Paper DS03

# Charting New Territory: R-Based Submissions Made Easy

Veena Terekar, Johnson & Johnson, Mumbai, India

## ABSTRACT

Are you uncertain about developing R programs due to integration challenges in submission packages? This concern is critical when submitting to regulatory bodies like the FDA and PMDA, where meticulous preparation ensures compliance and reproducibility.

This abstract outlines Johnson & Johnson's first experience with R-based regulatory submissions, detailing the challenges faced and our structured methodology. It serves as a step-by-step guide to constructing robust submission packages featuring R programs.

Central to this process is the Analysis Data Reviewer's Guide (ADRG), which includes sections on Program Submission and an Appendix that provide a framework for effective submissions. These sections focus on R environment setup, configuration, directory organization and program execution. With comprehensive instructions, the ADRG enables regulatory reviewers to easily replicate results, enhancing compliance and fostering transparency. This makes R an invaluable tool for efficient statistical analysis in drug development.

## R Mock submission- Regulatory pathway

**Step 1: Internal Trial Team Agreement and approval**

We started with strong internal alignment. Early adopters were identified, Trials from different therapeutic areas and phases particularly those with submission potential

A pilot trial was selected, Trials featuring TFLs developed in R, intended for submission to the FDA, with potential for other health authorities. A hybrid submission approach was presented at the trial submission kick-off meeting. This approach was endorsed at both the trial working group and compound governance levels, and the mock submission proposal was included in the FDA Type C meeting briefing book. Two studies from different therapeutic area were submitted to FDA following hybrid SAS/R strategy.

**Step 2: FDA Approval for Mock submission**

Next, we focused on early and structured FDA engagement. We received written responses to the briefing book, addressed FDA clarifications, and obtained explicit approval for the mock submission approach before execution.

**Step 3: Mock Submission**

The mock submission was then executed via eCTD, supported by ongoing FDA communication and a detailed walk-through meeting to ensure transparency of the R scripts and outputs.

**Step 4: BLA Filing**

This approach proved highly effective. This was the first R submission to the FDA within J&J, there were no information requests related to the submitted R scripts during health authority review, and the mock submission directly contributed to a smooth and successful final BLA filing.

Overall, this case study demonstrates how mock submissions can be used as a proactive risk-mitigation strategy for R-based regulatory deliverables and can be scaled for future submissions.

## Cross-Function collaboration throughout the Journey

The cross-functional collaboration throughout the entire mock submission journey, which was critical to the successful execution of this exercise.

We closely collaborated with the Publishing team to develop a dossier plan specifically tailored for the mock package, ensuring that document structure and sequencing aligned with regulatory expectations.

In parallel, we worked with the Regulatory team on drafting the submission cover letter. Since this was a mock submission and a CSR was not available, we combined the individual RTF documents into a single

PDF, which was attached to the cover letter in Module 1.
From a submission readiness perspective, we submitted a standard eSub package, including trial dummy information in Module 5, to closely simulate a real regulatory submission.
The mock package then underwent a formal eCTD validation process and successfully received confirmation from the FDA, validating both the technical and structural integrity of the submission.
This was a truly end-to-end, cross-functional effort, involving Statistical Programming, Biostatistics, Regulatory, Publishing, and the R Submission initiative team, all working together with a shared objective—delivering a high-quality, submission-ready package.

## Submission Planning: Hybrid SAS/R Programs

This section explains how we approached submission planning using a hybrid SAS and R programming strategy.
Our starting point was to retain SAS for ADaM dataset generation, leveraging its long-standing acceptance and stability in regulated submissions.
At the same time, we made a deliberate shift to R for all Tables, Figures, and Listings, allowing us to explore modern visualization, reproducibility, and efficiency—while still operating within a submission-ready framework.
An important principle in this journey was that all R packages used were open-source. This aligns with the broader industry movement toward transparency and community-driven development, which is also reflected in Johnson & Johnson's open-source journey in R.
From a compliance perspective, all R packages were validated following JNJ GxP standards and executed within a GxP-compliant R environment. This was critical in demonstrating that open-source does not mean ungoverned—it can be both flexible and compliant.
In terms of tooling, we focused on a small, well-understood package set, including tidyverse components for data manipulation, ggplot2 for visualization, and epiR and survival for statistical analyses, supported by utility packages such as envsetup for environment control.
The key learning from this approach was that a hybrid SAS–R model is not only feasible for regulatory submissions but also provides a pragmatic transition path—allowing teams to adopt R without disrupting established submission processes.

## Constructing ADRG for Hybrid SAS/R Submission

To facilitate the seamless execution of R programs by reviewers, the Analysis Data Reviewer's Guide (ADRG) plays a pivotal role, particularly sections 7 (Submission of Programs) and 8 (Appendix). These sections must detail a comprehensive roadmap for setting up the R environment and running the programs.

### A. ADRG: Section 7: Submission of Programs

Section 7 of the ADRG focuses on the submission of programs, and its primary purpose is to provide transparency and traceability for everything that was executed to generate the analysis datasets and outputs.
We structured this section to clearly separate different program types,
starting with Section 7.1, ADaM programs. These programs describe how the ADaM datasets were created from the source data, including key inputs, macros, and functions used. This allows reviewers to understand both the data flow and the derivation logic.
Section 7.2 covers the analysis output programs, which were used to generate the Tables, Figures, and Listings. Each program is documented with its purpose, inputs, and dependencies, ensuring that outputs can be reproduced independently.
In Section 7.3, we document macro programs. Rather than embedding complex logic repeatedly, common derivations—such as baseline flags or dose-adjustment logic—were modularized into macros. This improves consistency, reuse, and maintainability across programs.
Section 7.4 lists all packages used, including package names, versions, and brief descriptions. This is especially important in a hybrid SAS–R environment, as it supports reproducibility and aligns with regulatory expectations around software transparency.

Finally, Section 7.5 documents common R functions used across programs. By explicitly listing these functions, we make it easier for reviewers to understand reusable components and assess how open-source functionality was applied within a controlled framework.
Overall, this section demonstrates how ADRG documentation can go beyond compliance—serving as a roadmap for reviewers and enabling efficient traceability from datasets to outputs.

### B. ADRG: Appendix: Set-up R -environment and Analysis

This appendix was included to facilitate independent reproducibility of the R environment used for our submission.
The goal was to ensure that FDA reviewers could recreate our analysis environment regardless of the sponsor's internal computing system.
Appendix provides a structured, step-by-step guide. It begins with installation of R and RStudio, followed by defining the working directory and setting up the project structure to mirror the sponsor's computing environment.
Let's see Step-by-Step Instructions:

1. **Set Up the R Environment:**

- Specify the R version and RStudio (or other IDE) used during program development.
- Provide detailed guidance for installing R and RStudio on the reviewer's environment.

2. **Define Working Directories:**

- Document how to organize project files and establish directory structures for inputs, scripts, and outputs.

3. **Install Required Packages:**

- List all the R packages used and ensure their versions are specified explicitly.
- Include instructions to install the required packages using the renv package for creating similar environment.

4. **Create renv.lock File:**

- Use the package renv to capture the project's package versions in a renv.lock file.
- This ensures identical environments can be recreated.

5. **Set Up Analytic Paths:**

- Define and document paths necessary for data access and program execution.

6. **Program Execution:**

- Provide simplified steps or commands to execute key programs (e.g., using batch scripts or standard R commands).

Steps 2 through 5 guide the reviewer in creating an R project and reproducing the folder structure to ensure consistency with our submission package.
For package management, we simplified the process by centralizing all required packages into a single file — 'packages.txt'. This file contains all necessary R packages mirrored by our validated container environment. Installation is performed programmatically to ensure version control and traceability.

To further enhance reproducibility, we created a renv lock file. This captures the exact package versions used in the analysis, ensuring stability of the R environment when recreated on the reviewer's system.

Finally, paths and data libraries are configured via an environment setup file, enabling seamless execution of the R TFL scripts and reproduction of outputs in RTF format.

Overall, this appendix operationalizes reproducibility — transforming R-based analyses from code submission to a fully reconstructable regulatory-ready environment.

Let's learn more about each step mentioned above.

- **Define working directories:**

We will see now how the working directory is defined to ensure reproducibility and alignment with the ADRG instructions.

The reviewer is first instructed to create a working directory — for example, C:\studyid01csr' — and copy the submission folder, including the m5 structure, into this location.

Within this structure, each folder has a clearly defined purpose.

The 'adam' folder serves as the repository for SAS7BDAT datasets used by the table, figure, and listing programs. These datasets are programmatically converted from XPT files located in the 'xpt' folder.

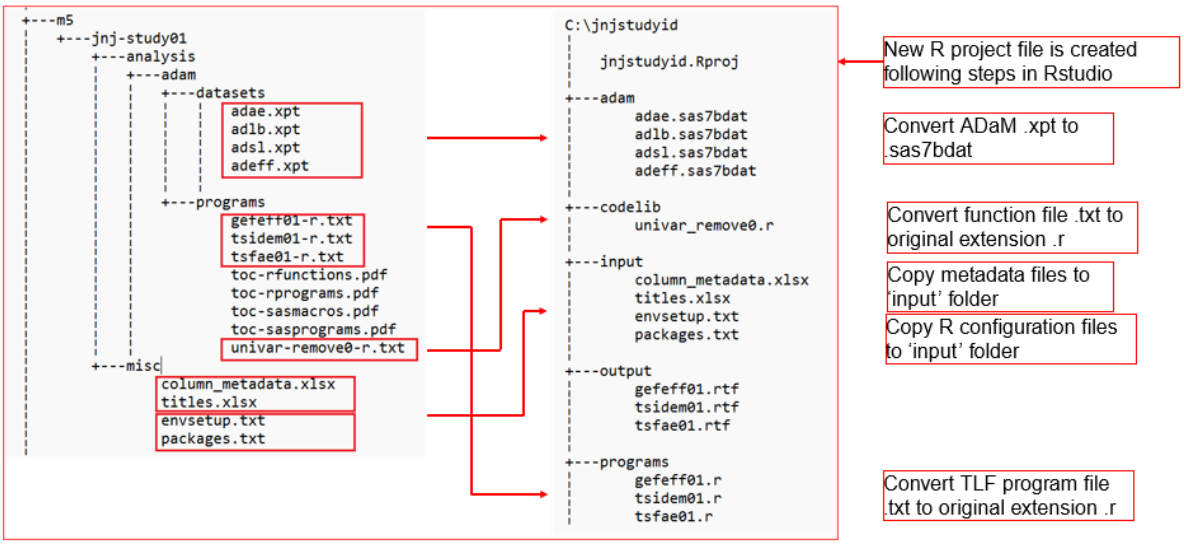
The 'program' folder contains the R TFL scripts, which are converted from .txt format back to their original .r extension prior to execution. Similarly, reusable R function files are restored to their native .r format. To facilitate hybrid SAS/R submissions, we maintained original naming conventions. The programming language is indicated before the .txt extension — for example, adsl.sas becomes adsl-sas.txt, and tsidem01.r becomes tsidem01-r.txt — preserving traceability while meeting submission requirements.

Metadata files are maintained separately to preserve traceability and documentation integrity.

The 'output' folder captures all generated analysis outputs, ensuring a clean separation between source code and results.

The 'xpt' folder acts as a centralized location for datasets copied from different submission components. This streamlines the conversion process from XPT to SAS7BDAT and ensures consistency across analyses.

Below image explains how set up can be done from m5 to working directories:



Overall, this structured working directory enables seamless execution and traceability.

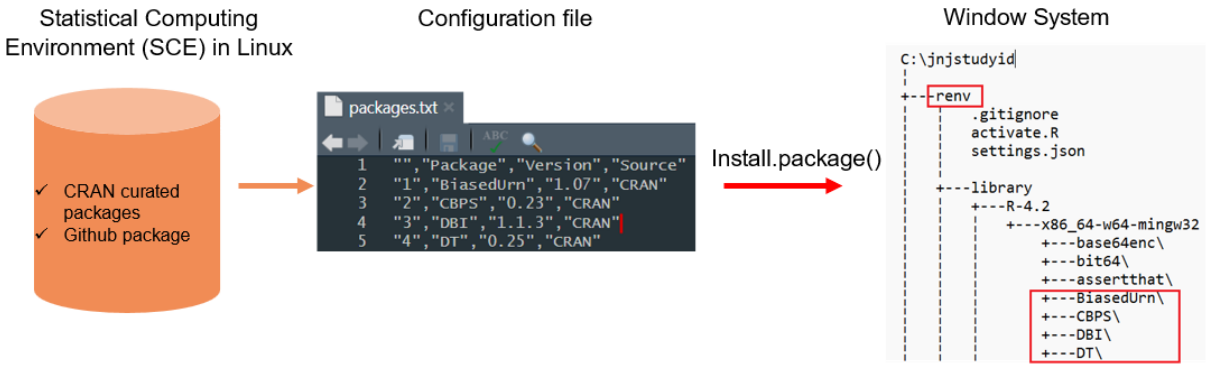
- **Reproduce Environment: Install packages:**

This section explains how we reproduced the R environment by installing the required packages in a controlled and traceable manner.

The starting point is our Statistical Computing Environment in Linux, where all validated packages — including CRAN-curated and selected GitHub packages — were identified. We consolidated these packages into a single configuration file called 'packages.txt'. This file lists the package name, version, and source, ensuring transparency of exactly what was used in the analysis. During setup, the reviewer runs a scripted installation process using install.package(), which programmatically installs each required package from the specified repository.

We leveraged the 'remotes' package to enable installation from remote sources such as CRAN and GitHub, ensuring flexibility when non-CRAN packages are required.

To guarantee reproducibility, we used the 'renv' package. This creates a project-specific library within the renv folder, isolating dependencies from the system-level R installation.




As shown on the right, the Windows environment mirrors this structure, with the renv folder containing activation files and a dedicated project library. This ensures that the exact validated versions — for example, biasedurn, CBPS, DBI, and DT — are installed consistently.

By combining a centralized package list, controlled installation, and renv-based isolation, we created a regulator-ready reproducible R environment.

- **Create a renv lock file:**

This section highlights a critical step in ensuring reproducibility — creating the renv lock file.
After installing all required packages within the project-specific renv environment, we generate a lock file using renv::snapshot(). This captures a complete snapshot of the project's package ecosystem, including exact versions and repository sources.
The renv.lock file is stored at the project root level and serves as the single source of truth for the analysis environment. The lockfile records validated package versions — for example, biasedurn 1.07, CBPS 0.23, DBI 1.1.3 — ensuring that what is executed on the reviewer's system is identical to what was validated in our environment.
This approach delivers four key benefits.
First, it locks down the snapshot of the project's package ecosystem.

Second, it allows multiple reviewers to recreate the same analysis — both now and in the future.
Third, it facilitates seamless sharing of the project between collaborators.
And finally, it isolates project dependencies from the global R installation, preventing version conflicts.
In a regulatory context, this step transforms R from a flexible programming environment into a controlled, reproducible, and submission-ready platform.

- **Set paths for Analysis (envsetup):**

This part describes how we configured path management to ensure seamless execution of the analysis programs.
To standardize execution, we use a configuration file called envsetup.txt. This file defines the project path, input locations, output directories, code libraries, and environment settings.
At project initialization, the .Rprofile file activates renv and calls the envsetup configuration using rprofile(). This ensures that, when the reviewer opens the R project, all required paths are automatically set — without manual intervention.
Within envsetup.txt, we define key directories such as the ADaM data folder, the code library folder, the output directory, and the input folder. These paths are dynamically built using the working directory, which ensures portability across systems.
By centralizing path definitions in a single configuration file, we avoid hardcoded directories inside analysis programs. This improves traceability, reduces execution errors, and simplifies review.
Importantly, we instruct the reviewer to incorporate the path configuration within the .Rprofile file so that the environment is initialized consistently at project startup.
Overall, this approach ensures that the analysis can be executed in a clean, controlled, and regulator-ready manner — independent of the sponsor's internal infrastructure.

## Challenges/Finding in Hybrid SAS/R submission

While implementing a reproducible R-based submission framework, we encountered several practical challenges.
The first major challenge was replicating the sponsor's exact R environment. Even with renv in place, fully mirroring the development environment across systems was not always straightforward.
Different operating systems — Windows, Linux, or Mac — require different system-level settings and may generate different warnings during package installation."
In some cases, specific renv options had to be adjusted to ensure successful restoration of the environment.
Another key finding was that different R versions do not always perform identically.
Minor version differences can impact package behavior, dependencies, or compiled code — which directly affects reproducibility. At the same time, it is not practical to switch R versions in RStudio for every submission. While renv helps lock package versions, it does not fully eliminate cross-OS or R-version variability. So, our key learning was this: renv is a powerful solution for package reproducibility — but true end-to-end reproducibility also requires alignment of R version and operating system considerations.
These findings helped us refine our framework by clearly documenting R version requirements and providing guidance for cross-platform execution.

## Conclusion

To conclude, our framework focuses on making R submissions reproducible, transparent, and regulatory-friendly.
First, we prioritize open-source R packages. This minimizes additional submission overhead and makes the setup process simpler and more transparent for reviewers.
All R packages used to generate analysis results are validated following GxP principles. This ensures compliance, traceability, and confidence in analytical outputs.

To enhance reliability and build reviewer trust, we provide detailed explanations of program logic, save complete R session information in the log files, and validate the generated results.
Reproducibility is not assumed — it is documented.
From a data perspective, ADaM datasets are structured to maintain clear traceability to TFLs, while minimizing customized functions that introduce unnecessary complexity.
The ADRG clearly documents R package information and provides environment setup guidance, leveraging learnings from FDA pilot projects and other R-based submissions.
We include a single consolidated file containing all required R packages in the submission package. The renv.lock file is generated after installation to stabilize and lock the project environment.
This simplifies installation for reviewers and ensures consistency.
Finally, the submission fully complies with eCTD file structure and format requirements, aligned with the latest FDA Study Data Technical Conformance Guide.

This framework demonstrates that open-source R can meet regulatory expectations — not by compromise, but by design. By combining validated R packages, controlled environments, and clear documentation, we are not just submitting code — we are submitting confidence.
Our goal is simple: when a reviewer clicks 'Run,' they see exactly what we saw.
That is the future of transparent, efficient, and trusted regulatory submissions.

## References


R packages validation process in Johnson & Johnson
- ❑ Using R in a GxP environment: https://www.youtube.com/watch?v=lWXqfuaxNL8

Johnson & Johnson's Open Source Journey in R
- ❑ https://posit.co/webinar/johnson-johnsons-open-source-journey-in-r/

R Consortium Pilot Projects
- ❑ R Consortium Pilot 1, https://github.com/RConsortium/submissions-pilot1-to-fda
- ❑ R Consortium Pilot 2, https://github.com/RConsortium/submissions-pilot2-to-fda
- ❑ R Consortium Pilot 3, https://github.com/RConsortium/submissions-pilot3-adam-to-fda
- ❑ R Consortium Pilot 4, https://github.com/RConsortium/submissions-pilot4-container
- ❑ R Consortium Pilot 5, https://github.com/RConsortium/submissions-pilot5-datasetjson


## Contact Information

Your comments and questions are valued and encouraged.

Contact the author at: Veena Terekar, Johnson & Johnson APAC
Email: vterekar@its.jnj.com
Web: https://www.linkedin.com/in/veena-terekar-1b3a08245




## Disclaimer