PharmaSUG 2013 - Paper TF13

Mission Possible: Your Assignment is to Validate Output for a Study

Susan Fehrer Coulson, BioClin, Inc., Emporia, KS
Kevin R. Coulson, Emporia State University, Emporia, KS

ABSTRACT

You have been called into your manager's office and have been told that your next assignment is to validate the programming in a study that you did not work on before. Why does this task have to be done?

It can be a daunting task to start validating a project, any project, if it is one that you worked on or not. And, where is the best place to start?

With examples and best practices of where and how to start, you will have a Mission Possible.

INTRODUCTION

The biopharmaceutical industry is the most heavily regulated industry in the US, even more than those of banking and finance. The Food and Drug Administration (FDA) is the US regulatory body that has approval authority of drugs and devices. The European Medicines Agency (EMEA) is the FDA's equivalent in the European Union and the Health Protection Branch of the Canada Department of National Health and Welfare (HPB) is Canada's. These regulatory agencies each have requirements that a clinical trial's data are correct and the output 'makes sense.' This paper will refer to the FDA, though the other pharmaceutical regulatory agencies have similar requirements.

Validation of the programs creating derived or analysis data sets, CDISC SDTM or ADaM, and the programs creating tables, listings, and figures (TLFs) assures the correctness of subject data for submission.

The regulations for submitting clinical trials data are not within the scope of this paper. This paper assumes the reader's familiarity with CDISC SDTM.

WHY VALIDATE

Aside from each programmer and statistician having pride in their work and confidence that what s/he produces is of the highest possible quality, there are federal regulations which require that a clinical trial's output be validated. It is not uncommon for the FDA to request copies of SAS® programming code as well as SAS Logs. Would you want the FDA to uncover your programming issues (what non-industry folks may call mistakes)?

The FDA has guidances for submitting clinical trial data as well as requirements for clean subject data. Also, company management will have a validation requirement that may vary depending on the task and the possible risk assessment related to data importance.

We all have heard of the Systems Development Life Cycle (SDLC) and, though not all of the SDLC applies to a clinical trial's output, there are many aspects that do apply, and following the SDLC is the best way to develop a validation program.

Validating programs and their output does reduce development costs, especially if validation begins as soon as the data sets have been created. And biopharmaceutical companies want to reduce development time to get their submission to the FDA as quickly as possible. The sooner the submission, hopefully, the sooner the drug or device will be approved and the company will start to recoup the development costs.

Validation provides correct conclusions about the safety and efficacy of drugs and devices. The FDA must approve all protocols before any clinical trial is conducted in the US. The study output must address each of the study objectives as set forth in the protocol. Without correct conclusions drawn from the tables, listings, and figures, the drug may be incorrectly approved and may be withdrawn from the market.

If the FDA were to find issues or errors, they may come to the biopharmaceutical company to do a full audit, they may levy a substantial fine against the company, and/or they may issue a "refusal to file." If any of these actions are attributable to a programmer's work, s/he may be looking for a new place of employment sooner rather than later.

Make the FDA's job easier – don't let them find programming issues or errors. Prove that what was supposed to have been done, has been done - correctly.

A good website to check for current information from the Bioresearch Monitoring Program (BIMO) is www.fda.gov/oc/gcp.

WHERE TO START

All programmers and statisticians must begin a validation project by reading the protocol! Programmers must read about the study design (double blind, single blind, or crossover), study conduct location, study population, safety parameters, efficacy objectives and parameters, laboratory tests, and questionnaires. In other words, RTFM – read the fine manual, and in this case, RTFP, read the fine protocol.

After taking time to read the protocol, read the annotated Case Report Form (aCRF). Read how dosing information is collected and note what data are collected within each raw data set. Is the study following the SDTM Implementation Guide (IG)? After reading the aCRF, read the Analysis Plan (AP) or Statistical Analysis Plan (SAP) or Report and Analysis Plan (RAP). All three names are terms for the same type of document that should summarize the protocol, have programming considerations, and provide layouts for the output TLFs.

VALIDATING DERIVED DATA SETS

If the task at hand is to validate data sets, start with the demography (DM) data set. All other data sets depend on the correctness of the DM data.

The DM data set contains one record per subject. Run a PROC FREQ on the subjects to confirm. And, while writing the DM validation program, run a PROC FREQ on all required variables.

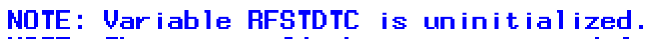
For example:

```
proc freq data=urstudy.dm ;
    tables studyid domain usubjid subjid siteid sex armcd country / list missing
    nopct ;
title1 'urstudy.dm – required variables' ;
```

In this example, the author adds spaces before the semi-colons, before and after the '/', and in the title statement for readability. In adding a space before the semi-colons, it is apparent to the author when (and if) a semi-colon is missed. The author also makes every effort to type code within the visible enhanced editor window of SAS. Allowing programming code to run over is counter to good programming practice.

If these required variables are not all populated, open the raw data sets that contribute to DM and run a similar PROC FREQ on the raw data. The STUDYID, DOMAIN, USUBJID, and COUNTRY, standard CDISC SDTM variables, may or may not have raw data values. Often the DM creation program derives these values. If the study is conducted in only one country, as per the protocol, COUNTRY is often created in the DM program (using ISO format). If these variables have not been created in the DM program, follow the company's operating guidelines, which will likely instruct the validation programmer to meet with the programmer who wrote the DM program.

If any of the expected variables have null values, check with the project manager. It is rare to have null values for them, especially for RFSTDTC. But, when running a PROC FREQ on the expected variables and a message in the SAS log appears stating that a variable is uninitialized, the variable needs to be added in the program creating DM.



```
NOTE: Variable RFSTDTC is uninitialized.
```

Display 1. Uninitialized variable message in SAS LOG

Part of the DM validation includes the variables in the order in which they appear in the CDISC SDTM Implementation Guide. When running a PROC CONTENTS to see all of the variables in DM, add the option ORDER=VARNUM to print the variables in the order in which they are in the data set. The SDTM IG explicitly states the order in which variables are to be listed in the data set and this order must be followed. Without the ORDER=VARNUM statement, the variables will print out in alphabetical order.

For example (the bold font for the option is for effect):

```
proc contents data=urstudy.dm order=varnum ;
title1 'urstudy.dm –variables in data set order' ;
```

Visually check the variable labels for spelling and case, and visually check that the variable names are spelled correctly. It is easy to transpose letters, particularly if the programmer is operating in his/her second (or third) language. Also confirm that the formats listed in the PROC CONTENTS output exist.

Follow a similar method to validate the data in other data sets – check the required variables, check for the presence of the expected variables. And, check to see if the values make sense. Programmers and statisticians are often not clinicians who can discern potential clinical issues. Look at a PROC PRINT of the data and if many of the subjects have a RFSTDTC value of 2012-11-22 be concerned. Why? That happened to be Thanksgiving Day in 2012. It is possible that many subjects started dosing on that day, but unlikely. Follow the company's operational guidelines for possible fraud if something like this should be found.

Validating a laboratory (LB) data set takes a significant amount of time. This is because of the many variables and possible conversions from conventional (US) to standard (SI) units. In the protocol, the table of lab categories and tests is the reference listing all of the lab tests that must be performed and reported in the study. Often a lab raw data file will contain many more tests than what are listed in the protocol. It is the option of the biopharmaceutical company to submit additional lab tests.

To check values of other lab tests that are not required by protocol, but will be submitted to the FDA, there are online websites with lab tests, units, and normal ranges. One of the better sites for lab tests and conventional and standard units, including conversion units (multiplier to convert values from conventional to standard units), is http://www.unc.edu/~rowlett/units/scales/clinical_data.html. Although the normal ranges that are found online are 'general' values, note that each clinical laboratory has its own set of normal ranges which are calibrated to their equipment.

In the SDTM IG, both original and standard values and units are presented. Check to be sure that the standard units, often the SI units, for each lab test are the same using PROC FREQ of LBTESTCD with LBSTRESU, standard units.

For example:

```
proc freq data=urstudy.lb ;
    tables lbtestcd * lbstresu / list missing nopct ;
title1 'urstudy.lb – look for all standard units per lab test to be the same' ;
```

If, for example, creatinine (LBTESTCD=CREAT) has LBSTRESU with values of both 'mg/dL' and 'umol/L,' add code to the PROC FREQ to get more information. Adding the subject identifier and the visit will provide information that will be valuable to resolve the value for LBSTRESU.

For example:

```
proc freq data=urstudy.lb ;
    tables usubjid * visit * lbtestcd * lbstresu / list missing nopct ;
    where lbtestcd = 'CREAT' & lbstresu = 'mg/dL' ;
title1 'urstudy.lb – subjects with non-standard units for Creatinine' ;
```

Another common check is to check in the lab file is to check that lab tests (LBTESTCD/LBTEST) have the correct lab categories (LBCAT).

For example:

```
proc freq data=urstudy.lb ;
    tables lbcat * lbtestcd / list missing nopct ;
title1 'urstudy.lb – check lab category with lab test name codes' ;
```

Often, for a data set as important as the laboratory data, a validation programmer will be asked to re-program the LB data set. As SAS programmers know, there are multiple ways to achieve the same results. At the end of the validation program for LB you should provide a PROC COMPARE to compare the production LB data set with that data set created in the validation program.

For example:

```
proc compare base=urstudy.lb
             compare=work.lb ;
    id usubjid visit lbtestcd ;
title1 'urstudy.lb – proc compare with work.lb data set' ;
```

Some programming groups (departments) have a SAS macro in their macro library that scans the SAS log and prints any discovered errors, warnings, and notes. These macros help the validation programmer quickly review the SAS log, though they are not meant to replace a visual inspection of the SAS log.

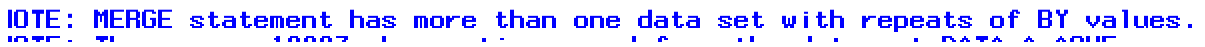
Remember, the sooner the validation for data sets is complete, the less chance for data issues to be found in TLFs. It is helpful to have a validation checklist to follow (sample in Appendix A-1).

VALIDATING OUTPUT (TABLES, LISTINGS, FIGURES)

When beginning the task of validating any output, find the layout of the output in the Analysis Plan. Check each output for the correct output table, listing, or figure number, the spelling in the titles, column headers, row headers, axes, and footnotes, and check for correct pagination. In addition, check the order of groupings within the output as the layout sample may have groupings in a different order.

Initially, visually check the output from the production program. If there are columns of data, are they aligned correctly? Are there extra spaces within parentheses? Is there variability in the number of spaces of column indenting, even within the first column? Does the page number print correctly on all pages? Even if there are hundreds of pages, a validation programmer should visually inspect this. This is especially true when the number of pages increases to the next higher number of positions, e.g., from ‘Page 99’ to ‘Page 100’ or from ‘Page 99 of 243’ to ‘Page 100 of 243.’

Visually check subject data listings for content. It has happened, in a vital signs listing, that a faulty merge statement produced output where all subjects possessed the same height and weight. Other than in the ‘merge’ SAS log statement, these were not flagged as errors and the production programmer obviously did not check his/her output. There are rare occasions when the merge note in the SAS log is acceptable. If a programmer has this message in the SAS log, both data sets in the MERGE statement must be thoroughly checked to ascertain that this message is ok. Because many companies have SAS log checking programs, this message would be flagged. If so, a way to remove this duplicate message is to use a PROC SQL join statement instead. This note in Display 2 and the note in Display 1 are generally the only note statements in the SAS log that are not acceptable.



```
IOTE: MERGE statement has more than one data set with repeats of BY values.
```

Display 2. Generally undesirable MERGE statement in SAS Log

A validation programmer may be asked to write a program to duplicate the output. Another option is to create a work data set which will be compared to a SAS data set which is going into the output reporting step. In either case, it is tempting to copy and paste from the production program. This is not validation. Because ‘drag and drop’ gives results identical to the original program, which may be wrong, you must write validation code from scratch if you are to be effective. As in validating data sets, there are many ways to achieve the same output using SAS software. The original programmer(s) is/are unlikely to have intentionally made mistakes. However, since s/he/they did not intend to err, they are not likely to uncover their own mistakes if/when they proof their own work. That is why you will be assigned the task.

It is helpful for those starting to follow a validation checklist (sample in Appendix A-2).

HAVE ISSUES BEEN IDENTIFIED?

It is extremely rare to validate derived data sets and/or output and not have programming or data issues identified. Most companies have a procedure in place, described either in Standard Operating Procedures (SOPs) or Operating Guidelines (OG), which explain what must be done to rectify the errors. All validation procedures are similar and it is imperative that they be followed until complete resolution has been made and the data set or output that is being validated is truly free of all programming and/or data issues.

The validation programmer will work with statisticians, clinical data managers, other programmers, and other project team members if issues have been identified. Sometimes identified issues are on the validation side and sometimes they are on the production side. If data issues have been identified, the clinical data managers may have to update the raw data based on responses to data clarification forms. Perhaps the specifications in the Statistical Analysis Plan were not clear or were misleading, so the project statistician may have to update the document.

CONCLUSION

The assignment of validating derived data sets and/or table, listing, and figure output has been successfully completed. Each biopharmaceutical company and CRO has documentation of completing validation. There may be checklists to complete, spreadsheets to complete, and additional data sets and/or output to validate.

Validation provides conclusions about the safety and efficacy of drugs and devices. All of the data and the output must be correct or the clinical trials conclusions may be misleading or false.

REFERENCES

Howard, Neil. 2004. “Beyond Debugging: Program Validation,” Proceedings of the SAS Users Group International 2003 Conference, Seattle, Washington. Available at http://www2.sas.com/proceedings/sugi28/058-28.pdf

ACKNOWLEDGMENTS

The author would like to thank Neil Howard, John Gerlach, Angela Ringelberg, Stephanie Lopatosky, Annette Bove, and Debora Knepp for their wise counsel and their commiseration in validating clinical trials.

RECOMMENDED READING

- Base SAS® Procedures Guide
- CDISC SDTM Implementation Guide
- 21 CFR 11

CONTACT INFORMATION

Your comments and questions are valued and encouraged. You may contact the author at:

Name: Susan Fehrer Coulson
Enterprise: BioClin, Inc.
Work Phone: 609.351.3302
E-mail: susanfehrer@yahoo.com

Appendix A-1

Data Set Validation Checklist Sample

Task	Completion Date	Validation Programmer	Comments
Read protocol, aCRF, SAP			
Check SAS LOG for errors, warnings, notes			
Check that all formats and subgroups conform to the SAP			
Confirm all data values accurately represent the raw clinical data			
Check that all mathematical algorithms detailed in SAP have been correctly applied			
Check that all decimals have correct precision as per SAP			
Check derived data values against raw data for subject sample to ensure correct derivation			
Check data points for values exceeding expected ranges, if appropriate			
If dual programming is required, include PROC COMPARE to run with production version of data set			
If SDTM data sets:			
Check order of variables in data set matches order specified in IG			
Check that all required variables are present and populated			
Check that all expected variables are present			
Check that all variables have correct assignment, numeric or character			
Check that controlled terminology has been correctly applied, where necessary			

Appendix A-2

Output Validation Checklist Sample

Task	Completion Date	Validation Programmer	Comments
Check TLF number from Analysis Plan Table of Contents (TOC)			
Confirm that all titles and footnotes conform to the SAP			
Confirm that all column headers conform to the SAP			
Confirm that all row text conforms to the SAP			
Confirm that order of the output conforms to the SAP			
Confirm that the output format conforms to the output shells in the SAP			
Confirm that all range categories, subgroups, and abbreviations conform to the SAP			
Confirm that statistics, if any, have correct numeric precision as per SAP			
Confirm that there are no duplicate rows of data			
Confirm that data fields are not truncated			
Confirm that pagination is correct			
Confirm demographic subject counts across related TLFs			
If dual programming is required, check SAS LOG for warnings, errors, messages of uninitialized variables, ‘merge’ statement			
If dual programming is required, run PROC COMPARE with data set produced before output producing procedure			
Check data for values outside expected ranges, where appropriate			