How to validate a SAS macro? – *A case study*

Edwin Ponraj Thangarajan | Giri Balasubramanian | PRA Health Sciences, Chennai, India.

ThangarajanEdwin@prahs.com | BalasubramanianGiri@prahs.com

"Establishing documented evidence which provides a high degree of assurance that a specific application program will consistently produce information or data meeting its predetermined specifications and quality attributes."

Abstract

Validation is a process of establishing documentary evidence demonstrating that a procedure, process, or activity carried out in production or testing maintains the desired level of compliance at all stages. This process is extended into all aspects of Clinical Trials including Data Management, Data Conversion and Analysis. The focus of the poster is to demonstrate the validation process that needs to be followed while using SAS macros for performing Clinical Trial Analysis. The validation process adheres to SDLC (Software Development Lifecycle) and needs to demonstrate the compliance towards the guidelines issued by regulatory bodies like US FDA, EMA, MHRA. SAS Macros being used within Clinical Trials are broadly classified into Utility, Reporting, Statistical, Interface, Extraction etc. Each of this classified macro has their own challenges while performing validation work. The poster would address the major challenges faced while performing validation and provides solution to overcome the same.

- Development & Integration of macros and delivering it with quality and bug free
- Regulatory adherence
- System Compatibility

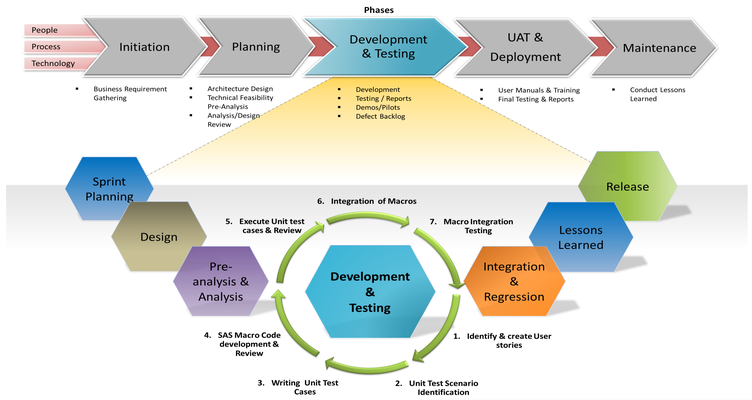
Our SAS Macros

SAS® Macro Language is a powerful feature for extending the capabilities of the SAS System. The Macro Language has its own rules for using various statements and parameters. Validating SAS macros is really complex based on their nature of functionality, business rules and debugging in different ways using different options. Our SAS macros are developed & validated based on different categories.

Macro Categorization	Regulatory Compliance	Software/System Compliance
• Import/Export Macros • Edit Check Macros • Template Macros • Analysis Macros • Conversion Macros • Integration Macros • Utility Macros	• CDISC SDTM • CDISC ADaM • Define-XML • ODM-XML • Dataset-XML • OpenCDISC Validations • FDA – SDS • ICH, GxP, Analysis & Reporting (TFLs)	• Windows Servers 2003, 2008, 2012 • SAS 9.2, 9.3, 9.4 • SAS/CONNECT, SAS Grid • .NET, JAVA • Microsoft Office 2003, 2010 • DataLabs, Medidata Rave, Oracle Clinical • Oracle, Microsoft SQL Servers

Validate SAS Macro in Agile

The phases in agile are iterative and the process may return to a previous phase at any point as required. Each phase has one or more outputs that are natural outcome of following a validation process that consistently produces high-quality system solutions that meet the needs of the business. The outputs that provide evidence of this process, for the purposes of audit, inspection & regulatory compliance, are compiled into a validation package.



Identify & Create User Stories

User stories are characterized as independent, negotiable, valuable, small and testable representing a small piece of a functionality that a team can deliver in an iteration. User story creation involves,

- Requirement understanding & analysis
- Development & Validation team discussion
- User story creation
- Task identification and estimation

Unit Test Scenario Identification

- Identifying sequence of events when everything goes as expected *(Happy path)*
- Identifying an alternative set of steps that run parallel with part of happy path *(Alternative path)*
- Identifying result of a step in happy path that fails to produce successful result *(Exception path)*

Writing Unit Test Cases

Unit tests are written by a developer who codes and executes the macro. All the identified scenarios are written into unit test cases and executed with the coverage of at least 90%. The developer codes and unit tests each requirement ensuring that it meets the acceptance criteria defined for that requirement.

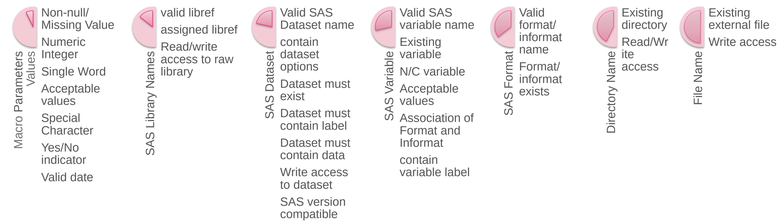
SAS Macro Code Development & Review

Macros are developed with strict adherence to a set of standard guidelines encompassing Design Principles, Best Programming Practices, Regulatory Standards and Unit Testing Framework.

All SAS macro programs are developed with the promise of enhanced readability, improved commenting practices, adhering to naming guidelines while declaring names for global and local macro variables, dataset and its variables and coding compatible to various SAS versions as all our developers are trained with a predefined adherence checklist.

Log Review Keywords		Debugging Options
• ERROR • UERROR • WARNING • UWARNING • IS UNINITIALIZED • MERGE STATEMENT • REPEATS OF BY VALUES • MISSING VALUES WERE • DIVISION BY ZERO • NOT FOUND • W.D FORMAT • INVALID • OUTSIDE THE AXIS RANGE • NOTIFY DATA MANAGER • HARD CODING • TRUNCATED TO • UNEXPECTED VALUE • MISSPELLED AS	• NO OBSERVATION • INTERACTIVITY DISABLED WITH • MATHEMATICAL OPERATIONS COULD NOT • BUT WITHOUT A LENGTH • COULD NOT BE PERFORMED • COULD NOT BE LOADED • DATA SET IS EMPTY • DOES NOT EXIST • EXPAND FORMAT • OR PROVIDE A SUFFICIENT • REPEAT VALUE • RE-WORK ALGORITHM • WILL ASSUME A DEFAULT LENGTH • NOTIFY STATISTICIAN • ST PROBLEM • CP PROBLEM • HARD-CODING	• NOTES • PRINTMSGLIST • SOURCE • SOURCE2 • STIMER • MERROR • MLOGIC • MPRINT • MFILE • SERROR • SYMBOLGEN • DETAILS • MACROGEN • MEXECNOTE • BYERR • DKRICOND • DKROCOND • DSNFERR • ERRORABEND • ERRORCHECK • FMTERR • INVALIDDATA • VNFERR

The developers should care…..



The validation checking, error messages and exception handling are maintained unique and consistent for all the macros. The exception messages are designed to have unique error code, description, resolution, priority and severity. For example, a standard user error message will look like, **UERROR [ERRORCODE]: YY-MM-DDTHH:MM:SS+GMT ERROR MESSAGE**

Execute Unit Test Cases & Review

Developers execute all the unit test cases through Microsoft Test Manager to ensure that the code works as intended. The various aspects covered under this phase are global macro variables, Code testing, input/output parameters code testing, database code testing, macro logic testing, etc.

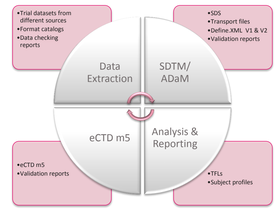
Different debugging options are used to debug the macro on different modes to find and resolve the issues. Error messages generated by these options appear in SAS log are prefixed by the name of option responsible for the message.

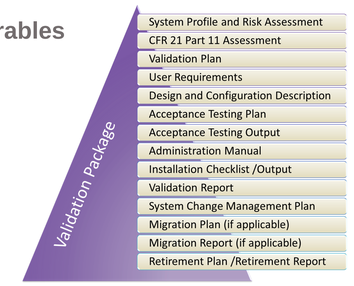
Macro Integration & Testing

Every macro created are individually tested and integrated according to the execution flow. The macro integration is considered as most challenging and need the following:

- Developers and testers are expected to work together during the integration process
- Effective handling of global macro variables
- Accessing library databases and effective cleanup between macros and programs.

Validated Outputs & Deliverables





Conclusion

Validation process presented ensures high degree of quality assurance with evidence in meeting the requirements thereby adhering to the regulatory standards. In addition, the agile approach allows fewer changes & easy to fix, flexibility to incorporate new requirements, better communication and enabling team accountability.

PRAHEALTHSCIENCES