PharmaSUG 2018 - Paper SI-02

Risk-based Validation in Clinical Trial Reporting: Focus on What Matters Most

Amber Randall and Bill Coar, Axio Research, Seattle, WA

ABSTRACT

Quality control is fundamental to ensuring both correct results and sound interpretation of clinical trial data. Most validation procedures are a function of regulatory requirements, industry standards, and corporate philosophies. However, in the current environment of lean work forces and increased workload, the traditional approach of 100% independent programming may no longer be feasible. Further, we would argue that full double programming may not be the best approach for maintaining high quality. Instead, we propose the implementation of a risk-based validation system in which the validation method is chosen individually for each piece of a project. Risk assessment is based on matrix of factors and the outcome guides the assignment of a testing methodology appropriate for that particular risk level. This ensures that higher effort and rigor is focused in areas that matter most.

In order for risk-based approaches to work, it is critical that someone with the appropriate experience be involved in the initial risk assessment and that the risk level and testing methodology can be modified after initial programming is performed to address unexpected circumstances. It is also imperative to develop metrics to evaluate the system's effectiveness. Key information needs to be captured throughout the programming and testing process to improve risk assessment and assignment of appropriate testing methodology.

We will discuss our proposal for the application and implementation of a risk-based validation system and explore some of the tools that make such an approach possible.

INTRODUCTION

The reason for validation of SAS® programming is simple: important decisions are being made based on results of analyses generated by SAS programs. There are many approaches to validating a SAS program, or verification of these results, but not all employ the same degree of rigor and thoroughness. And this may be OK. It could be surmised that some results may not require the absolute highest level of attention as they may not have any impact on important decisions being made, or perhaps the risk of error occurring is extremely low due to previous validation efforts, or because they may employ only minimal programming syntax which is unlikely to contain errors. The current gold-standard for SAS program validation in the pharmaceutical industry is double programming where a second (or third) programmer independently reproduces results based on a single set of specifications and using the same raw datasets. This is certainly the most appropriate method for the validation of a primary endpoint or other parameter supporting an important decision, but it may be excessive for verification of a print of raw data, a listing trial sites, or even a filtered version of a previously validated table. We would suggest that a high level of overall accuracy can be maintained and efficiency can potentially be increased if rather than independently reprogramming every element of a report, an alternative, yet appropriate, method of validation is chosen that is commensurate with the risk and impact of error in any given individual SAS output. Time and effort should be focused where it matters most. This is the basis for risk-based validation approaches.

VALIDATION: ESSENTIAL FOR SUCCESS

Clinical trials are expensive and time consuming. Further, volunteers put their health and wellbeing at risk in the effort to provide a medical benefit for society. In many cases, patients in need are waiting eagerly for new treatment options. Given the financial and personal investment in these trials, it is critical that the primary questions are answered correctly and patient safety is vigilantly protected. However, the results of clinical trials are only as good as the quality of the data collected and the quality of the analyses

performed. There are many steps necessary to ensure overall quality in a trial including, but not limited to, trial design, collection database design and build, data collection and monitoring, analysis methods, and programming. However, for the purpose of this paper we will focus on the validation and quality control associated with the programming process that is performed to generate final results from raw data. Good validation of the programming process is fundamental to ensuring sound interpretation of the data.

THE CASE FOR A RISK-BASED APPROACH TO VALIDATION

There are many challenges facing the current clinical trial industry including a trend toward a leaner work forces and increased work load, difficulty in hiring good programmers, tighter budgets, and competition with offshore programming teams. Timelines and resources are often squeezed, but high quality output is still expected. Given these challenges, we are forced to look for ways to improve efficiency while still maintaining the expected high quality. The current gold standard for validation for many companies in our industry is for 100% double programming of all analysis datasets, tables, listings and figures. We would argue that a reexamination of this approach may present an opportunity for increased efficiency while still maintaining high quality.

In an ideal world every data point collected and every program written in the production of a clinical study report would be validated 100% by some sort of independent review and/or reproduction. Those that have participated in this process know that it can be time consuming and seemingly never ending. It is our experience that even when a full independent reproduction of a report has been performed, errors can still arise from data anomalies, spelling errors, formatting errors, incorrect interpretation of specifications, etc. Further, the identification of these remaining errors can seem to take an increasingly long time as compared to errors identified early on in the process. As seen in Figure 1, early in the validation process a relatively small amount of effort can lead to large gains in quality. However, as the proverbial low hanging fruit are quickly gathered, it takes substantially increased effort to reach those final incremental gains in quality. At this point, one might ask whether the identification of these late stage issues is really worth the time spent. How does that cost-benefit assessment pan out? Will those remaining items really affect the interpretability of the report? The answer to that question is different for every study. It is true that a substantial amount of data are collected for every trial. Not all of that data will ultimately be important for answering key questions or decision making. In the big picture some data are more important than others. Is it reasonable to ask whether validation efforts should instead be tailored to prioritize those data that are really the most important to achieve the primary safety and efficacy goals rather than just applying the 100% independent programming concept? Is there a more efficient way to ensure that the important questions are accurately answered?

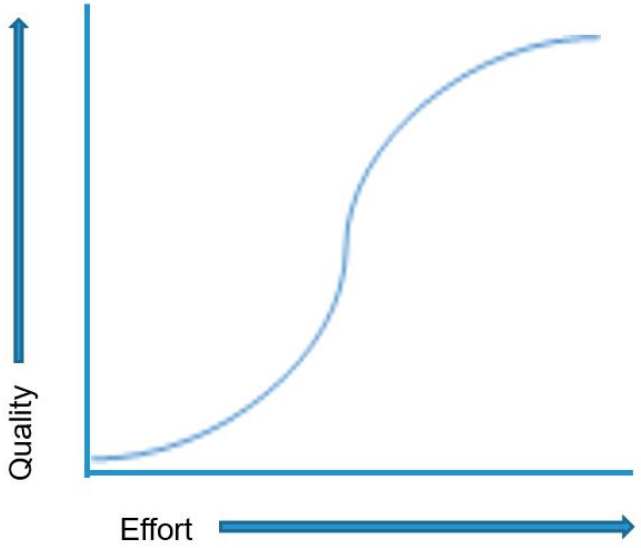

Figure 1. An Illustration of the Relationship Trend between Effort and Quality in Validation

Programming

Perhaps it is time to consider a smarter, more adaptive, and more efficient approach to validation. In an environment with limited resources, mandating double programming in lieu of a more targeted approach could conceivably lead to more errors as programming resources are further strained. A risk-based approach to validation might make more sense where double programming is reserved for areas that hold the greatest impact and risk while less time consuming methods are used in areas that are of lower impact and risk. Certain analyses and programming techniques are more complex and prone to error than others. For example, programming a survival analysis takes a lot more thought and planning than producing a concomitant medications listing. In addition, the potential impact from having an error in a survival analysis is likely much greater than that of having an error in a concomitant medications listing. Should the same amount of effort be put into validating both of these items? We would argue that doing so may prove to be a waste of valuable and finite resources.

Even the FDA recognizes the need for a new paradigm in validation and quality and now supports a risk-based approach to trial monitoring and provides an official guidance for implementing such activities [1]. We would suggest that it is time for statistical programming teams to consider the same. Rather than stretching available resources to double program every item, a risk-based approach should be applied. Double programming will be saved for those items with the highest risk and impact while other less expensive validation methods should be applied to those items where risk and impact are lower. If an adverse events table has already been fully produced and validated, then perhaps a table that is simply a filtered subset of the parent table could be validated using code review rather than by checking every value. If a fully validated report is rerun on new data, could a more efficient validation method be chosen? Perhaps a safety signal emerges midway through a trial that should be further investigated. In this case perhaps a more stringent validation method is warranted since the result may affect future trial conduct. Effort should be made to focus on what matters most. It should be noted that what matters most and which validation methods are appropriate for any given project at any given time should be determined by someone with a high level of authority and understanding, and may change throughout the course of an individual trial.

In designing a risk-based validation system it is important to keep high quality at the forefront of the discussion. Here we will discuss considerations from our own journey to design a process to accomplish these goals.

GETTING STARTED

Designing a risk-based validation system is not trivial. It requires an intense examination of your team's existing process and needs. We have defined a set of steps here describing tasks that we find to be critical to initiating a successful plan.

STEP 1: DEFINE YOUR PROGRAMMING PROCESS

Prior to developing any sort of validation system, it is often helpful to diagram your entire programming process. This helps to identify areas where validation needs to occur as well as steps where critical decisions must be made and items that should be documented.

The workflow shown in Figure 2 is a common example that you might see in a pharmaceutical programming environment. It allows for iterative deliverables such as DMC reporting, interim analyses, or other systematic and regular reporting and monitoring.

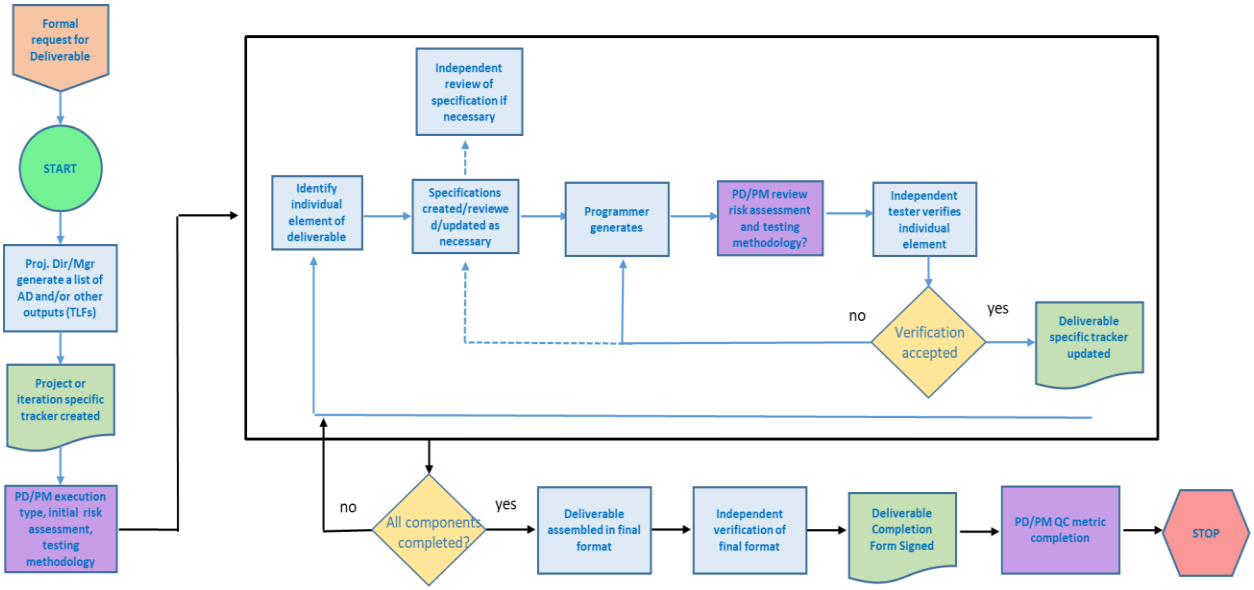

Figure 2. An Example of a Detailed Statistical Programming Process Flow

STEP 2: DEFINE THE DELIVERABLE COMPONENTS

Following a formal request for a deliverable, the Project Manager (PM) should define the type of deliverable and as well as its individual components. This includes a set of SAS programs necessary for the creation of the deliverable. This initial list may be amended throughout the process. This helps the PM develop a more clear understanding of the SAS programming efforts that may be required, and should help with requesting appropriate resources necessary to complete the deliverable in the necessary timeframe. Figure 3 shows a list of programs that might be used to create a set of tables, listings and figures from a set of analysis datasets.

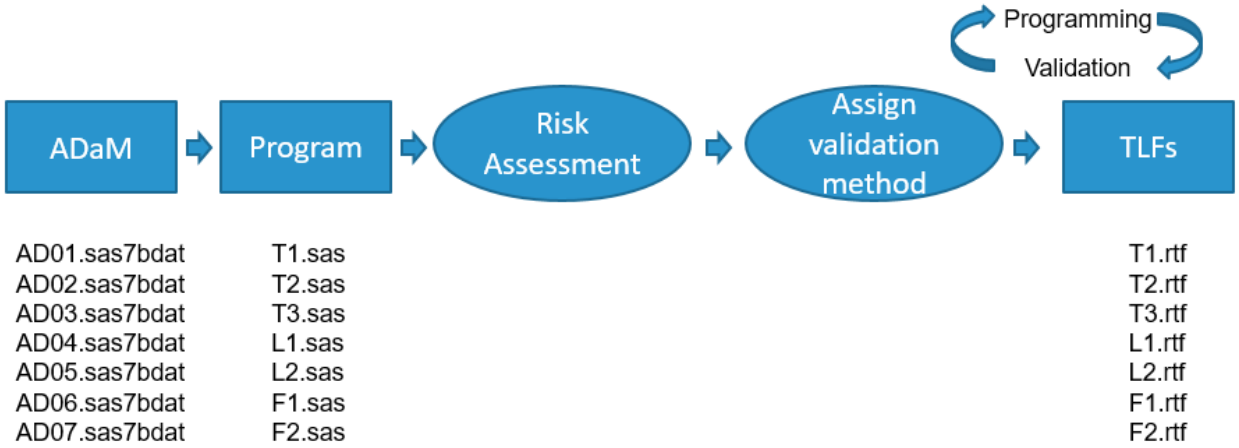

Figure 3. Define the Components of the Programming Process that Require Validation

Initial identification of the SAS programs also allows the PM to generate a deliverable-specific SAS program tracker to hold entries for the initial set of SAS programs. For each SAS program, the PM will indicate the type of execution, where execution type is defined by:

- Run of new dataset
- Run of new report element
- Re-run of existing dataset program w/o modification
- Re-run of existing dataset program with modification
- Re-run of existing report element program w/o modification

- Re-run of existing report element program with modification

The purpose of defining the execution type is to assist with the assignment of risk level as described in the next section as well as clarifying expectations with respect to specifications for each program. For example, re-running an existing program without modification should only require the specifications be reviewed. Rerunning a program with a modification requires that both the modification (and reason) be clearly documented in order to provide the necessary information for the initial and validation programmers. Minor or cosmetic modifications may only need to be documented in the tracker. More significant modifications may require that the original specifications documents be updated. This will depend on your own organization's process flow and SOPs. The tracker used in this context for tracking risk and method should be specific only to the current deliverable, and should not include a complete history of modifications. It should only include the modification (and reason) with respect to a previous deliverable.

STEP 3: ASSIGN A RISK LEVEL FOR EACH ITEM

Assessment of risk for each SAS program should be done by someone with the appropriate knowledge of the project, the data, and the programming efforts required. For the purpose of the paper, we will call this role the Project Manager. Someone in this role will likely be a senior-level statistician or programmer. The PM is encouraged to work with lead and validation programmers to assign risk levels to SAS programs in a collaborative manner. The same collaboration may also be used to identify appropriate methods of validation.

Assessing Risk

Risk is defined to be a function of three components:

- Overall impact on the interpretability of the deliverable
- Complexity of underlying data
- Likelihood of programming error

Risk ultimately falls into three categories (low/moderate/high) though the risk score provides more granular quantification within each category. We propose using a weighted average of each of these components to generate the final risk score. The risk score does not mandate a specific validation method, but instead should be used to help guide the decision.

Implementation

Depending on the platform used to develop your project tracker, existing functionality maybe be available to automate the calculation of the risk score. A spreadsheet, though arguably not the best option for this task, provides this functionality and is generally available to all teams. To implement, the PM first defines the weighting scheme and assigns a score of 1, 2, or 3 to each of the three categories for each program. If available, your tracker may automatically calculate the risk score and can be manipulated to color code accordingly if that is a function that might be useful for your team. In Figure 4, the second row (in white) consists of the weights whereas rows 3-7 would be the component scores for 5 different SAS programs.

	Potential Impact	Complexity	Likelihood of Error	Risk Score
Weight	4	1	3	
Program 1	3	3	3	3
Program 2	3	2	2	2.5
Program 3	3	2	1	2.125
Program 4	2	2	2	2
Program 5	2	1	1	1.5

Figure 4. An Example Risk Assessment Matrix

In this example, Potential Impact receives a greater weight as it is associated with error that may impact the overall interpretability of the results. Complexity of the underlying data receives a lower weight because even when the data are complex, errors generally occur in the programs that manipulate data. As such, more weight is given to Likelihood of Error. Your team should choose weighting based on what is appropriate for your own needs and environment.

STEP 4: CHOOSE A VALIDATION METHOD THAT IS APPROPRIATE FOR THE RISK LEVEL

Once the Risk Score for each component has been defined, the PM should use that information to assign an appropriate validation method. Your organization should first define a list of acceptable validation methods that are appropriate for the type of work that you do. This should include a general guidance on how to apply and perform each one as well as which would be most appropriate for each risk level. The assigned validation method or methods should be clearly documented in the tracker and a validation programmer should be assigned.

Below is a list of potential validation methods that you may consider.

Independent reproduction

This is the most rigorous of all validation methods. A second programmer or statistician completely and independently reproduces an analysis data set or all of the cells in a TLF. Comparison may be done electronically (for longer tables) or manually (for shorter tables) or via the COMPARE procedure for datasets. This would be chosen for high risk items.

Independent code review

For programs that are generally straight forward, involve little data manipulation, and have low risk, it may be appropriate to have a separate programmer or statistician review the SAS code used to generate the report element. Focus should be on a complete header, appropriate datasets and variables, appropriate sub setting, and adequate comments throughout.

Log search

All SAS logs should be searched for potentially concerning messages such as warnings and errors. Messages generated by defensive programming strategies are appropriate findings when searching a log file.

Review for consistency within the report

Sometimes data are displayed in multiple outputs. For example, there may be an overall adverse events summary that summarizes the first row of the other adverse events tables (by SOC and PT). In cases such as this, the overall AE summary should be consistent with the other tables. Another example may be a summary table of Kaplan Meier statistics that may also be displayed in a figure. The summary table may be used to aid in testing of the figure.

Review for consistency with other reports

There may be cases where summary tables are generated external to your programming team. These can be viewed as independently generated reports and may be used for comparison or validation.

Review for expected changes with previous report

In a setting where reports are generated iteratively, it may be the case that the expected changes from a previous iteration are known. For example, the number of new death events may be known (from external communication or dataset compares) so the corresponding summary should reflect these changes.

Review for consistency with other datasets

There may be times where information is recorded or referenced in multiple places, such as death information in ADSL and ADTTE. This may be considered in addition to independent replication in hopes of ensuring consistency in subsequent reporting.

Review for expected changes from previous dataset

In an iterative setting, minor updates may be expected with new data. It may be appropriate to save the current version prior to re-running with new data. Electronic comparisons can show that the expected changes occurred and that nothing unexpected occurred. If the expected differences are very minor, then a simple review of the changes may be appropriate.

Compare against previous version

There may be times where the output may have no expected changes. Testing may be greatly simplified by showing that nothing changed from the previous report. Electronic methods may be available to facilitate this depending on your output formats.

Manual compare against source

There are times were data are imported from a non-SAS file format such as txt, csv, or excel. If the file(s) are small enough, then the converted SAS file can simply be manually compared against the original source by a second programmer or statistician rather than independent reproduction of the SAS format file. This can be an important step to ensure that there are no systematic errors when the data are imported into SAS. Another example may be the visual comparison of a small dataset with few events to a table showing those few events as may be the case with a table showing overall number of deaths in a study where death is unexpected.

Electronic compare of reporting dataset

This should be the gold standard for testing summary tables. If an independent tester can re-create a similar dataset as a reporting dataset from execution of the REPORT procedure, then all cells in the body of a table can be electronically compared. If the tester uses the WARNING option in PROC COMPARE, than any differences will be identified via a standard log search. With this approach, tables can be independently verified with every iterative execution without straining resources.

Visual inspection of figure against reporting data

There may be times where a tester of a figure need not completely re-generate a figure to validate. Custom graphics may be difficult and time consuming to produce. Nothing is gained by a tester trying to exactly reproduce the figure. Emphasis may be on visual inspection against a testers own reporting data and/or comparison against a reporting dataset from the procedure generating the figure.

Review of titles/footnotes

This is intended to be used with other techniques such as independent code review to ensure that the titles/footnotes accurately reflect what is in the output. This should never be the only method used for testing.

Electronic compare of base SAS program

In cases where a base program is to be re-run on subsets (such as the case in repeat tables or listings), then a compare of the SAS program can provide assurance that the only change was in sub setting the reporting data. This complements independent code review.

Special testing instructions

Project specific situations may result in a need for special testing instructions that do not reflect any of the existing testing methodologies. While special testing instructions are acceptable, they need to be justified and documented appropriately.

STEP 5: DOCUMENT YOUR PROCESS AND DECISIONS

Even though we have saved discussion of documentation for Step 5, documentation should be maintained throughout the risk-based validation process. All critical steps and decisions should be documented as well as what was performed, when it was performed and who was responsible for doing it. In an ideal world, this information should be kept in a controlled environment with carefully defined editing permissions. In the practical world, this document may take the form of a spreadsheet. Regardless, your process for maintaining documentation should be well defined and followed.

AD01.sas7bdat	T1.sas	Med	Prog test	T1.rtf
AD02.sas7bdat	T2.sas	Med	Prog test	T2.rtf
AD03.sas7bdat	T3.sas	High	Indep Rep	T3.rtf
AD04.sas7bdat	L1.sas	Low	Code review	L1.rtf
AD05.sas7bdat	L2.sas	Low	Code review	L2.rtf
AD06.sas7bdat	F1.sas	Med	Prog test	F1.rtf
AD07.sas7bdat	F2.sas	High	Indep Rep	F2.rtf

Figure 5. Simplified Example of a Programming Tracker Showing the Determination of Risk and Assignment of Validation Methods

Figure 5 shows an example of a very simplified programming tracker. A well designed tracker would contain a list of items to be programmed, the corresponding risk assessment, the defined validation method, the assigned production programmer and date of program, the assigned validation programmer and date of validation, and outcome of the validation. In the case of iterative reporting, it should also contain notes pertinent to previous and future production runs.

As stated spreadsheets, due to lack of editing control, do not provide the best solution for this process. Historically, there have not been many off-the-shelf software options designed to support process management needs of statistical programmers in the pharmaceutical industry. Some companies have chosen the rather expensive route of in-house development, but that may not be an option for smaller companies. Fortunately, some new commercial options are in development and we look forward to seeing how they function in practice.

STEP 6: PERFORM THE VALIDATION

If your documentation has been meticulously completed, then the validation team should be able to easily and clearly use the tracker as reference to discover exactly what they should validate and how validation should be performed. The validation programmer should follow instructions and document the outcome of each validation. If issues are identified, they should be recorded and sent back to the programming team or PM for review and assessment.

STEP 7: REVIEW YOUR METRICS AND REASSESS PROCESS DECISIONS

One of the most important steps of risk-based validation is review and assessment. Once a round of programming and validation has been completed, it is important to go back to analyze where errors were found, when they were found, and whether your risk assessments were accurate and validation choices were indeed appropriate. Were errors found internally by your validation team or were they identified perhaps by a senior member doing a final review. Did the client identify an error? Obviously your goal should be to minimize the errors that slip through the initial validation. Reexamine your decisions and modify if necessary. It is important that risk-based systems have the flexibility for modification when necessary.

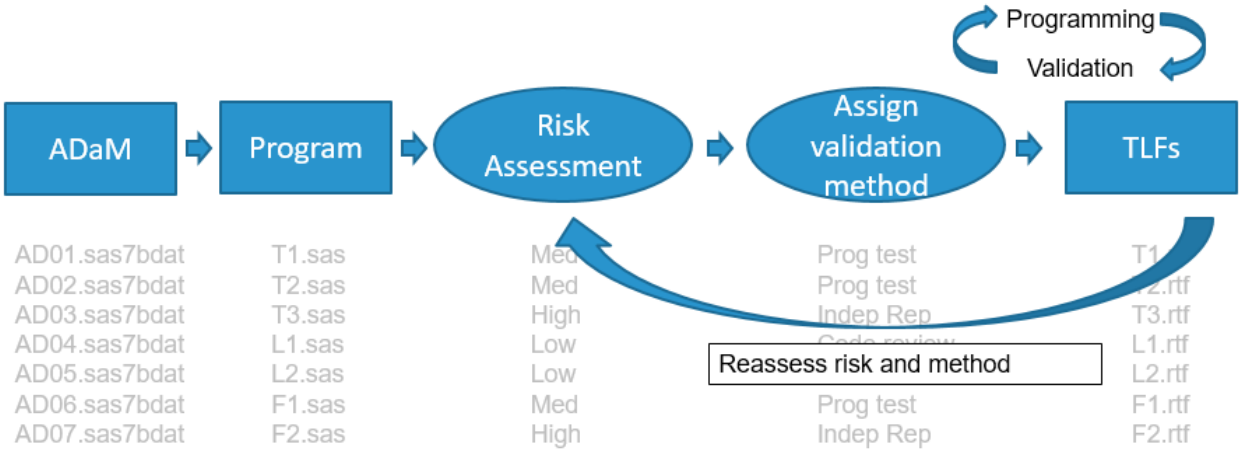

Figure 6. Review and Assessment is one of the Most Important Steps in Risk-based Validation

In the case of iterative reporting runs, as is the case in the production of reports for data monitoring committees, the most important and meaningful content of the report often changes as the study progresses. While medical history and demographics may be of importance in the beginning of the study, careful analysis of endpoints will likely become more important near the end. Perhaps there have been data updates, safety concerns or special requests since the last report run. It is important in this situation to review the risk assessment and validation method for each item and adjust as necessary to reflect the new levels of risk at each delivery.

Further, your assessment should include a more global examination of the errors found in several projects. Maintain metrics on your errors and review your process. If your overall validation process is not working, then take the necessary steps to improve it.

CONCLUSION

Designing a risk-based validation system that maintains high quality is not easy, but we feel that the benefit can be worth the effort. In our attempt to redesign our own validation processes, we have identified a few key factors that are necessary for success. The first is controlled decision making ability. The individual responsible for determining risk and assigning validation methods should be someone with a high level of experience and a thorough knowledge of the project as well as an understanding of the implications of these decisions. This decision should be fully documented and attributed to an individual with a date. Programming and validation personnel assignments should also be fully documented. Further, a secure and controlled audit trail should be preserved.

In order to achieve this controlled environment, you need the right tool. While many teams may choose to use a spreadsheet option, this is not the best solution. It may be attractive because it is affordable and available, but it does not provide a secured, controlled decision making environment nor does it maintain a true audit trail. In the recent past, there were not many process management tools available that really functioned specifically for the clinical trial programming environment. Some larger companies chose to develop their own tools, but this is a very expensive and time consuming undertaking. Fortunately, some commercial software options are becoming available.

Another important function to consider when choosing a tool is the ability to maintain metrics on the outcomes of your risk-based validation decisions. Is your risk assessment working, and are the chosen validation methods sufficient for finding errors before a report leaves your hands? You need to have the ability to assess your process and make changes when necessary. Changes may also be required throughout the life of a project. As time passes, different parts of an analysis may become more critical. The validation plan should be reassessed frequently by someone with a high level of global knowledge of the project.

As our work environment becomes more demanding, we are often met with the challenge to work smarter, not harder. The only way to begin to do that is through careful examination of existing processes

and the identification of areas where efficiency can be improved while still maintaining a high level of quality. We believe that the employment of carefully designed and controlled risk-based validation methods is one approach that may help to achieve that goal. We acknowledge that changing established practices can seem daunting at first, but if achieved, the benefits can be great.

REFERENCES

[1] Food and Drug Administration. Guidance for Industry: Oversight of Clinical Investigations – A Risk-based Approach to Monitoring. Accessed March 13, 2018. https://www.fda.gov/downloads/Drugs/GuidanceComplianceRegulatoryInformation/Guidances/UCM269919.pdf.

ACKNOWLEDGMENTS

We would like to acknowledge Paul Stutzman, Joshua Sanders, and Lynda McVarish for their efforts in strategically redefining our programming process, proposing risk scenarios, and tackling initial risk-based validation work instructions.

CONTACT INFORMATION

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

Amber Randall
Axio Research
amberr@axioresearch.com