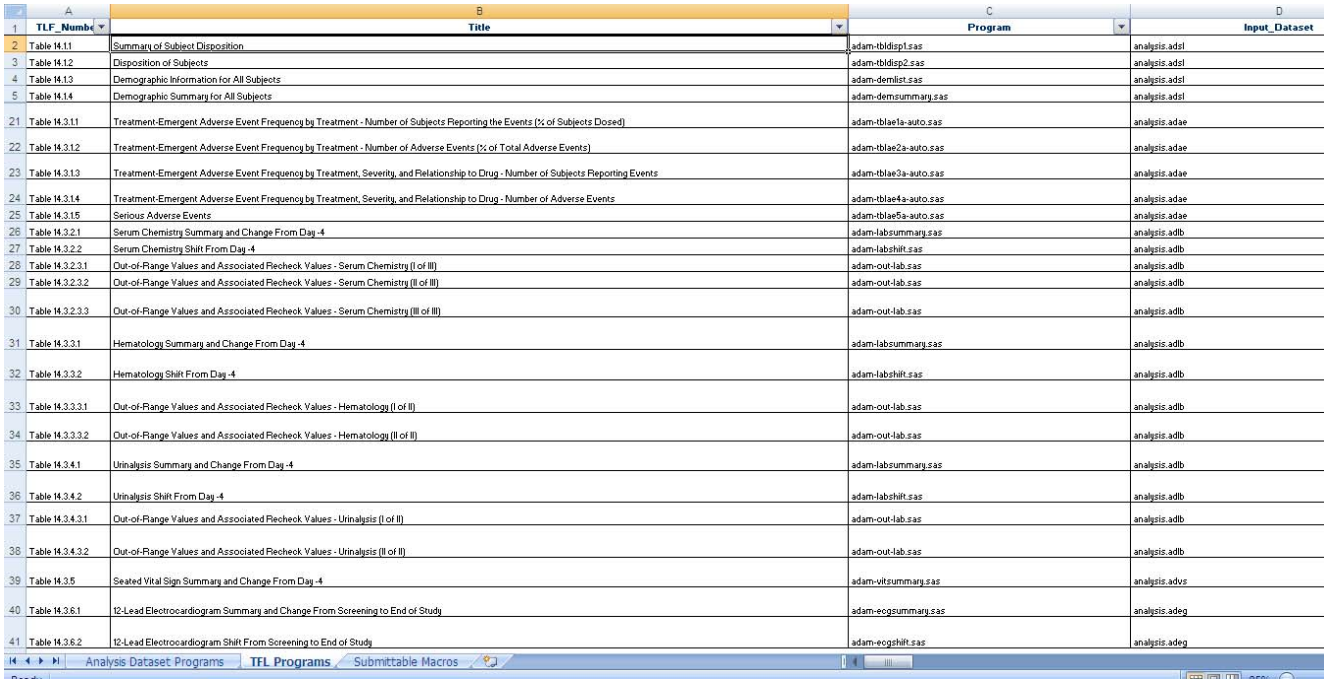
- *SDTM*: Study Data Tabulation Model: A database with structure readily accepted by the FDA. These standards are created and updated by the CDISC organization.
- *ADaM:* Analysis Data Model: A database structure derived from the SDTM database, designed to ideally facilitate efficient analysis.
- *DEFINE.XML*: Navigable documentation for SDTM and ADaM and a study's Case Report Form with four main parts: Table of Contents of Domains, Variable Level information, Value Level information, and Code list information.
- *ANNOTATED CRF:* Annotation of the original CRF using the mapped SDTM variable names with explanation when needed.
- *DATASET, TABLE AND FIGURE PROGRAMS:* The SAS programs that create all the ADaM datasets along with all the post report Safety, PK, PD, or efficacy tables and graphs which use the ADaM data as input data. Each of the programs should only have to input one ADaM dataset, and macros that perform statistical inference are included.
- *eCTD TOC: Electronic Common Technical Document Table of Contents:* A display of contents is inside the submission package. It displays type of item, item name, and titles. Often a SAS program will create many different tables or figures.
- *ANALYSIS PROGRAM AUDIT LIST:* A list displaying the titles of each table and figure with its corresponding table/figure number, input dataset, and program used to create the table or figure. Supplemental worksheets for ADaM dataset creation programs and submission required macros are also included.

Now that we have our submission metadata collected, and the ability to programmatically manipulate it, the information may be effectively directed to and presented in the eCTD TOC and Audit Program Lists. Several alternatives are possible for delivery of this information, such as linkable PDF or HTML documents, of Word readable RTF. Perhaps one of our more favored approaches to presenting the information makes use of the ExcelXP tagset. Our first foray into the production of a submission package and associated documentation required the production and delivery of multi-worksheet Excel spreadsheets and at the time were compiled manually. But with the use of the ExcelXP tagset we are able to do all production work from within SAS. A very simple sub-setted example of an eCTD TOC, viewed in Excel is shown here:

	A	B	C
1	File Type	File Name	Title
34	SDTM Documentation	blankcrf.pdf	Tabulations: Annotated Case Report Form for Tabulations Data
35	SDTM Documentation	tabulations-data-guide.doc	Tabulations: Guide to Tabulations Datasets - Celerion
36	SDTM Documentation	define.xml	Tabulations: Data Define Document
52	Analysis Documentation	analysis program audit list.xls	Analysis Tables and Figures: Guide to Table and Figure Programs
53	Analysis Documentation	analysis-data-guide.doc	Analysis: Guide to Analysis Datasets - Celerion
54	Analysis Documentation	define.xml	Analysis: Data Define Document
55	Program	adam-tbldisp1.sas	Analysis Table Program: Table 14.1.1. Summary of Subject Disposition
56	Program	adam-tbldisp2.sas	Analysis Table Program: Table 14.1.2. Disposition of Subjects
57	Program	adam-demlist.sas	Analysis Table Program: Table 14.1.3. Demographic Information for All Subjects
58	Program	adam-demsummary.sas	Analysis Table Program: Table 14.1.4. Demographic Summary for All Subjects
59	Program	adam-tblae5a-auto.sas	Analysis Table Program: Table 14.3.1.5. Serious Adverse Events
60	Program	adam-tblae1a-auto.sas	Analysis Table Program: Table 14.3.1.1. Treatment-Emergent Adverse Event Frequency by Treatment - Number of Subjects Reporting the Event (% of Subjects Dosed)
61	Program	adam-tblae2a-auto.sas	Analysis Table Program: Table 14.3.1.2. Treatment-Emergent Adverse Event Frequency by Treatment - Number of Adverse Events (% of Total Adverse Events)

The basic method by which these multi-sheet documentation spreadsheets are produced within SAS, using only the PRINT procedure and the ExcelXP tagset, with only a few of the many formatting options specifically determined, is illustrated here, using an Analysis Program Audit List as an example:

```
ods listing close;
ods tagsets.Excelxp
    file="&outfolder/ANALYSIS PROGRAM AUDIT LIST.xml" style=normal
    options(
           zoom='100'
           frozen_headers='Yes'
           autofilter="All"
           absolute_column_width="40"
           sheet_interval='Proc'
           sheet_name="Analysis Dataset Programs");

proc print label data=dsets.anal_sets noobs;
    label value='Dataset Name' memlabel='Title' pname='Program'
    idata='Input Datasets';
    var value memlabel pname idata;
run;

ods tagsets.Excelxp options(sheet_name="TFL Programs"
                            absolute_column_width='15,100,40,40'
                            autofit_height='Yes'
                            zoom='85');

proc print label data=dsets.all_titles noobs;
    var tnumber title program idata;
    where index(tnumber,'Appendix')=0;
    *** LISTINGS NOT REQUIRED TO BE DOCUMENTED ***;
run;

ods tagsets.Excelxp options(sheet_name="Submittable Macros"  absolute_column_width="50"
                           zoom='100');

proc print label data=dsets.macros noobs;
    var value purpose programs;
run;

ods tagsets.Excelxp close;
ods listing;
```



TLF_Numbe	Title	Program	Input_Dataset
Table 14.1.1	Summary of Subject Disposition	adam-tbldisp1.sas	analysis.adsl
Table 14.1.2	Disposition of Subjects	adam-tbldisp2.sas	analysis.adsl
Table 14.1.3	Demographic Information for All Subjects	adam-demlist.sas	analysis.adsl
Table 14.1.4	Demographic Summary for All Subjects	adam-demsummary.sas	analysis.adsl
Table 14.3.1.1	Treatment-Emergent Adverse Event Frequency by Treatment - Number of Subjects Reporting the Events (% of Subjects Dosed)	adam-tblae1a-auto.sas	analysis.adae
Table 14.3.1.2	Treatment-Emergent Adverse Event Frequency by Treatment - Number of Adverse Events (% of Total Adverse Events)	adam-tblae2a-auto.sas	analysis.adae
Table 14.3.1.3	Treatment-Emergent Adverse Event Frequency by Treatment, Severity, and Relationship to Drug - Number of Subjects Reporting Events	adam-tblae3a-auto.sas	analysis.adae
Table 14.3.1.4	Treatment-Emergent Adverse Event Frequency by Treatment, Severity, and Relationship to Drug - Number of Adverse Events	adam-tblae4a-auto.sas	analysis.adae
Table 14.3.1.5	Serious Adverse Events	adam-tblae5a-auto.sas	analysis.adae
Table 14.3.2.1	Serum Chemistry Summary and Change From Day -4	adam-labsummary.sas	analysis.adlb
Table 14.3.2.2	Serum Chemistry Shift From Day -4	adam-labshift.sas	analysis.adlb
Table 14.3.2.3.1	Out-of-Range Values and Associated Recheck Values - Serum Chemistry (I of III)	adam-out-lab.sas	analysis.adlb
Table 14.3.2.3.2	Out-of-Range Values and Associated Recheck Values - Serum Chemistry (II of III)	adam-out-lab.sas	analysis.adlb
Table 14.3.2.3.3	Out-of-Range Values and Associated Recheck Values - Serum Chemistry (III of III)	adam-out-lab.sas	analysis.adlb
Table 14.3.3.1	Hematology Summary and Change From Day -4	adam-labsummary.sas	analysis.adlb
Table 14.3.3.2	Hematology Shift From Day -4	adam-labshift.sas	analysis.adlb
Table 14.3.3.3.1	Out-of-Range Values and Associated Recheck Values - Hematology (I of II)	adam-out-lab.sas	analysis.adlb
Table 14.3.3.3.2	Out-of-Range Values and Associated Recheck Values - Hematology (II of II)	adam-out-lab.sas	analysis.adlb
Table 14.3.4.1	Urinalysis Summary and Change From Day -4	adam-labsummary.sas	analysis.adlb
Table 14.3.4.2	Urinalysis Shift From Day -4	adam-labshift.sas	analysis.adlb
Table 14.3.4.3.1	Out-of-Range Values and Associated Recheck Values - Urinalysis (I of II)	adam-out-lab.sas	analysis.adlb
Table 14.3.4.3.2	Out-of-Range Values and Associated Recheck Values - Urinalysis (II of II)	adam-out-lab.sas	analysis.adlb
Table 14.3.5	Seated Vital Sign Summary and Change From Day -4	adam-vitsummary.sas	analysis.advs
Table 14.3.6.1	12-Lead Electrocardiogram Summary and Change From Screening to End of Study	adam-ecgsummary.sas	analysis.adeg
Table 14.3.6.2	12-Lead Electrocardiogram Shift From Screening to End of Study	adam-ecgshift.sas	analysis.adeg

As briefly mentioned before, our first such attempt to compile a documentation package was accomplished primarily through manual means, often leading to many errors, and an intensive, or extensive, revision process, that neither we nor the client enjoyed. Through some simple tools available in Base SAS, and the helpful session metadata provided by the SDDPARMS dataset in SAS Drug Development, we are now capable of programmatically producing this deliverable, saving both the client and SAS programmers time and money, not to mention frustration. An additional benefit to accomplishing this task programmatically in SAS is the traceability that has been achieved.

DISCUSSIONS AND CONCLUSIONS

While much efficiency has been achieved through the adoption of this process, we are still continually finding areas for improvement and further innovation. Perhaps the initial, and primary caveat we discovered, was the recommendation to run, or activate the macro only once a program and been through validation and nearly finalized, to avoid adding false or incorrect data to the metadata dataset. Methods are available to clean such spurious records, and innovations in process to minimize the time necessary to make corrections.

Once the metadata is compiled, the permanent SAS dataset may be modified and updated if necessary by the submission or reporting programmer. As stated before, should corrections to be required to outputs, extraneous or incorrect records may become part of the metadata. Using Base SAS procedures, records can be corrected, added, or removed entirely prior to preparing the final submission package. Alternatively, with minor updates to the macro itself, its possible post-production modification could become unnecessary since updates are automatically incorporated by running original programs.

Accessing and compiling submission metadata through the SDDPARMS dataset using the **%metadata** macro (Canonizado *et al*., 2011), or a similar approach, we feel is a non-invasive tool programmers may use to help document and track data about individual programs and their outputs. However, in order to produce accurate metadata about the program, some investment of time may be required early on during program development. The most obvious example is the declaration of data inputs in a program. For example, when declaring libraries and locations of data, it is important to only include the datasets used by a program, and not entire folder contents, otherwise a program will claim to use more data than it actually has. While taking such care early on might be tedious (excessive button clicking in the Process Editor interface), it will result in cleaner programs that run more efficiently, and does encourage a ‘best practice’ approach to programming.

In conclusion, we believe the use of the SDDPARMS dataset in SAS Drug Development proposed herein offers an efficient programmatic approach to compiling metadata useful for documenting an electronic submission. The approach requires minimal administrative oversight, as compared to manually tracking all components within the submission, and has great potential to save both time and money. Although testing has been limited, we estimate at least a 90% reduction in time necessary to compile (relative to earliest manual attempts deliver), and occurrence of errors and time to complete review cycles has dramatically decreased. While accessing the SDDPARMS dataset as we have has offered a quick and efficient application to compile metadata, we believe other dynamic endpoints may be accomplished through its use in SAS Drug Development. Possible examples include document versioning, file compilations, and dynamically driven code generation.

Similar to how the SASHELP and DICTIONARY libraries may be used, we believe SDDPARMS on the SAS Drug Development platform offers an equally powerful tool to developers willing and able to access it.

REFERENCES AND RECOMMENDED READING

Canonizado, K.C., B.J. Danner and M.J. Wiedel. A Non-Invasive Macro to Track Submission Metadata in SAS® Drug Development, SAS Global Forum, 2011.

SAS 9.1.3 Online Documentation

SAS Drug Development 3.4 User's Guides

ACKNOWLEDGEMENTS

We would like to thank our colleagues in the Biostatistics and Pharmacokinetics groups in Clinical Pharmacological Sciences at Celerion who provided comments on earlier drafts, and reviewers in the Statistical Programming department at i3 Statprobe. Their insights and guidance are appreciated.

CONTACT INFORMATION

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

Bradford J. Danner
i3 Statprobe
brad.danner@i3statprobe.com

Matthew J. Wiedel
Celerion
621 Rose Street
Lincoln, NE 68502
matthew.wiedel@celerion.com

Katrina E. Canonizado
Celerion
621 Rose Street
Lincoln, NE 68502
katrina.canonizado@celerion.com