

PharmaSUG 2023 - Paper SD-264

R Package Quality & Validation: Current Landscape

Phil Bowsher, Posit/RStudio PBC

ABSTRACT

Many organizations are migrating to an open-source backbone for the next-gen Clinical Trials environment. Quality teams often have questions about implementing the open source environment such as managing R packages and processes around validation/qualification/verification. The term Validation can have many meanings but usually reflects the internal processes within an organization and can be implemented differently per the interpretation of guidance. Pharmaceutical companies will often ask for sample Standard Operating Procedures (SOPs) for validating open source software. This can be a challenge as often each pharma is unique in the execution of SOPs. The information below can be used to aid the Quality Teams in their unique expectations and SOPs around validation.

This paper will highlight the current landscape and discuss key information for program leaders and quality professionals implementing the software and overseeing the change management. Many of the key items discussed in the paper can be found at:

<https://solutions.posit.co/envs-pkgs/environments/validation/>

INTRODUCTION

Statistical Programming and Clinical Quality Assurance (QA) teams are often looking for guidance and advice about R and open source languages used in a clinical trials setting. Control, management and documentation are critical efforts for the clinical environment as they help to establish the dialog and communication with reviewers on the regulatory side.

R and open source are dynamic and pharmaceutical companies are looking to establish the change control needed for managing the environment. Often the implementation of the open source tooling are in a Server environment either within the organization or in the cloud (EC2 VPC based etc.). In this paper we will look at the current information for taking a risk-based approach to controlling these environments for clinical reporting and submissions. This includes environments for early stage clinical trials into late stage work such as ADAM data prep, TLFs and submission.

Approaching validation uniformly can be challenging as many organizations have different internal processes and definitions of Validation. There are 3 main areas that come up in conversations around quality and validation when establishing a risk-based approach. This paper will break these topics into 3 parts:

1. Posit Team Software
2. R Language
3. R Packages and functions

The first major recommended action item is to create a **R version and Package plan** that will define how the components below are implemented within an organization. This plan will be used as the guiding document to provide clarity for team members about various topics such as adding R versions and the process for adding packages to the repositories.

https://colorado.posit.co/rsc/Transitioning_R_FDA_Submissions/Transitioning_R_FDA_Submissions.html#6

<https://solutions.posit.co/envs-pkgs/environments/reproduce/>

The plan can be referenced during an inspection and can help in complying with requests for information as it will bring clarity throughout the organization. For reference, Project Management methods by Merck are below:

<https://www.pharmasug.org/proceedings/2021/SI/PharmaSUG-2021-SI-083.pdf>

Many public talks have documented the use of R at the *R in Pharma* conference and are publicly available:

<https://rinpharma.com/>

Many organizations are migrating to an Open Source backbone for clinical trials such as detailed in the Roche video below:

<https://posit.co/blog/roche-shifting-to-an-open-source-backbone-in-clinical-trials/>

Merck also has information about the strategies implemented:

Analysis and Reporting in Regulated Clinical Trial Environment using R

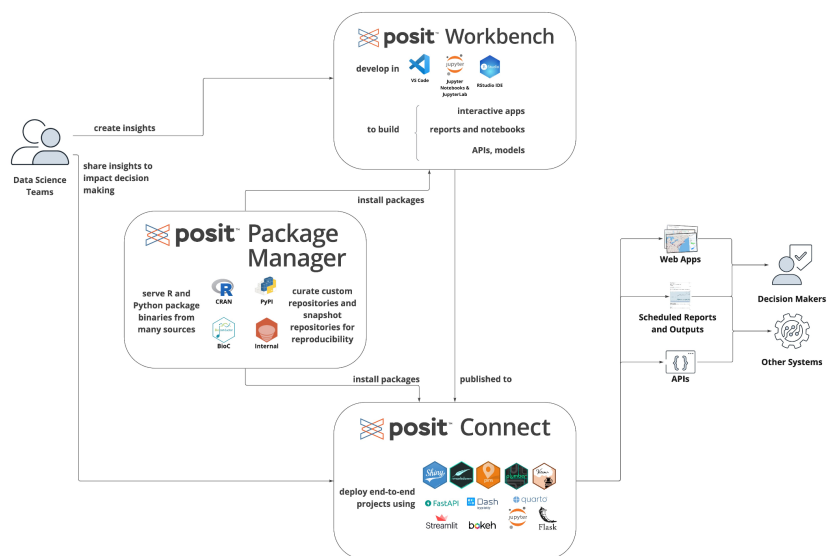
<https://www.pharmasug.org/proceedings/2021/AD/PharmaSUG-2021-AD-079.pdf>

<https://www.lexjansen.com/phuse-us/2020/tt/TT12.pdf>

The focus on this paper will be on Software Validation and qualification. Statistical and numeric validation is an important topic but beyond the scope of this paper.

PART 1 - POSIT TEAM

Many pharmaceutical companies setting up the Open-source backbone will use Posit Team. Posit Team architecture looks like:



Posit Team consist of 3 software parts:

1. Posit Workbench

Posit workbench is the entry point into the development environment. RStudio IDE as well as Jupyter, and VS Code are housed within Posit Workbench. This is where statistical programmers will build their workflows and content.

2. Posit Connect

Posit Connect is the production environment that will host content such as Shiny apps, APIs, workflows etc. Statistical programmers will publish their final work to Posit Connect.

3. Posit Package Manager

Posit Package Manager is the tool surfacing packages into the development and production environments. Posit Package Manager is the tool that provides the controls that will manage and house the validated packages within the organization.

The core item of interest for most quality teams is around the management of packages used for clinical reporting. Posit Package Manager is a key component of creating and reproducing the environments used for clinical reporting as it houses the Packages as defined by IT and Quality within an organization per the Strategy document.

Posit provides a guidance document for the Posit Team software stack of Workbench, Package Manager and Connect here:

https://rstudio.com/wp-content/uploads/2019/06/rstudio_compliance_validation.pdf

More information is in the following slides, "Transitioning to R for Clinical Submissions":

https://colorado.posit.co/rsc/Transitioning_R_FDA_Submissions/Transitioning_R_FDA_Submissions.html#1

The Installation Verification documents for Posit Team are below:

Posit Connect:

<https://docs.posit.co/rsc/post-setup-assistant/#verification-tasks>

Posit Package Manager:

<https://docs.posit.co/rpm/installation/#step-3-verify-installation>

<https://docs.posit.co/rspm/admin/getting-started/configuration/#quickstart-cran>

Posit Workbench:

<https://docs.posit.co/rsw/installation/#step-4-verify-installation>

After the review of the Posit Team Validation and Verification documentation (documenting how the software is maintained, trusted and controlled), many pharmaceutical organizations will add Posit as a trusted vendor which adds its software and packages as approved by IT & Quality.

PART 2 - R LANGUAGE

It is well understood that R is a core language used for clinical reporting. FDA discusses its use here:

<http://washstat.org/presentations/20181024/Schuette.pdf>

The software clarifying statement helped bring awareness to the pharma community that any language could be used for submissions:

<https://www.fda.gov/media/161196/download>

Base R is often downloaded from CRAN:

<https://cran.r-project.org/>

In 2018, the R Foundation for Statistical Computing released "R: Regulatory Compliance and Validation Issues: A Guidance Document for the Use of R in Regulated Clinical Trial Environments"

<https://www.r-project.org/doc/R-FDA.pdf>

To date no such document exist for Python but organization have started to use Python for late-stage clinical work as documented here:

<https://github.com/philbowsher/Open-Source-in-New-Drug-Applications-NDAs-FDA#python>

Most Python usage currently is focused on automation as in the Roche environment discussed in the webinar above.

PART 3 A (PUBLIC R PACKAGES) & B (INTERNAL PACKAGES)

Part A - Public Packages

It is important to focus early on package management. There are 2 key things to focus on when it comes to packages in a clinical environment:

1. **Stable repositories inside the organization**
2. **Users documenting the packages used for clinical reporting**

As discussed above, Posit Package Manager will provide the tools to surface out packages specifically selected by the organization for clinical work. Each pharma will have unique processes and you can review an example here from Merck:

External R Package Qualification Process in Regulated Environment

<https://www.pharmasug.org/proceedings/2022/SI/PharmaSUG-2022-SI-057.pdf>

Posit Packages

RStudio in 2020, in collaboration with the community and pharma, released validation guidance documents covering tidyverse, tidymodels, r-lib, gt, shiny and rmarkdown. The main page and direct links are below:

<https://posit.co/solutions/pharma/>

<https://www.rstudio.com/assets/img/validation-tidy.pdf>

<https://www.rstudio.com/assets/img/validation-shiny-rmd.pdf>

These documents were meant to supplement the base R document referenced above and includes guidance for many of the top packages used in clinical workflows such as dplyr, shiny, rmarkdown, etc. Posit maintains many of the most popular R packages and the documentation above covers the core packages statistical programmers will run for managing data and reporting.

Open Source Package Development and Maintenance

Open Source is often a new area for many quality professionals. For R packages, maintenance is open and in public view. This can be very helpful for many Quality Teams as it makes an abundance of information available that is often hidden for commercial software. Below we will review this for the ggplot2 package available here: <https://github.com/tidyverse/ggplot2>

For open source tools, it is important for Quality Professionals to have a high-level understanding of version control (git etc.) and the maintenance of public open source code via tools like github.

The ggplot2 package is maintained by professional programmers at Posit publicly on github. You can see the most active developers here: <https://github.com/tidyverse/ggplot2/graphs/contributors>

The top contributor is Hadley Wickham and qualifications are listed here:

<https://hadley.nz/>

Only approved pull requests will be added to the package source by the maintainers. This means changes to the source code are approved and governed by the maintainers as well as publicly documented.

ggplot2, as of May 2023, has a 83% code coverage on github. This means 83% of the base functions are covered by test. As a general rule, code coverage should be above 50%. The public test for ggplot2 are available here: <https://github.com/tidyverse/ggplot2/tree/main/tests>

Periodically, R developers will submit their package to CRAN, the Comprehensive R Archive Network. The CRAN site describes CRAN as: "... a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R."

In addition to distributing R, CRAN is the primary package repository in the R community. A team of individuals with the R Foundation approves packages submitted and added to CRAN. The R Foundation is a not for profit organization founded by the members of the R Development Core Team. The R Foundation, among other things, works to provide support for the R project and other innovations in statistical computing and hold and administer the copyright of R software and documentation.

Getting a package added to CRAN is no small feat (pass R CMD CHECK and survive the review process). CRAN maintainers test packages across a matrix of R versions and operating systems. The review process is outlined here:

<https://cran.r-project.org/web/packages/policies.html>

https://cran.r-project.org/web/packages/submission_checklist.html

Any package accepted on CRAN must pass a series of automated tests:

<https://cran.r-project.org/doc/manuals/r-release/R-exts.html#Checking-packages>

The automated tests enforce the CRAN submission policies. The results of these tests are available on CRAN for each package:

http://cran.us.r-project.org/web/checks/check_summary_by_package.html#summary_by_package

Here are the results for ggplot2.

https://cran.r-project.org/web/checks/check_results_ggplot2.html

The same tests can be run locally against any package source tar file using a command like: R CMD check package_0.1.tar.gz

The actual test code is part of R itself, not part of the package being tested. CRAN checks are run for the package plus the latest release of R, patch releases, and the current development branch of R. The tests are run on a variety of operating systems including Linux, Windows, and Mac OS.

In addition to the CRAN checks, many package authors have added their own tests which help ensure that the package's functions work as intended. Most often, these tests are built into the package in a specific way so that whenever the automatic CRAN checks are run the custom tests also run.

These additional tests are part of the package source and for more information please review:

<http://r-pkgs.had.co.nz/tests.html#test-cran>

Packages added to supplement base R are often referred to as "contributed packages" and are available for download directly from CRAN. You can read more on this topic here:

https://github.com/philbowsher/phuse-2019-r-packages/blob/master/How%20Do%20I%20Pick%20a%20R%20Package%20for%20My%20Clinical%20Workflow_.pdf

Validation Hub's Risk-Based Approach guidance

In addition to the Posit Packages (ggplot2 <https://posit.co/products/open-source/rpackages/> etc.), pharmaceutical organizations **will also add other packages** to the baseline of packages to be used by statistical programmers. Organizations will often select other packages to be included based on the **R Validation Hub's Risk-Based Approach guidance** here:

"A Risk-Based Approach For Assessing R Package Accuracy Within A Validated Infrastructure"

<https://www.pharmar.org/white-paper/>

Metrics in the white paper have been collected in the riskmetric R package to help organizations evaluate the quality of R packages.

<https://github.com/pharmaR/riskmetric>

The package was also implemented visually in a Shiny app here:

<https://rinpharma.shinyapps.io/riskassessment/>

Presentation:

https://www.youtube.com/watch?v=gsWc_oSTb9c

Organizations can run the Shiny app using the code here:

<https://github.com/pharmaR/riskassessment>

Organizations often point out that while the Metrics are helpful, it is important to have a Quality team review the results and then decide if further review/validation is required. For example, the Quality team may want to review the “quality” of tests and documentation for packages as the metrics check for availability. It has been pointed out that assessment by a few pharmaceutical companies could help bring clarity on the metrics as generally acceptable.

Further Validation

A popular question is if regulatory bodies require “validated” packages. The R FDA Submission Pilot used as-is CRAN packages as outlined below. This submission was accepted successfully by the FDA.

<https://github.com/RConsortium/submissions-pilot1/blob/main/renv.lock>

<https://github.com/RConsortium/submissions-pilot1>

In “GUIDELINE FOR GOOD CLINICAL PRACTICE E6(R2)” it states that “The sponsor is responsible for implementing and maintaining quality assurance and quality control systems with written SOPs...”

https://database.ich.org/sites/default/files/E6_R2_Addendum.pdf#page=28&zoom=100,89,429

Much is left to be interpreted and pharmaceutical companies have different processes and SOPs. Some organizations may want to provide further validation on top of public packages based on their own requirements. This is often completed with the testthat package: <https://testthat.r-lib.org/>

Further information can be found here:

https://phuse.s3.eu-central-1.amazonaws.com/Archive/2023/Connect/US/Florida/PRE_OS14.pdf

Community Work is underway by the R Validation Hub to help bring availability of validated packages for clinical trials. This includes support for the test and validation of public packages commonly used. You can follow this effort here:

<https://pharmar.github.io/regulatory-r-repo-wg/>

Pharmaverse

Many of the packages used for clinical reporting come from an R ecosystem called the Pharmaverse:

<https://pharmaverse.org/>

The Pharmaverse is built and maintained by many professional statistical programmers and developers at Roche, GSK, JnJ, Merck and other pharmas. You can read more about it here:

<https://posit.co/blog/pharmaverse-packages-for-clinical-reporting-workflows/>

The Pharmaverse packages are publicly maintained and the test-coverage and code is available such as:

<https://github.com/pharmaverse/admiral/tree/main/tests/testthat>

☰ README.md

admiral

pharmaverse **admiral** CRAN 0.10.1 Test Coverage 97%

ADaM in R Asset Library

Purpose

To provide an open source, modularized toolbox that enables the pharmaceutical programming community to develop ADaM datasets in R.

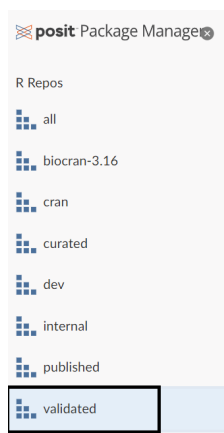
Internal Repository of Packages for Clinical Trials

The documents above, and the implemented risk-based approach, allow organizations to establish a baseline of existing packages that are deemed validated and controlled for internal use.

Organizations can use Posit Package Manager mentioned above to create an internal repository of packages selected specifically by Quality and IT for clinical trials. Steps to do this are outlined here:

<https://github.com/philbowsher/Creating-an-Internal-Repository-of-Validated-R-Packages>

Posit Package Manager can host many repositories within the organization such as a Validated repo:



These repositories can allow various use-cases within Posit Package Manager. These packages then can be surfaced to Windows desktop users as well as Linux Servers. IT can also configure the development environments so that packages are installed automatically by users from the validated environment or surfaced pre-installed into the System library. IT can also create a process to set the appropriate repositories depending on the active version of R selected by users in RStudio. You can read more about best practices here:

<https://github.com/philbowsher/Managing-Multiple-Versions-of-R-in-a-Clinical-Environment>


The Validated repository within your organization is often referred to as the “Shared Baseline” or Curated subset. These packages can also be installed into Docker builds and launched into Kubernetes orchestration layers.

On the clinical side, Posit Package Manager allows for multiple repos for sharing one Validated source with snapshots or Multiple Validated repos for each version of R. Posit Package Manager also provides versioned package repositories (Snapshots) which are a copy of CRAN from a specific date. This enables time-traveling across different package versions so IT and users can recreate environments used on an historic date. IT will often upgrade the packages once or twice a year based on major releases of R.

Repository URL

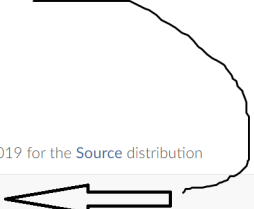
Use the latest packages, or freeze to the packages that were available on a particular date. All dates presented by Package Manager are in UTC to avoid any timezone discrepancies between environments.

December 2018							April 2019						
SUN	MON	TUE	WED	THU	FRI	SAT	SUN	MON	TUE	WED	THU	FRI	SAT
						1	1	2	3	4	5	6	
2	3	4	5	6	7	8	7	8	9	10	11	12	13
9	10	11	12	13	14	15	14	15	16	17	18	19	20
16	17	18	19	20	21	22	21	22	23	24	25	26	27
23	24	25	26	27	28	29	28	29	30				
30	31												

Lock Package Data 

Use this URL to receive packages available from snapshots as of Apr 26, 2019 for the Source distribution

<https://colorado.posit.co/rspm/validated/2019-04-26+1SmyftT3>



Some organizations prefer to have a separate Validated Repo per R version rather than using Snapshots.

renv for Statistical Programmers

To help document the packages used and for collaboration with colleagues, statistical programmers will use a R package called `renv`:

<https://rstudio.github.io/renv/articles/renv.html>

`renv` is a companion package used by users in conjunction with the IT best practices used for managing packages. `renv` creates a file called `renv.lock` that describes the state of the users project's library at some point in time and is created with `renv::init()`. `renv.lock` records repositories, package versions and can be versioned with git! Here is the R FDA Pilot Submission `renv.lock`!:

<https://github.com/RConsortium/submissions-pilot1/blob/main/renv.lock>

The `renv.lock` is "actionable" & supports collaboration as others can use it to recreate the Stat programmers package environment with `renv::restore()`. Moreover, `renv` replicates normal engagement with R & doesn't require users to change their workflow of installing/removing & using packages and requires little technical skill to set up via: `renv::init()` & `renv::snapshot()`.

As Merck writes in the "R for Clinical Study Reports and Submission" book "Reproducibility of analysis is one of the most important aspects of regulatory deliverables" and goes on to explain the importance of `renv`:

<https://r4csr.org/folder.html>

You can read more about `renv` and Posit Package Manager here:

<https://colorado.posit.co/rsc/PPM-renv-Clinical-Reproducibility/template.html#/title-slide>

Part B (Internal Packages)

Quality professionals will often point out that Validation is more than testing. Much has been discussed at the yearly R in Pharma conference about Validation such as the 2022 talk "Roches approach to software validation: an automated pipeline from validation to publication":

<https://www.youtube.com/watch?v=ZfZpypQ1jSM>

Merck documented its processes via these papers:

A Process to Validate Internal Developed R Package under Regulatory Environment

<https://www.pharmasug.org/proceedings/2021/SI/PharmaSUG-2021-SI-084.pdf>

A Strategy to Develop Specification for R Functions in Regulated Clinical Trial Environments

<https://www.pharmasug.org/proceedings/2021/SI/PharmaSUG-2021-SI-074.pdf>

Many organizations create internal functions and or R packages used for clinical reporting. It is important to include these packages under the same organization management strategy for public packages. Often this includes validating your packages to the same level of management you expect with public packages. Organizations will surface internal packages into the internal environments via Posit Package Manager in the same process as CRAN packages. There are various tools to help implement the validation on internal code such as:

Testing R Functions:

thevalidatoR - <https://github.com/insightengineering/thevalidatoR>

valtools - <https://github.com/phuse-org/valtools>

testthat - <https://testthat.r-lib.org/>

Many pharmaceutical organizations have started to use Shiny for internal clinical work and reporting as seen in this Shiny app:

https://williamnoble.shinyapps.io/the_future_of_clinical_tfls/

Testing Shiny apps and R code is an important part of using Shiny for clinical reporting. Below are tools developed for these purposes:

Test and Reproducibility for Shiny:

shinytest - <https://rstudio.github.io/shinytest/>

{shinyValidator} - <http://opensource.nibr.com/shinyValidator/>

rhino - <https://appsilon.github.io/rhino/>

golem - <https://engineering-shiny.org/golem.html>

shinymeta - <https://github.com/rstudio/shinymeta>

TEAL - <http://examples.pharmaverse.org/app/teal/>

ALTERNATIVE APPROACH - BUYING VALIDATED PACKAGES

Some organizations prefer to buy validation documentation for existing open source packages. There are various vendors selling validation tests and documents as well as groups selling services to support validation. One such group is Atorus that provides OpenVal which is a subscription-based repository of nearly 200 validated R packages and growing.

CONTACT & SUMMARY

The information above highlights information for pharmaceutical companies switching to an Open Source environment for clinical trials.. The approaches above are used by many organizations and finding the right path varies greatly from organization to organization.

Phil Bowsheer

phil@posit.co

<https://github.com/philbowsheer>