

Bilingual execution

- Bridging SAS and R in clinical trials
- Initial thoughts

Disclaimer:

The views and opinions expressed in this presentation are those of the authors and do not necessarily reflect the official policy of Novo Nordisk A/S

Steffen Falgreen Larsen

Senior Statistician, Msc, PhD
sffl@novonordisk.com

Ari Siggaard Knoph

International Lead Programmer
Statistical Programming Specialist
aikp@novonordisk.com

9-13 November 2020

PHUSE EU Connect 2020

- # Agenda
1. Introduction and motivation
 2. The **R** environment
 3. Internal packages
 4. Initial usage
 5. Conclusion

Introduction



SAS is the golden standard at Novo Nordisk

Huge knowledge base

Pipeline build with offset in SAS



Reasons for looking into R

Experience of new employees

Extremely **flexible** language

Ability to **bundle code** in a versioned **package**:

- Documentation, standard code checks, Unit test, Code coverage, and Sharing examples



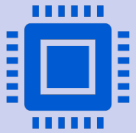
The right tool for the job

We do not wish to use **R** exclusively

Several tools used at Novo Nordisk is based on SAS and the outputs generated by SAS

Don't recode what we already have

Requirements to the **R** environment



We want to produce **output** similar to those produced in **SAS**

Stakeholders should not be affected by language choice

Mimic **existing structure** to facilitate **co-existing languages**



Hassle free system for the end user

Easy to **get started** with programming – also for **R** novices

Easily **reproduce results** later by executing scripts in the same version of **R**



We want an environment where it is easy to initiate **collaborative** work

Initiate version control of code used in trials/exploratory tasks

Initiate version control for internal **R-packages** together with a build pipeline

Implemented R environment

WEB IDE



- Rstudio server Pro
- Web interface – no setup for the user
- Server based solution ensuring common ground

Version control



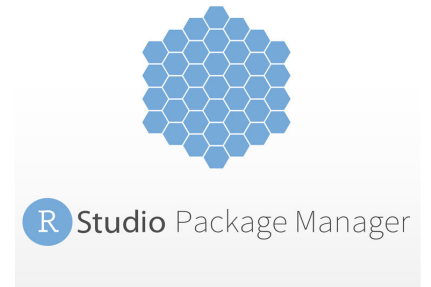
- Use version control
- Improved collaboration
- Feature branching for development

Project management



- Organizing tasks
- Shared code repository
- CI/CD
 - Build pipeline
 - Release pipeline

Package distribution



- Manage the aspects of R extensions - packages
- Share internal packages
- Create subsets of approved packages

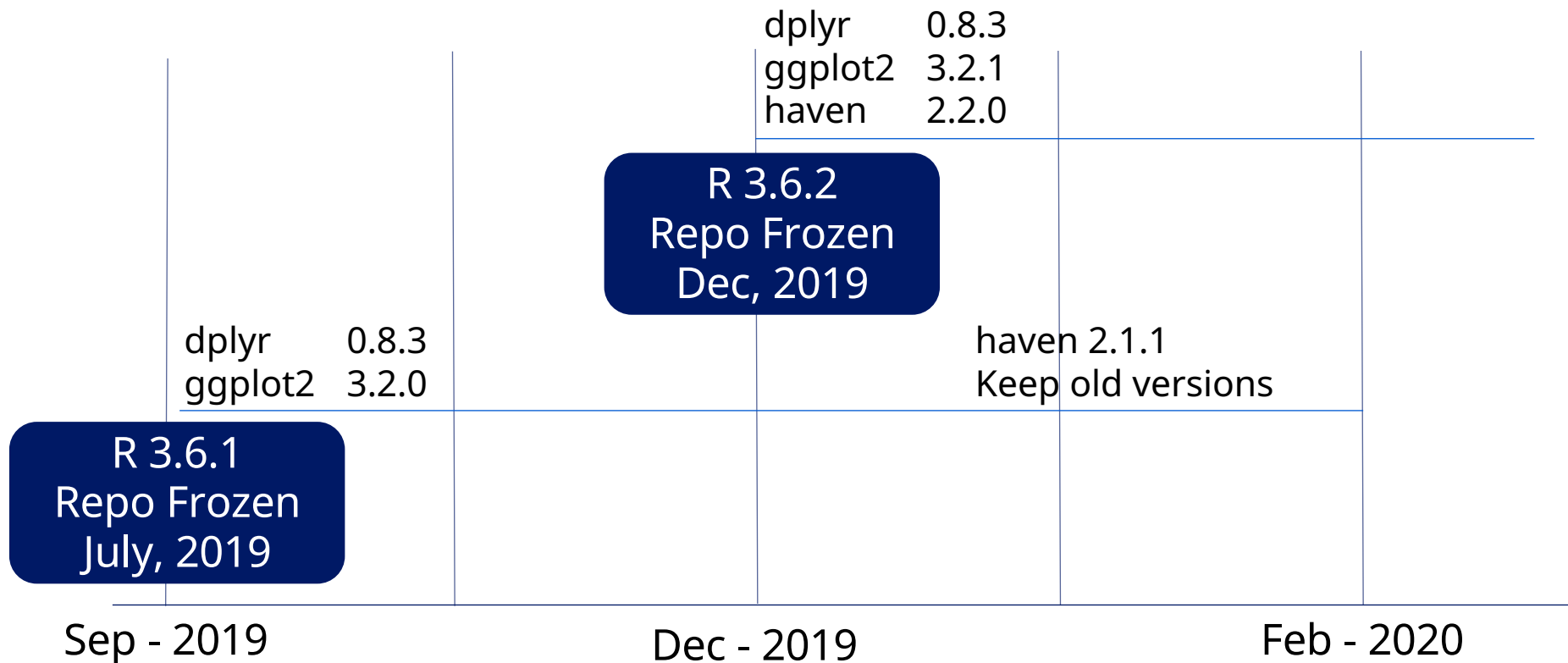
R extensions



- **R** is a language and environment for statistical computing and graphics
- The language is easily **extendible** by versioned bundles of code with documentation – **packages**
- Each **package** can **depend** on multiple **other** packages and have system dependencies
- **CRAN** is a network that stores **R**-packages and
- **Current packages** on CRAN are tested to ensure **compatibility**

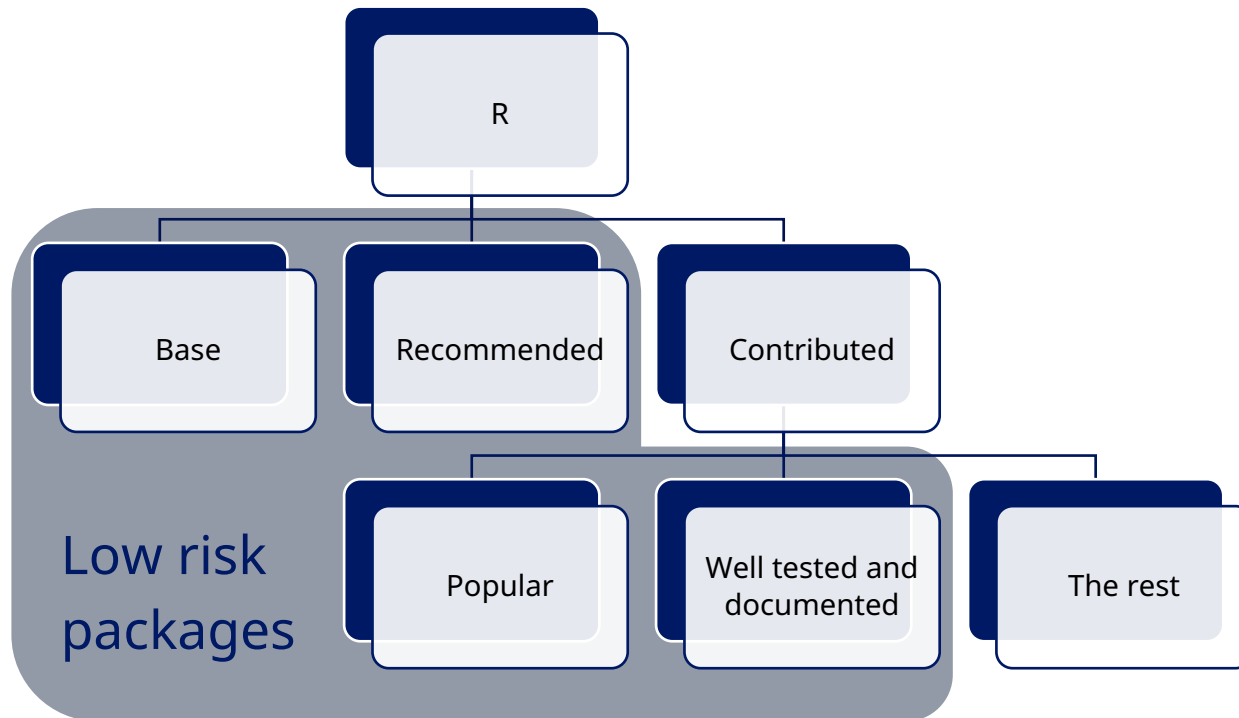
Package strategy – validated environment

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



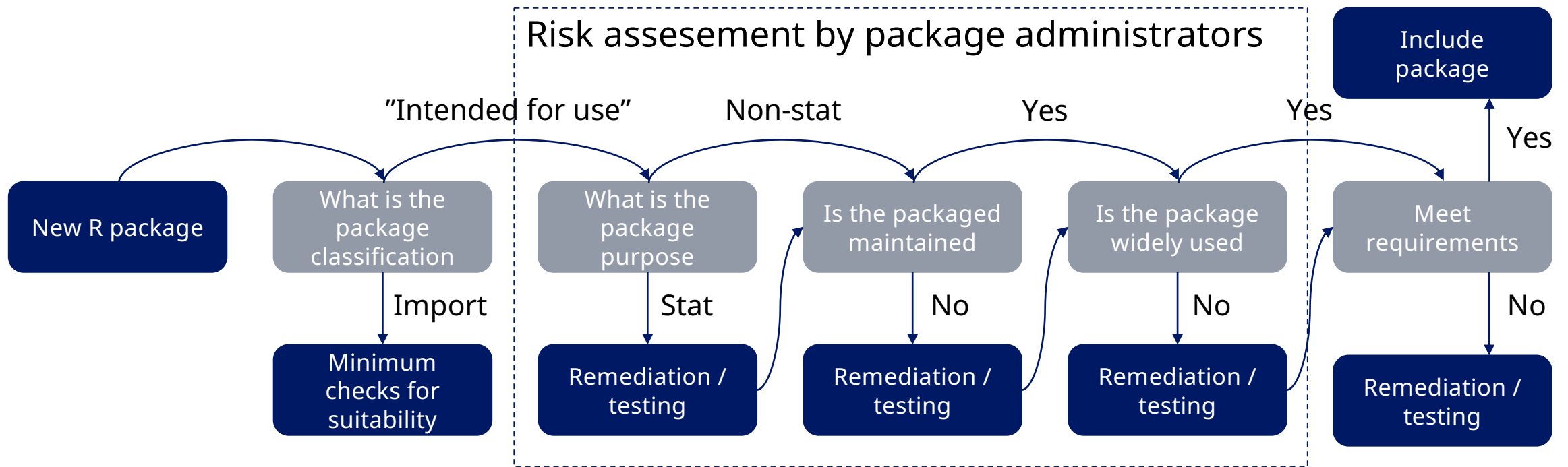
- Teams specify the version of **R** used for a specific trial
- The version of packages are not changed before they opt in to a new version

Which packages 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, ...

Risk assessment strategy – R Validation Hub



Internal packages

The internal R-package NNR

- The **NNR** package handles tasks necessary for creating an output:



- Get trial data into R
 - The function **nnaccess** is used to generate access to data repositories



- Create tables using the **NNTable** function suite that can **coexist** with current output



- Export both figures and tables with titles and footnotes and create xml file using **exportOutput**



- Create **logs** for each program with automatic session and execution info

Internal package pipeline

- Package development is versioned using git and a CI/CD pipeline is setup to make sure new updates to code is fast available to the end user

Development and test of R packages

- Documentation is one of the most important aspects of good code
- Document the code whilst writing it
- Not as steep learning curve

Documentation



- Build package
- Run R cmd checks
 - Numerous checks that ensure the integrity of the code

Helper tools



- Each package is tested by unit tests
- Easy to test if changes to code cause problems

Unit testing



Approval

- Test coverage of the unit test
- Get an easy overview of which lines in your code have and have not been tested

Test coverage



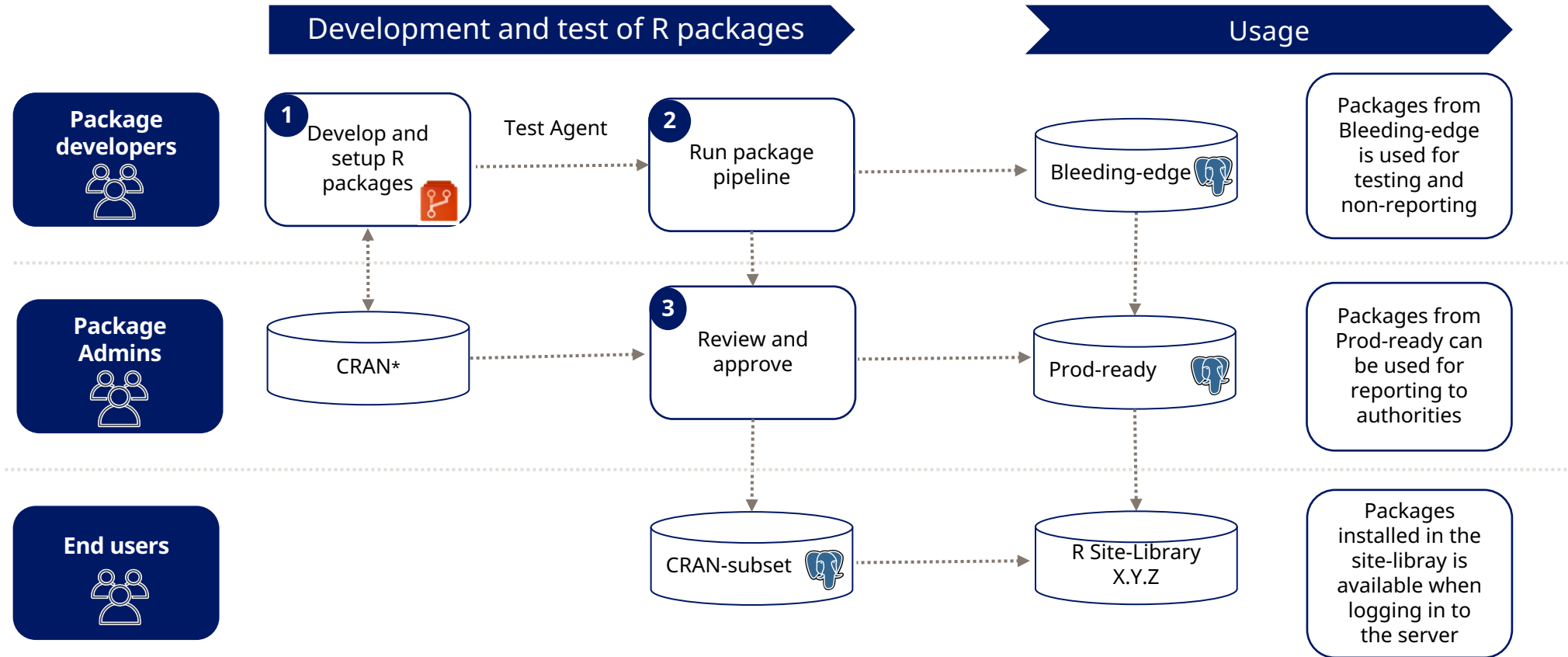
End usage

- Overview of all internal packages
- Easy access to documentation of internal packages

Build/update
Homepage



Management of packages



* CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R

Initial usage

Initial usage of **R** and future

Current usage of **R**

- Non-authority reporting activities
- Building applications
 - Supporting PK/PD review
 - Supporting data investigation

Testing the setup on a trial

- ADaM in SAS and TFL in **R**
- Parallel-programming of TFL in SAS
- Gather insights on problems

Further plans

- Validated outputs used as unit-tests
- Non-exclusive language selection

Conclusion

Setup an **R** environment – server solution

- Ensure that everybody can execute scripts and obtain the same result
- Package strategy, consider how to ensure package consistency
- Validation, the system that runs **R** must be documented
- Many packages have system dependencies

The **R** environment at department of Biostatistics

- Enables the user to simply log in and start working
- Can coexist with SAS
- Reuses the already existing tools for generating TFL
- Internal package pipeline for easy collaboration and avoid silo development

Current usage

- Non-reporting activities
- Currently being tested for use in trial reporting