Paper OS17

risk.assessr: A reliable way to validate R packages

Edward Gillian, Cytel, Gorzów Wielkopolski, Poland

Hugo Bottois, Sanofi, Paris, France

Paulin Charliquart, Sanofi, Stockholm, Sweden

Andre Couturier, Sanofi, New Jersey, USA

ABSTRACT

The increasing use of open-source R packages in regulatory submissions has created a pressing need for robust validation strategies to ensure compliance with industry regulations. Regulatory agencies such as the FDA and EMA have issued guidance on computerized systems validation, emphasizing the importance of a proportionate, risk-based approach to software validation which applies to open-source as well. Notably, guidelines such as ICH E6(R2/R3) Good Clinical Practice, FDA's General Principles of Software Validation, and EMA's Guideline on Computerized Systems and Electronic Data in Clinical Trials reinforce the need for assessing software reliability to determine appropriate validation requirements for submission-related analyses.

To address this challenge, the Sanofi R validation team has developed risk.assessr, a validated tool designed to assess the risk profile of R packages used in regulatory submissions and other statistical reporting activities. When combined with a risk-based approach to validation, risk.assessr helps organizations effectively manage risks associated with internal, third-party and open-source packages. The package provides key features such as RCMD check, test coverage analysis, a traceability matrix, and other metrics used to define package's risk profile. Based on this information, appropriate controls are put in place to ensuring the accuracy, reliability, and regulatory compliance of statistical analyses using these packages. As part of its commitment to fostering transparency and collaboration, the Sanofi R validation team plans to open-source risk.assessr in Q1 2025, enabling industry-wide contributions, continuous improvement, and broader adoption of risk-based approaches to R package validation.

INTRODUCTION

Rationales for validation

To validate packages for regulatory submission, companies and other organizations need methods to ensure the reliability, validity and traceability of results through testing and documentation. Health authorities approving regulatory submissions follow a number of standards. Firstly, international standards state, "all clinical trial information should be recorded, handled and stored in a way that allows its accurate reporting, interpretation and verification" (U.S. Department of Health and Human Services, Food and Drug Administration, 2018). The Food and Drug Administration (FDA) regulatory authority reports; "The computer software used for data management and statistical analysis should be reliable, and documentation of appropriate software testing procedures should be available" (Center for Drug Evaluation and Research, Center for Biologics Evaluation and Research, 1998). The European Medicines Agency notes; "the ultimate responsibility with regards to the clinical trial conduct — in particular related to the safety of subjects and the integrity, reliability and robustness of the data generated in the clinical trial — remains with the sponsor" (European Medicines Agency, 2020). The approach to validation should be based on a risk assessment that takes into consideration the intended use of the system and the potential of the system to affect human subject protection and reliability of clinical trial results. (ICH E6 R2 1.65). In summary, these agencies are looking for accurate, reliable reporting and interpretation of clinical trial results from computer systems and the sponsors bear the responsibility for ensuring the reliability and validity of their computerized systems to produce documented accurate and traceable results.

RISK.ASSESSR PACKAGE RATIONALES

Our evaluation of guidance, validation best practices and existing R Package risk assessment tools lead us to conclude that we needed to invest in our own risk metric assessment tool which coupled with the implementation of a risk based approach to R package validation would allow us to focus on "managing risk, especially when using third-party or open-source packages where the development process is outside our control" (Gift, 2024). This involves:

- Performing Risk Assessments: Evaluate the potential risks linked to each R package by considering factors such as the package's complexity, its importance to the analysis, its documentation, its code quality, its development history, etc.
- Prioritizing Resources: Allocate resources to the high-risk or most critical components identified during the risk assessment.
- Creating Test Cases and Validation Procedures: Develop test cases and validation procedures tailored to the level of risk identified.

Development of the risk.assessr package (Gillian, Bottois, Charliquart, & Couturier, 2025), which was greatly inspired by the pharmaverse riskmetric package (R Validation Hub, 2021) started in 2024 and had its first release internally later that year. The open-source version was released to the community in February 2025 with a limited set of features and a second release is planned later this year.

RISK.ASSESSR PACKAGE FEATURES

The risk.assessr package contains a number of features to produce reliable risk metrics that are leveraged to assess the risk profile of R packages that will drive the validation plan. This section will outline and discuss those features:

The risk.assessr package offers two methods to analyze source code for the risk metric assessment process:

Analysis Method	Description	Comments
tar.gz	Saving the source code for a package as a tar file	This is a standard input method used by many risk metric packages including: pharmaverse risk metric (R Validation Hub, 2021) mpn.scorecard (Metrum Research Group, 2023) pharmapkgs (R Validation Hub, 2024)
renv.lock and pak.lock files	Saving the submission package environment and its dependencies (package names and versions) in 1 file	This method allows for an entire environment to be assessed for risk; especially considering package dependencies. At Sanofi, we have set up the running of this input method type as a github action so it can be run as a background process

Table 1: risk.assessr input methods

RCMD CHECK

The RCMD check is the largest and most comprehensive check of R packages in the R eco system. It offers more than 50 individual checks of an R package including meta-data, package structure, the DESCRIPTION file (meta-information, dependencies), NAMESPACE, R code checks, data, documentation, demonstrations, compiled code, tests and vignettes. (Hadley Wickham, 2023) The risk.assessr package ensures that RCMD checks are reliably executed on the target R package and produce accurate results as shown in Table 2:

```
results$check_list
```

```
$res_check
── R CMD check results ──────────────────────────────────────────── here 1.0.1 ────
Duration: 56.3s

❯ checking Rd cross-references ... WARNING
  Package non disponible pour vérifier les xrefs Rd : 'uuid'

❯ checking package dependencies ... NOTE
  Packages suggested but not available for checking:
    'conflicted', 'palmerpenguins', 'plyr', 'uuid'

0 errors ✓ | 1 warning ✕ | 1 note ✕

$check_score
[1] 0.65
```

Table 2: risk.assessr RCMD check results

The RCMD check in risk.assessr produces detailed information on the errors, warnings, and notes generated by this function to enable developers to fix these issues (See Table 6 in advanced reporting features). The RCMD check in risk.assessr has two options: the basic check as outlined above as well as a CRAN feasibility check which carries out an additional 46 checks.

TEST COVERAGE

Test coverage is a key validation component in assessing the accuracy of R packages. Our implementation not only provides a global package coverage estimate but also provides the test coverage of all individual exported functions as shown in Table 3:

```
results$covr_list
```

```
$total_cov
[1] 0.9867

$res_cov
$res_cov$name
[1] "here"

$res_cov$coverage
$res_cov$coverage$filecoverage
     R/aaa.R  R/dr_here.R     R/here.R     R/i_am.R R/set_here.R      R/zzz.R
      100.00       100.00       100.00        95.83       100.00       100.00

$res_cov$coverage$totalcoverage
[1] 98.67


$res_cov$errors
[1] NA

$res_cov$notes
[1] NA
```

Table 3: test coverage results

Like RCMD check, test coverage in risk.assessr produces detailed information on the errors and notes generated when running this function to enable developers to fix these. If test coverage is successfully executed, this allows a detailed review of the Package test quality and enables the next feature to be successfully carried out.

TRACEABILITY MATRIX

The traceability matrix in risk.assessr that matches the function / test descriptions to tests and match to test pass/fail, as seen in Table 4. This latter feature of matching functions to tests passing or failing is a component of "Requirement Traceability Matrix (RTM) is a document that maps and traces user requirement with test cases" (Hamilton, 2024). To our knowledge, risk.assessr is the only package currently capable of producing this detailed traceability matrix.

```
results$tm
```

```
# A tibble: 4 x 5
  exported_function code_script  documentation description                  coverage_percent
  <chr>             <chr>        <chr>         <chr>                                  <dbl>
1 dr_here           R/dr_here.R  dr_here.Rd    "dr_here() shows a message t…          100
2 here              R/here.R     here.Rd       "here() uses a reasonable he…          100
3 i_am              R/i_am.R     i_am.Rd       "Add a call to here::i_am(\"…           95.8
4 set_here          R/set_here.R set_here.Rd   "html<a href='https://www.ti…          100
```

Table 4: traceability matrix

This feature allows developers to identify functions that may be critical in the submission environment and take remedial action such as writing supplemental unit tests and/or referring the functions to subject matter experts (SMEs) to critically evaluate the output of these functions for reliability and accuracy.

ADVANCED REPORTING FEATURES

The advanced reporting features in risk.assessr includes conditional formatting and interactive elements in HTML format to allow developers to quickly identify risk metric values that may be of interest in validating the source code. An example of conditional formatting is in the Documentation metrics area, as shown in Table 5:

Documentation Metrics	
Metric Type	***Value***
Has Bug Reports URL	Included
Has License	Included
Has Examples	Included
Has Maintainer	Included
Has News	Included
Has Source Control	Included
Has Vignettes	Included
Has Website	Included
News Current	Included
Export Help	Not Included

Table 5: Documentation Metrics

As can be seen in Table 5, the Export Help metric is conditionally highlighted in red to allow developers to quickly identify the possibly problematic risk metric. Similarly, the RCMD check details are highlighted in color, as shown in Table 6:

R CMD check				
	Score	*Errors*	*Warnings*	*Notes*
RCMD Check score	0	checking running R code from vignettes ... 'here.Rmd' using 'UTF-8'... failed 'rmarkdown.Rmd' using 'UTF-8'... failed ERROR Errors in running code in vignettes: when running code in 'here.Rmd' ... [1] "/tmp/Rtmpk77j3n/file118f379c6946/vignettes" > here::i_am("prepare/penguins.R") When sourcing 'here.R': Error: Could not find associated project in working directory or any parent directory. - Path in project: prepare/penguins.R - Current working directory: /tmp/Rtmpk77j3n/file118f379c6946/vignettes Please open the project associated with this file and try again. Execution halted when running code in 'rmarkdown.Rmd' ... [1] "/tmp/RtmpiK9Ifh/file1197260f45e8/vignettes" > here::i_am("analysis/report.Rmd") When sourcing 'rmarkdown.R': Error: Could not find associated project in working directory or any parent directory. - Path in project: analysis/report.Rmd - Current working directory: /tmp/RtmpiK9Ifh/file1197260f45e8/vignettes Please open the project associated with this file and try again. Execution halted	No warnings	checking package dependencies ... NOTE Packages suggested but not available for checking: 'conflicted', 'palmerpenguins'

Table 6: RCMD check details

Again, the conditional formatting in Table 6 allows developers to quickly find risk metric areas that need remediation in the validation process.

The interactive elements such as filtering and sorting mean that developers can quickly identify functions in the traceability matrix that may need additional test coverage, as shown in Table 7.

Traceability Matrix

Search:

	Exported Function	Code Script	Documentation	Description	Test Coverage %
248	starts_with		reexports.Rd	NA	
263	sym		tidyeval-compat.Rd	NA	
265	syms		tidyeval-compat.Rd	NA	
273	tibble		reexports.Rd	NA	
281	tribble		reexports.Rd	NA	
282	type_sum		reexports.Rd	NA	
284	union		setops.Rd	NA	
289	where		reexports.Rd	NA	
38	common_by	R/join-common-by.R	common_by.Rd	Extract out common by variables	0
102	explain	R/explain.R	explain.Rd	This is a generic function which gives more details about an object than print(), and is more focused on human readable output than str().	0

Showing 31 to 40 of 292 entries Previous 1 2 3 4 5 … 30 Next

Table 7: traceability matrix sorting feature

The traceability matrix for dplyr version 1.1.4 shows that there are 292 exported functions in this package. By clicking the sorting arrows for Test Coverage %, the user can get this information in descending order and quickly find functions like common_by that have no or low test coverage and that may need remediation.

CHECK PACKAGE STRUCTURE FOR IMPORTS/SUGGESTS

This functionality is only available in risk.assessr and checks the exported functions of “target package” against the exported functions of “Suggested dependencies” to see if any Suggested packages should be in Imports in the DESCRIPTION file. An example of this is from teal.code version 0.5.0 (Insights Engineering, 2024), where the dependency cli was in Suggests and not Imports in the DESCRIPTION file, as shown in Table 8: Note that this issue was later discovered by the Teal team and recently corrected.

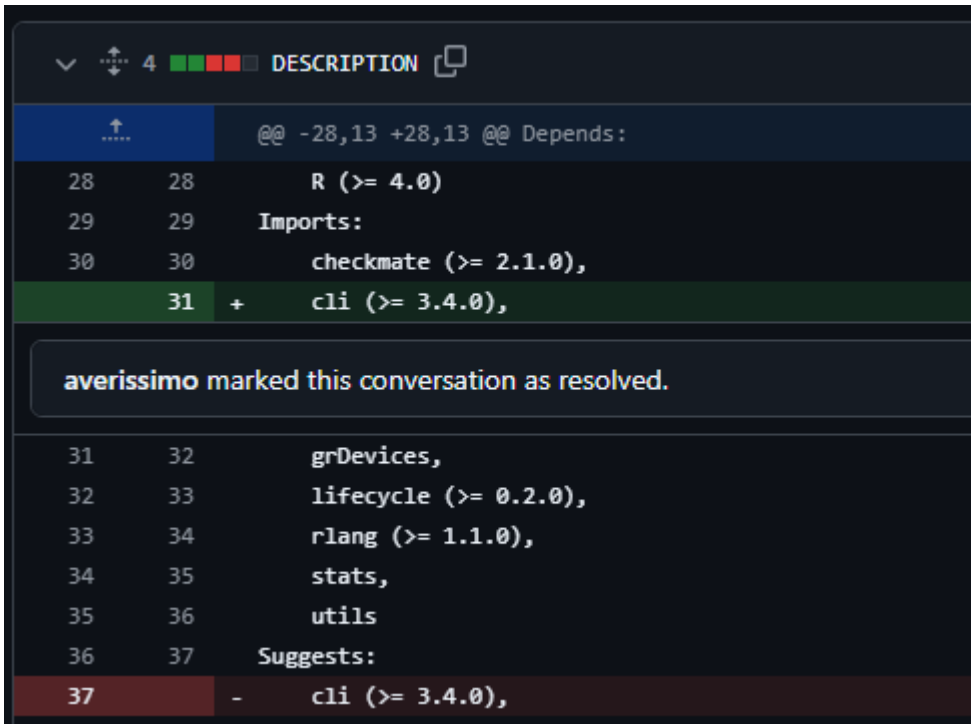

Table 8: teal.code DESCRIPTION

The code in risk.assessr correctly identified the S3 function that contained code from cli in the function body and recommended action be taken in regard to dependency usage, as seen in Table 9:

```
> assessment_results$results$suggested_deps
# A tibble: 1 × 4
  source    suggested_function targeted_package message
  <chr>     <chr>              <chr>            <chr>
1 eval_code ansi_strip         cli              Please check if the targeted package should be in Imports
```

Table 9: risk.assessr dependency check

The example in Table 9 identifies the source function in the target package (source), the function in the source body (suggested function ansi_strip), the package (cli) that the function comes from, and message about what remediation can be done. This feature works with S3, S4, R6 and standard functions in R packages.

CHECK PACKAGE ACTIVITY, MAINTENANCE AND COMMUNITY ENGAGEMENT MEASURES

The risk.assessr package checks for activity and community engagement measures such as reverse dependencies stars, forks, open issues, and recent commits to provide a gauge of how popular the package is Table 10. The package can calculate the number of packages that refer to the target package searching for the terms "Depends", "Imports", "Suggests", "LinkingTo". It then generates a reverse dependency score, the total number of reverse dependencies, and an interactive table with search and sorting capabilities, as shown in Table 11:

```
results$github_data
```

```
$created_at
[1] "2012-10-28T13:39:17Z"

$stars
[1] 4821

$forks
[1] 2122

$date
[1] "2025-02-04"

$recent_commits_count
[1] 0
```

```
results$download
```

```
$total_download
[1] 119300000

$last_month_download
[1] 1400000
```

Table 10: risk.assessr GitHub related data including package creation date, number of stars, fork, data acquisition date and recent commits count (last 30 days) and CRAN download (total and last month downloads) for dplyr package.

Reverse Dependencies	
Reverse Dependency Score	*Number of Reverse Dependencies*
1	4855

Show 10 entries Search:

1	AATtools
2	abjutils
3	abstr
4	academictwitteR
5	AccelStab
6	accept
7	accessr
8	accrualPlot
9	accucor
10	ACDm

Showing 1 to 10 of 4,855 entries Previous 1 2 3 4 5 … 486 Next

Table 11: risk.assessr reverse dependencies

The risk.assessr package also checks for the number of available versions for this package on CRAN and checks if the package is stored on CRAN, GitHub, Bioconductor and in the internal Sanofi mirror and provide links as can be seen in Table 12:.

```
results$available_version
```

```
 [1] "0.1.1"   "0.1.2"   "0.1.3"   "0.1"     "0.2"     "0.3.0.1" "0.3.0.2" "0.3"     "0.4.0"
[10] "0.4.1"   "0.4.2"   "0.4.3"   "0.5.0"   "0.7.0"   "0.7.1"   "0.7.2"   "0.7.3"   "0.7.4"
[19] "0.7.5"   "0.7.6"   "0.7.7"   "0.7.8"   "0.8.0.1" "0.8.0"   "0.8.1"   "0.8.2"   "0.8.3"
[28] "0.8.4"   "0.8.5"   "1.0.0"   "1.0.1"   "1.0.2"   "1.0.3"   "1.0.4"   "1.0.5"   "1.0.6"
[37] "1.0.7"   "1.0.8"   "1.0.9"   "1.0.10"  "1.1.0"   "1.1.1"   "1.1.2"   "1.1.3"   "1.1.4"
```

```
results$host
```

```
$github_links
[1] "https://github.com/tidyverse/dplyr"

$cran_links
[1] "https://cran.r-project.org/src/contrib/dplyr_1.1.4.tar.gz"

$internal_links
NULL

$bioconductor_links
NULL
```

Table 12: List of available versions and the repository host links from GitHub, Cran and Bioconductor for dplyr package

DISCUSSION

We are still in the learning phase of our R adoption journey for submission. We have made great progress in a short amount of time thanks to the experience shared by other Pharma, the different PhUSE and R related working groups, conferences attendance and webinars and our great people. We still have some way to go on the technical side and people side, but will mainly focus on the technical side and discuss our current internal implementation of the risk.assessr to support the creation of our R submission environment and mention a few next steps we target to resolve in 2025.

Internally the risk.assessr package runs in a validated production Docker environment to ensure the reliability, reproducibility, and integrity of each metrics used to definine the risk profile of all our submission packages. By using Docker, several advantages will be gained. Docker containers enable the encapsulation of target R packages and its dependencies, ensuring that the risk.assessr package runs consistently across different environments.

The use of Docker containers follows the recommended best practice to manage a reproducible R environment (The R Foundation for Statistical Computing, 2021). It allows us to control the architecture (e.g. operating system, R version, package version, dependency versions), controlled updates of packages (maintenance), automated testing, documentation of the Docker setup (e.g. the Dockerfile, environment variables, and any custom scripts), and implement security measures (e.g. scan Docker images for vulnerabilities and apply security patches as needed).

We implemented a CI/CD pipeline to automatically run risk.assessr on any package, storing environment specific results in a database. To evaluate package quality, we defined custom policy rules based on key metrics such as test coverage, successful R CMD checks, and dependency count, leveraging the risk.assessr data. Quantitative package risk is automatically assessed and categorized into 3 levels of risk (low, medium or high) based on the risk.assessr metrics and stored in the database with corresponding metrics for tracking temporal changes in risk and further analysis. We are also looking at how to integrate qualitative risks into the risk assessment process.

An R environment governance body will be put in place later this year to manage the evolution of our different R environments through its life cycle.

CONCLUSION

The risk.assessr package is a reliable and validated tool that will be used by the Sanofi R Validation team to check R packages risk profile in the submission environment. Combined with a risk-based approach, this tool plays a key role to ensure packages used for submission provide accurate and reliable results that meet regulatory submission requirements. The package has features such as RCMD check and test coverage that produce reliable and valid results. This package has new features such as a function specific test coverage, traceability matrix that reports test results, checks packages for imported and suggested dependencies. The package also has advanced reporting features, different input methods, and functionality to gather data from GitHub repositories in the areas of project activity and community engagement.

The Sanofi Validation team has released the first open-source version of this package in February 2025 to enable sharing of resources to aid in the creation of common risk metric standards for creating faster deliverables for future submissions.

CONTACT INFORMATION

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

Author Name: Edward Gillian
Company: Cytel
Address: 675 Massachusetts Ave Cambridge, MA 02139 USA
Work Phone: +1 (617) 661-2011
Email: edward.gilllian-ext@sanofi.com
Website: www.cytel.com

References

Center for Drug Evaluation and Research, Center for Biologics Evaluation and Research. (1998, September). *E9 Statistical Principles for Clinical Trials.* Retrieved from https://www.fda.gov/regulatory-information/search-fda-guidance-documents/e9-statistical-principles-clinical-trials

CRAN Team. (2004, January 21). *The Comprehensive R Archive Network.* Retrieved from https://cran.r-project.org/

European Medicines Agency. (2020, April 7). *Q&A: Good clinical practice (GCP).* Retrieved from https://www.ema.europa.eu/en/documents/regulatory-procedural-guideline/notice-sponsors-validation-and-qualification-computerised-systems-used-clinical-trials_en.pdf

Genetech. (2024, July 10). *covtracer*. Retrieved from https://github.com/Genentech/covtracer

Gift, K. (2024, October 16). *A Guide to R Package Validation in Pharma.* Retrieved from https://r-craft.org/a-guide-to-r-package-validation-in-pharma/

Gillian, E., Bottois, H., Charliquart, P., & Couturier, A. (2025, February 6). *risk.assessr*. Retrieved from risk.assessr: https://github.com/Sanofi-Public/risk.assessr

Hadley Wickham, J. B. (2023, July 25). *R Packages: Organize, Test, Document, and Share Your Code 2nd Edition.* Retrieved from https://r-pkgs.org/r-cmd-check.html

Hamilton, T. (2024, December 31). *What is Requirements Traceability Matrix (RTM) in Testing?* Retrieved from https://www.guru99.com/traceability-matrix.html

Insights Engineering. (2024, January 11). *teal.code*. Retrieved from https://github.com/insightsengineering/teal.code

Metrum Research Group. (2023, August 25). *mpn.scorecard.* Retrieved from https://github.com/metrumresearchgroup/mpn.scorecard

R Core Team. (2024, October 31). *R Internals.* Retrieved from https://cran.r-project.org/doc/manuals/r-release/R-ints.pdf

R Validation Hub. (2021, August 6). *riskmetric.* Retrieved from https://github.com/pharmaR/riskmetric

R Validation Hub. (2024, May 16). *pharmapkgs.* Retrieved from https://github.com/pharmaR/pharmapkgs

Sanofi. (2025, February 6). *risk.assessr*. Retrieved from risk.assessr: https://github.com/Sanofi-Public/risk.assessr

The R Foundation for Statistical Computing. (2021, October 18). Retrieved from 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

U.S. Department of Health and Human Services, Food and Drug Administration, . (2018, March). *E6(R2) Good Clinical Practice: Integrated Addendum to ICH E6(R1) Guidance to Industry.* Retrieved from

https://www.fda.gov/media/93884/download

ACKNOWLEDGMENTS

Edward Gillian and Hugo Bottois participated in providing and implementing the technical solutions. Andre Couturier handled project management and guided the project direction. Paulin Charliquart conducted the project and documentation review.

The project is inspired by the riskmetric package and the mpn.scorecard package and draws on some of their ideas and functions.