R Package Development at Novo Nordisk (Biostatistics)

Anders Ellern Bilgrau
abiu@novonordisk.com // 2022-11-14

Views and opinions expressed are those of the speaker and not necessarily Novo Nordisk
R extensions

• **R** is a language and environment for statistical computing and graphics

• R is easily **extendible** by versioned bundles of code and documentation called **packages**

• Each **package can depend** on multiple **other** packages and have system dependencies

• **CRAN** is a network that stores and serves **R-packages**

• **Current packages** on CRAN are checked and tested to ensure **compatibility** and a minimum standard
Packages

R packages
- The organisational unit of extending R and bundling functionality
- The package source is a collection of .R files, datasets, documentation, needed libraries and/or other source code (C, C++)
- The package source can be bundled (into a tar.gz) and/or built (into a binary) for distribution
- Can easily be tested and checked for standards
- Can easily be shared with others
  - CRAN for public, global sharing

SAS macros
- Wraps functionality into function-like behavior
- Does not extend well to complex functionality
- No common way of testing and documenting

Python packages/modules
- One organisational unit of extending Python and bundling functionality
- Can easily be shared with others
  - PyPI for public, global sharing

PyPI: Python Package Index; CRAN: Comprehensive R Archive Network
NN SCE-R infrastructure

NN: Novo Nordisk
SCE: Scientific Computing Environment
Package strategy in a validated environment

- Create an environment where less experienced users can easily share and re-run work, but restrict immediate access to a particular set of packages.

Teams specify the version of R used for a specific trial.

Teams can choose to use specific versions of packages, but this should always be in conjunction with renv.

People can installed packages into their own user library.
Which packages can/should we use

- **Base/Recommended**: (Can be trusted)
  - The R Foundation develops both the base and recommended packages, and follows practices that ensures the accuracy of each

- **Contributed**: (Need internal testing)
  - 15000+ R packages on CRAN, all tested to some extent, but not all can be considered validated

- **Popular**: (Very low risk)
  - A subset of the contributed packages have an extremely large userbase and extensive test-suites
  - tidyverse, data.table, ...

Low risk packages

- Base
- Recommended
- Contributed
- Popular
- Well tested and documented
- The rest
Risk assessment strategy – R Validation Hub

New R package

What is the package classification

What is the package purpose

Is the packaged maintained

Is the package widely used

Minimum checks for suitability

Remediation / testing

Remediation / testing

Remediation / testing

“Intended for use”

Stat

Non-stat

Yes

Yes

Yes

Include package

Meet requirements

Remediation / testing

Remediation / testing

Remediation / testing

Remediation / testing

Risk assessment by package administrators

NN package development philosophy
Tools-on-top development

Packages

- Packages builds on top of each other (from low to high level)
- Lego-bricks principle
- Build up more complex functionality toward automation
- Output/app/content programming *all* uses the same packages
- Ability to make APIs to package functions with the connect server
Tools-on-top development

Reused, modular, and replacable code

- Pkg A
- Pkg B
- Pkg 1
- Pkg 2
- Pkg X
- Pkg Y
- Pkg Z

Abstraction level

- High-level (Task dependent)
- Low-level (Architecture dependent)

Stakeholders

Programmers

App / Content

SCE-R
Tools-on-top development
Reused, modular, and replacable code

- Pkg A
- NNaccess
- Pkg X
- NNremote
- NNexport
- Reactive TFL

Abstraction level:
- High-level (Task dependent)
- Low-level (Architecture dependent)

Programmers
Stakeholders

SCE-R
Tools-on-top development
Reused, modular, and replacable code

SAS compute node

Pre-SDTM → SDTM → ADaM → Output → Reports

Trials

Stakeholders

Programmers

Reactive TFL

NNtfl

Pkg 1

Pkg A

Pkg X

NNaccess

NNremote

NNexport

High-level (Task dependent)

Low-level (Architecture dependent)
Tools-on-top development
Reused, modular, and replacable code

High-level (Task dependent)

Low-level (Architecture dependent)

Programmers

Reactive TFL

Stakeholders

SAS compute node

Pre-SDTM

SDTM

ADaM

Output

Reports

SDTM

ADaM

Output

Reports

Trials

Insights

Pkg 1

Pkg A

Pkg X

NNaccess

NNremote

NNexport

PKg 1

NNaccess

Pkg X

NNremote

NNexport

SCE-R
R Package development in NN
Internal packages

- Started with the package "NNR" developed from local laptops (no SCE supporting R)
- A "NNBiostat" package defined to be "accepted packages"
- NNR redefined to umbrella of logically related packages
Internal packages producing TFL

- Supplies easy access to read and write data
- Easy to direct outputs to wanted folders
- Plots with titles, footnotes, and NN colours
- Plots with colours and symbols as defined by MDPARAM
- Tables similar to current layout from SAS
- Designed in a flexible format so that we can add stuff to it
- Export tables and figures in a format that can be used in our current TFL pipeline
- Generate associated xml files automatically
- Run R and Rmd scripts in batch mode
- Produce nicely formatted html and markdown logs – integrated with Azure DevOps
Internal packages – Network

- The internally developed packages are dependent on each other
- No cyclic dependencies

- Each package is hosted version controled on Azure DevOps and has:
  - Version number
  - Maintainer group
  - Repository
  - Pipeline
  - Board for bugs, questions, feature requests
Internal package / tool chain

- .git
- azure-pipelines.yml
- DESCRIPTION
- docs
  - *.html
- inst
  - extdata
- man
  - *.Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
- _pkgdown.yml
- R
  - *.R
- README.md
- tests
  - testthat
    - *.R
    - testthat.R
- vignettes
  - *.Rmd
Internal package / tool chain

directory structure:

- .git
- azure-pipelines.yml
- DESCRIPTION
- docs
  - *.html
- inst
  - extdata
- man
  - *.Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
  - _pkgdown.yml
- R
  - *.R
- README.md
- tests
  - testthat
    - *.R
    - testthat.R
- vignettes
  - *.Rmd

git version control
Internal package / tool chain

```
.git azure-pipelines.yml  DESCRIPTION  docs
     __ * . html
   inst  __ extdata
   man
     __ * . Rd
  NAMESPACE
  NEWS . md
  pkgdown
     __ ...
  _pkgdown . yml
  R
     __ * . R
  README . md
  tests
  __ testthat
    __ * . R
    __ testthat . R
  vignettes
     __ * . Rmd
```

git version control

Azure DevOps
Internal package / tool chain

```
├── .git
├── azure-pipelines.yml
├── DESCRIPTION
├── docs
│   └── *.html
├── inst
│   └── extdata
├── man
│   └── *.Rd
├── NAMESPACE
├── NEWS.md
├── pkgdown
│   └── ...
│       _pkgdown.yml
├── R
│   └── *.R
├── README.md
├── tests
│   └── testthat
│       └── testthat.R
└── vignettes
    └── *.Rmd
```
Internal package / tool chain

- .git
- azure-pipelines.yml
- DESCRIPTION
- docs
  - *.html
- inst
  - extdata
- man
  - *.Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
  - _pkgdown.yml
- R
  - *.R
- README.md
- tests
  - testthat
    - *.R
    - testthat.R
- vignettes
  - *.Rmd
Internal package / tool chain

- .git
- azure-pipelines.yml
- DESCRIPTION
- docs
  - * .html
- inst
  - extdata
- man
  - * .Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
  - pkgdown.yml
- R
  - * .R
- README.md
- tests
  - testthat
    - * .R
    - testthat.R
- vignettes
  - * .Rmd

git version control

Azure DevOps

Azure Pipelines
Internal package / tool chain

```
|-- .git
  |-- azure-pipelines.yml
  |-- DESCRIPTION
  |-- docs
  |   |-- *.html
  |-- inst
  |   |-- extdata
  |-- man
  |   |-- *.Rd
  |-- NAMESPACE
  |-- NEWS.md
  |-- pkgdown
  |   |-- ...
  |   |-- pkgdown.yml
  |-- R
  |   |-- *.R
  |-- README.md
  |-- tests
  |   |-- testthat
  |   |   |-- *.R
  |   |   |-- testthat.R
  |-- vignettes
  |   |-- *.Rmd
```

- **git version control**
- **Azure DevOps**
- **Azure Pipelines**
Internal package / tool chain

- .git
- azure-pipelines.yml
- DESCRIPTION
- docs
  - *.html
- inst
  - extdata
- man
  - *.Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
  - _pkgdown.yml
- R
  - *
- README.md
- tests
  - testthat
    - *
    - testthat.R
  - vignettes
    - *

Azure DevOps

git version control

Azure Pipelines
Internal package / tool chain

- .git
  - azure-pipelines.yml
  - DESCRIPTION
- docs
  - *.html
- inst
  - extdata
- man
  - *.Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
  - _pkgdown.yml
- R
  - *.R
- README.md
- tests
  - testthat
    - *
  - testthat.R
- vignettes
  - *.Rmd
Internal package / tool chain

- .git
- azure-pipelines.yml
- DESCRIPTION
- docs
  - *.html
- inst
  - extdata
- man
  - *.Rd
- NAMESPACE
- NEWS.md
- pkgdown
  - ...
  - _pkgdown.yml
- R
  - *.R
- README.md
- tests
  - testthat
    - *.R
  - testthat.R
- vignettes
  - *.Rmd
Continuous integration and deployment

On each code-change/commit, the package pipeline is triggered and runs:

- **Test:**
  - Matrix checks and tests in **all** released versions of R

- **Build:**
  - Builds the package “tarball” (package_X.Y.Z.tar.gz)

- **Release:**
  - to the **production** package manager (on master and new version X.Y.Z)
  - to a **test** package manager (if dev-version “X.Y.Z.D”)
  - the new package homepage
Continuous integration and deployment

• A git master-branch policy is enforced:
  • Only via pull-requests
  • A person in the maintainer group should approve the code changes
  • Pipeline runs with success
  • A feature/bug is linked

Ensures validation of internal packages
Package and function lifecycle

- Functions are marked using the lifecycle package to communicate to users the risk level of functions within a package

- **Future:**
  - Stable functions must be covered thoroughly by tests
  - Demands on experimental functions are relaxed
User support & education
User support & education

Support
• One-entry point documentation
• Establishing a dedicated support team
• Establishing a StackOverflow environment

Education
1. Git is a prerequisite
   • All trial/exploratory tasks that use R should use git
2. Conduct courses in how to get started with R in NN
   • Will include some training in git
3. R self-learning and learning-by-doing
   • Material for self-learning is provided via degreed/r-doc
Everything needed to use R within Biostatistics should be available at r-doc.

Go to the homepage by typing r-doc in the browser.
Vignettes

• Directly available from training platform (and from within R)
• Re-run when package is built (to ensure it runs)
• Anybody can contribute
• Written in R-markdown
• Gradually being made
2 episodes released per week
10-15 minutes ‘how-to’
All code examples available in vignettes
DevOps system for getting feedback
Anybody can contribute with episodes
Help at your fingertips (within R)

```r
> vignette("nnaccess", package="nnaccess")
> help("nnaccess")
```

**Easy access to NN data**

**Description**

The `nnaccess`-function sets up access to the SAS (and other data formats) data models and datasets within the specified trial instance. See `DataAccessR` for more on the usage of the output of `nnaccess`.

**Usage**

```r
is.nnaccess(x)

nnaccess(
  trial,
  instance = default_instance(trial, libref = libref),
  project = get_project(trial, libref = libref),
  libref = libref(root),
  root = getroot(),
  use_cache = TRUE,
  build_export = TRUE
)
```

```r
# S3 method for class 'nnaccess'
print(x, ...)

# S3 method for class 'nnaccess'
summary(object, ...) 
```

**Arguments**

- `x`, `object` - Annaccess object.
- `trial` - A character of length 1 specifying the trial ID (e.g. "XX"). If omitted, the trial, project, and instance is attempted derived from the `workspace` directory.
- `instance` - A character of length 1 specifying the instance.
- `project` - A character of length 1 the project ID (e.g. mYYYY).
- `libref` - An object as returned by `libref()`. A tibble or data frame giving the reference for searching for trials.
StackOverflow

- We have no good place for Q&A
  - **Tried:** MS SharePoint, MS Teams, Azure DevOps, Service Now, Wiki
- Avoid answering the same question 100 times
- Important as it alleviates support burden
  - R support team should check questions here
- StackOverflow is familiar for programmers
- Not just for code