

Agile programming: Towards a more collaborative programming setup

Frederik Vandvig Heinen (Senior Statistical Programmer) &
Bo Sølgaard Andersen (Statistical Programming Specialist,
International Lead Programmer)

Novo Nordisk A/S

16-11-2021

Disclaimer

“Views and opinions expressed are those of the speaker and not necessarily Novo Nordisk.”

Outline

In focus: The programmer / reviewer role (then developing and reviewing)

- What we want to do
- A classic programmer/reviewer setup
- Challenges with the current setup
- Implementation of possible solution
- Learnings

What we wanted to do

A more collaborative programmer/reviewer setup:

- To keep track of changes to a program at any given timepoint
 - To allow for more than one programmer to contribute to a program
- To support the programmer/reviewer collaboration
 - Programmer and reviewer have access to the same information (a shared communication platform)



A classic programmer/reviewer setup

Programmer (responsible/owner of the code)

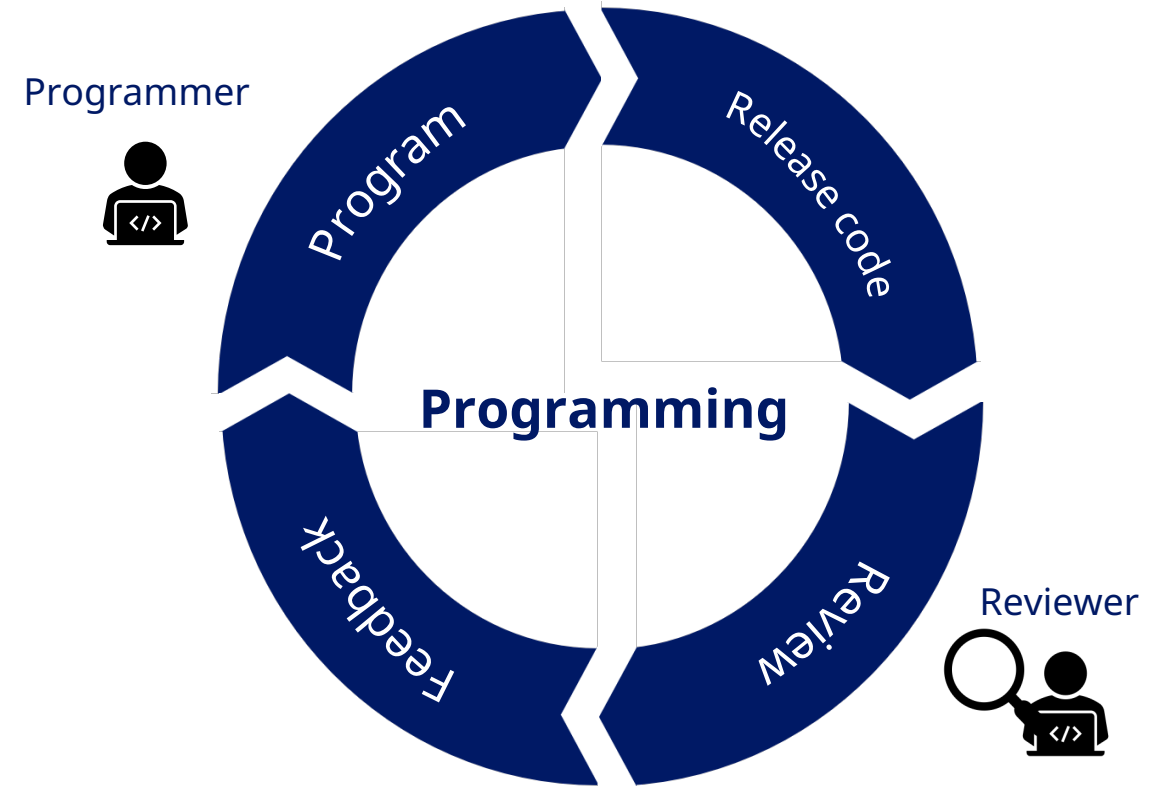
- Write the program and specifications

Reviewer

- Reviews the code and provides feedback
- Do not modify the original code

Communication-line

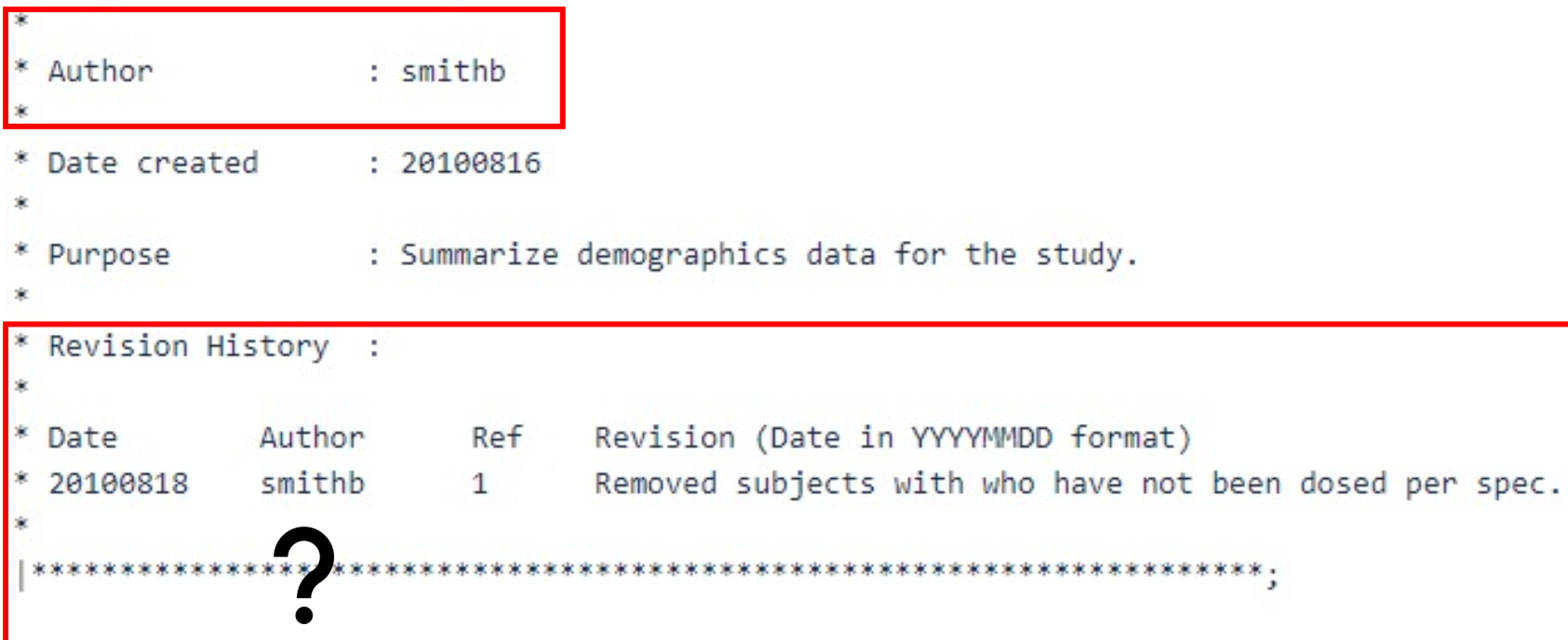
- Mail, team chat, issue log etc.



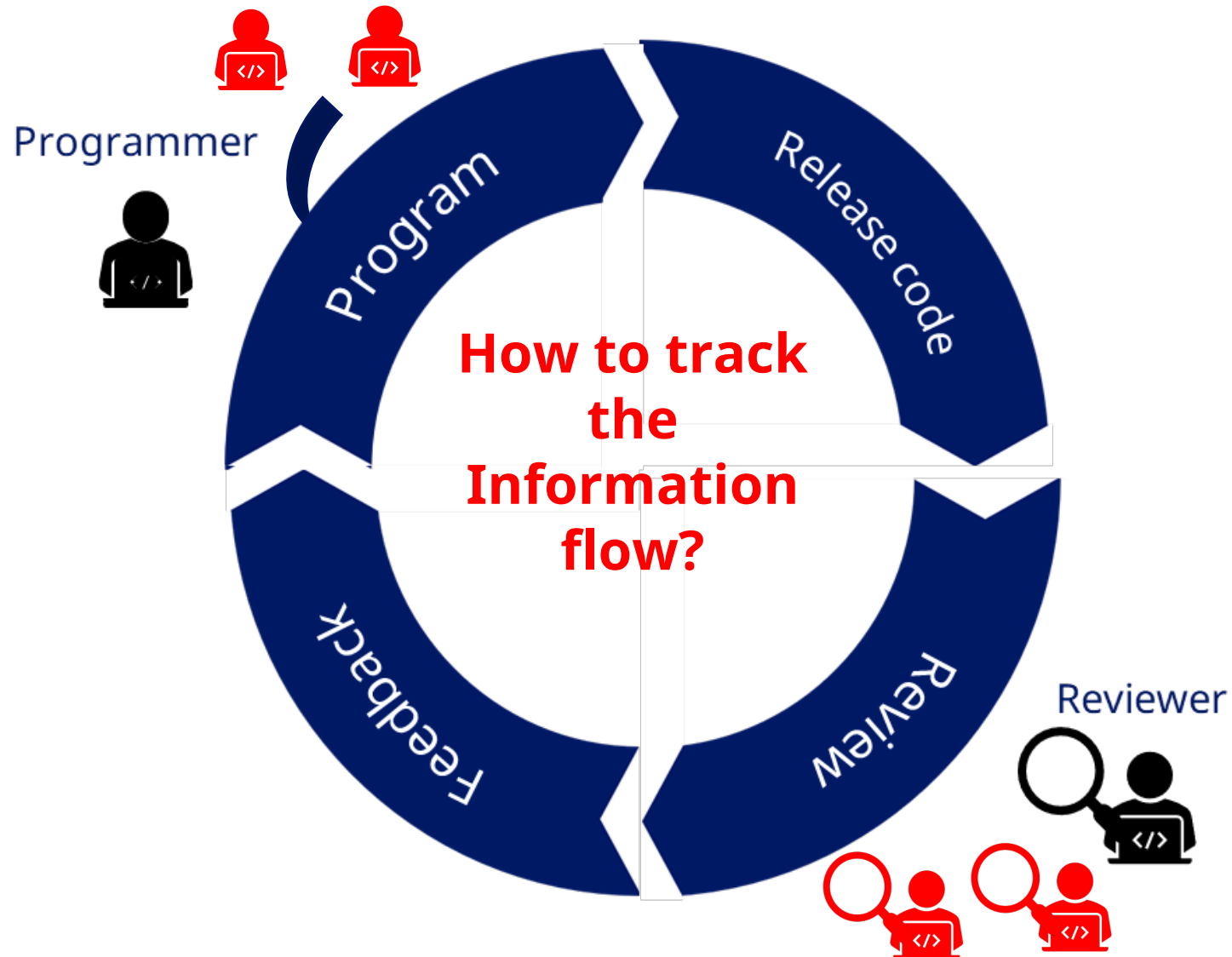
A classic programmer/reviewer setup

PHUSE GPP - Headers

```
|*****;  
* Project          : Sample Drug, Sample Indication, Study1  
*  
* Program name     : sample.sas  
*  
* Author           : smithb  
*  
* Date created      : 20100816  
*  
* Purpose           : Summarize demographics data for the study.  
*  
* Revision History :  
*  
* Date      Author      Ref      Revision (Date in YYYYMMDD format)  
* 20100818  smithb      1        Removed subjects with who have not been dosed per spec.  
*  
|*****;
```



Working in a team



Agile mindset in programming

- Version control softwares
- All aiming at providing a platform for version control of files
 - Mixture of open source and licensed software
 - A common communication platform (share information)
- By implementing a version control system
 - All changes can be tracked and documented (who, when and what)

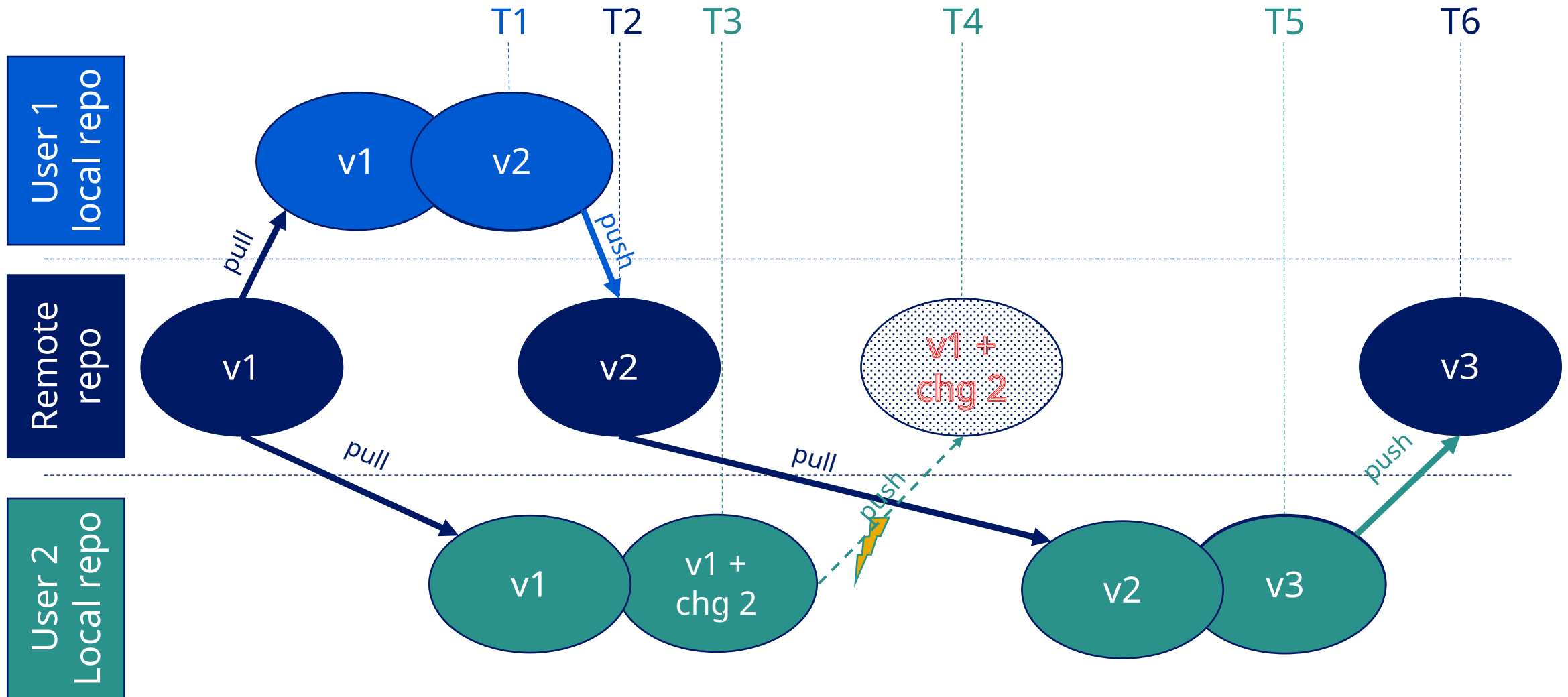


Version control and Agile mindset

Does not seem simple and Agile!
It is quite simple when you get use to the way of working
Seems complicated and time consuming!



Basic principle of version control

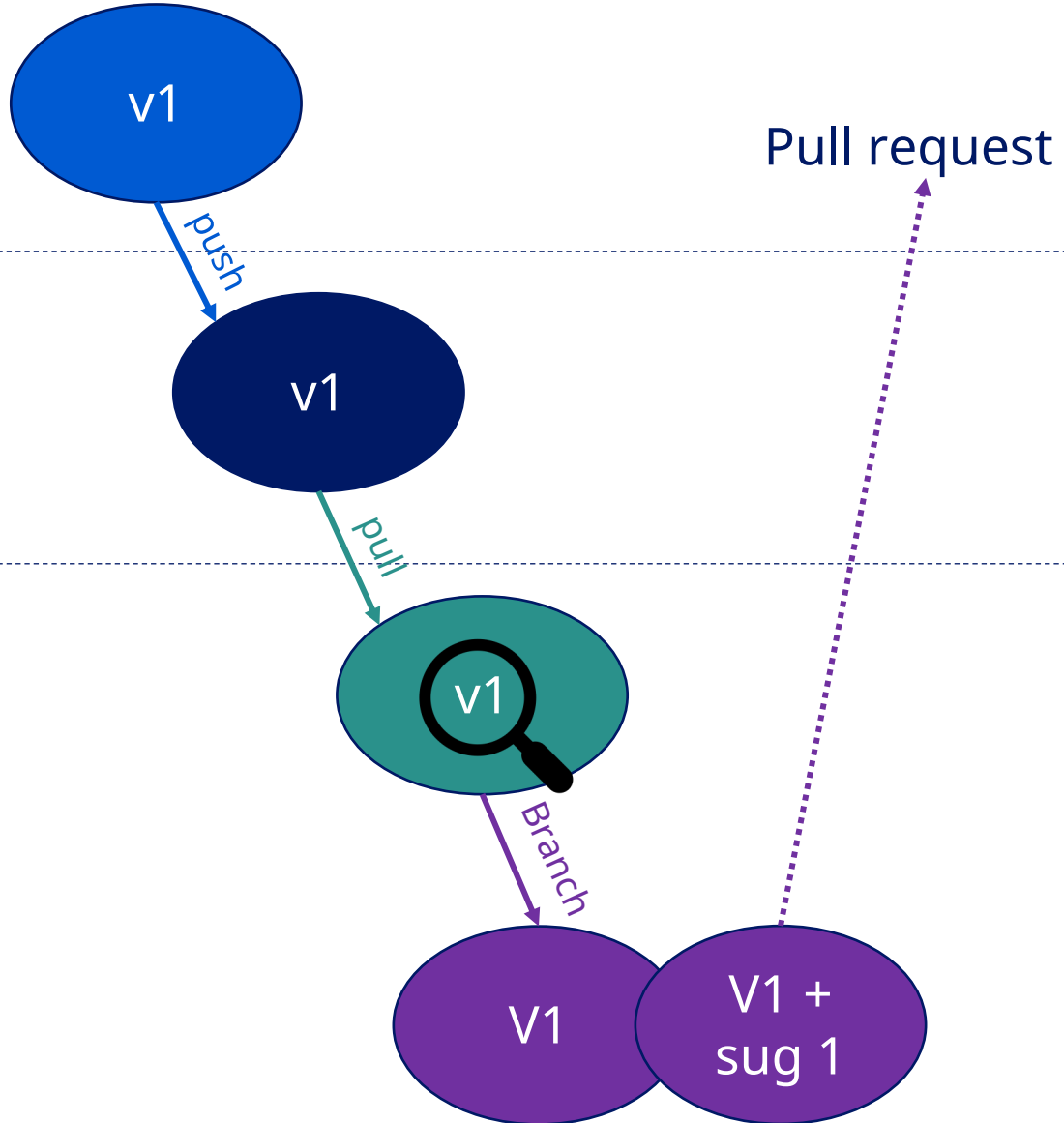
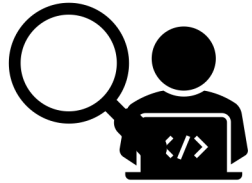


Programmer



Remote
repo

Reviewer

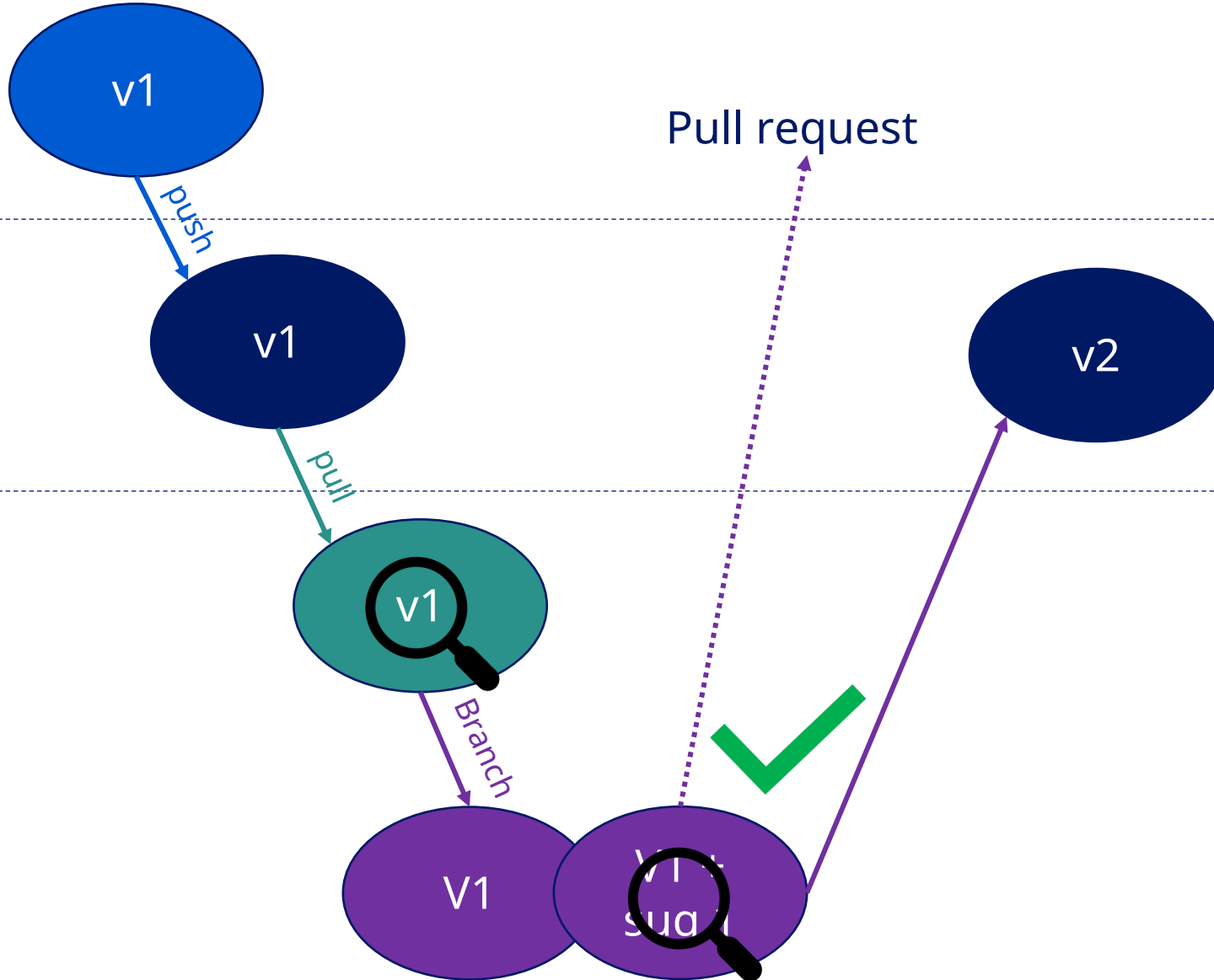
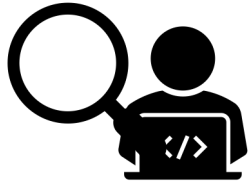


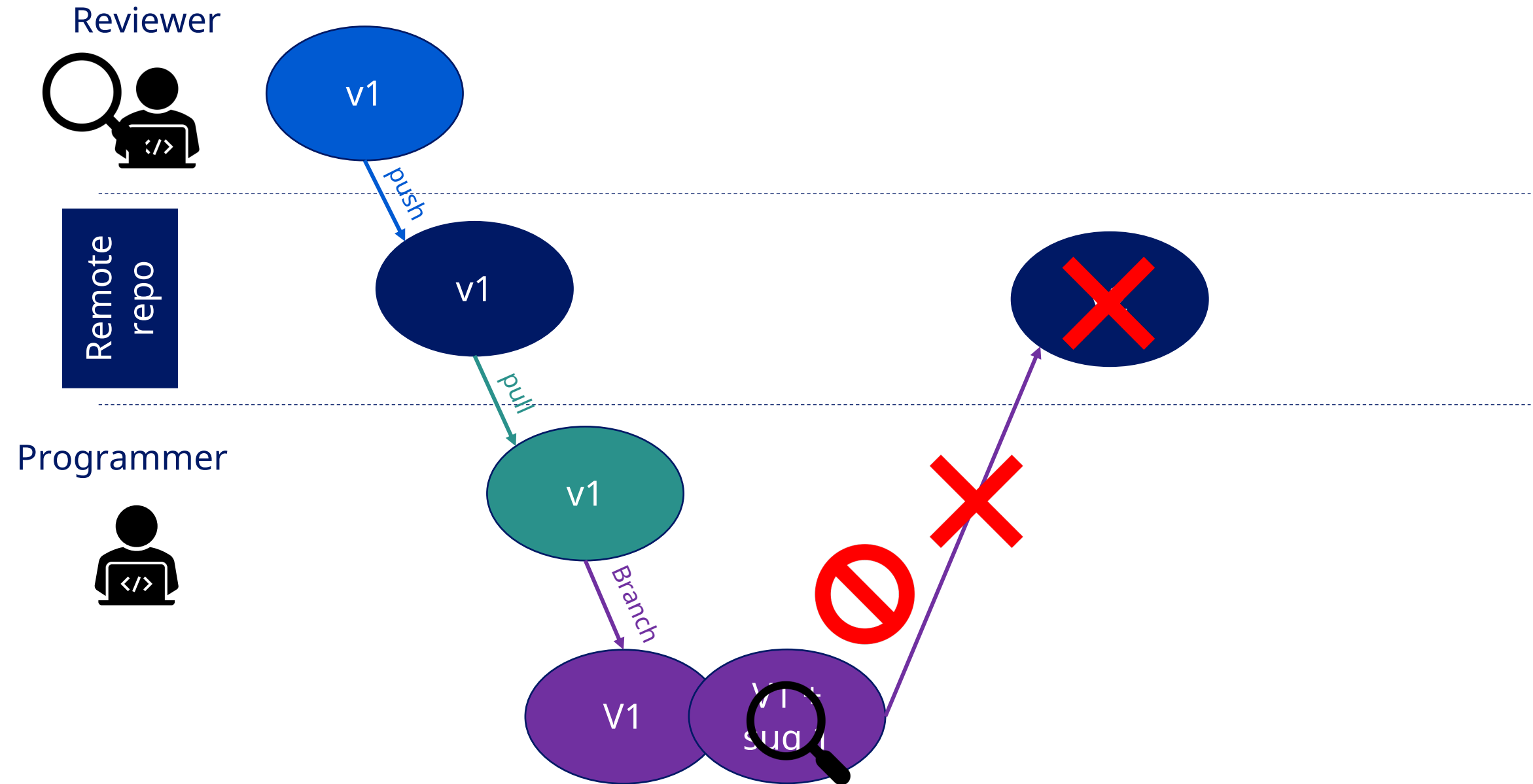
Programmer

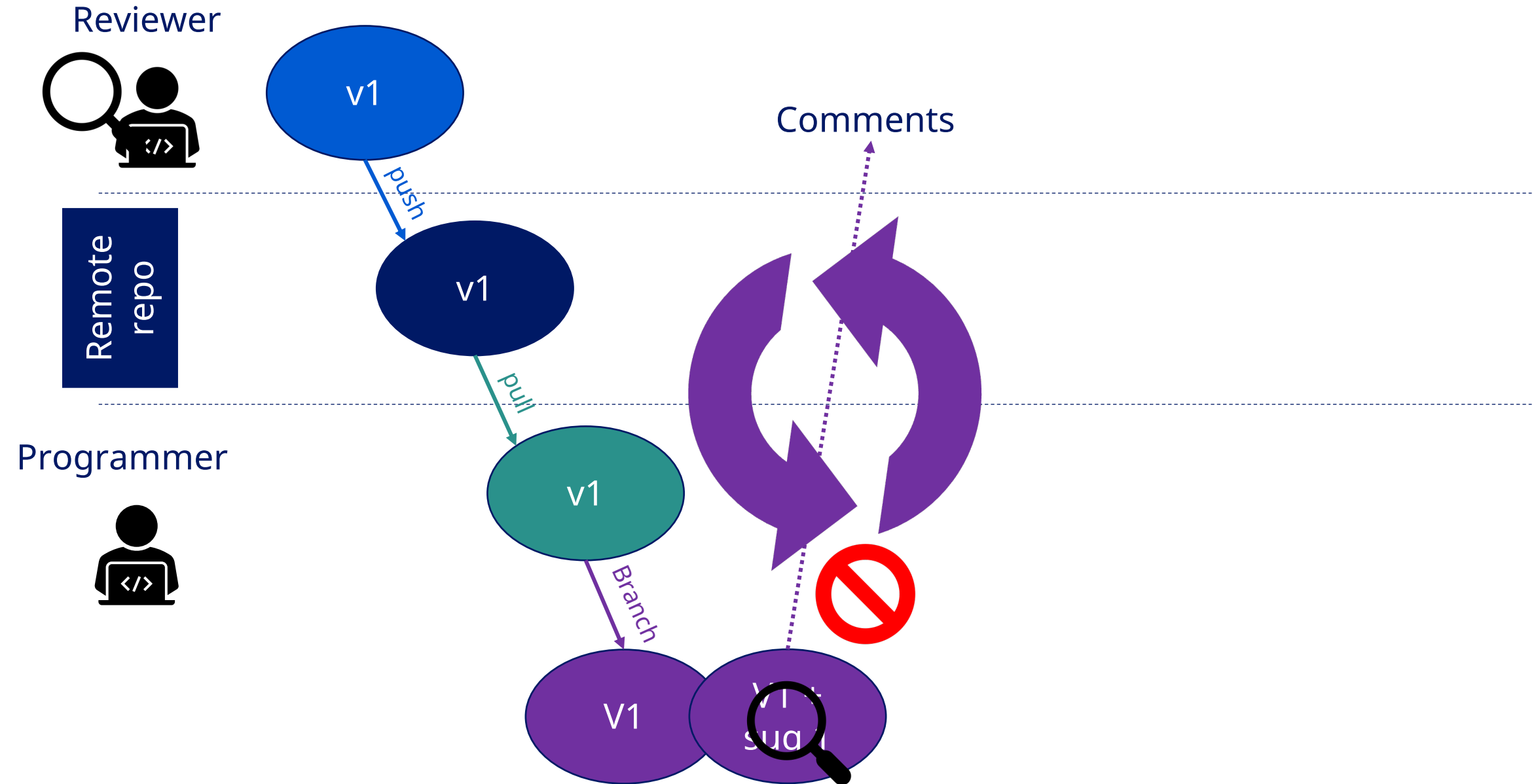


Remote
repo

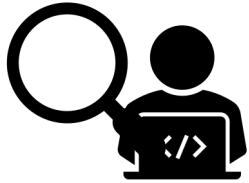
Reviewer







Reviewer



Comments

Remote
repo

v1

push

v1

pull

v1

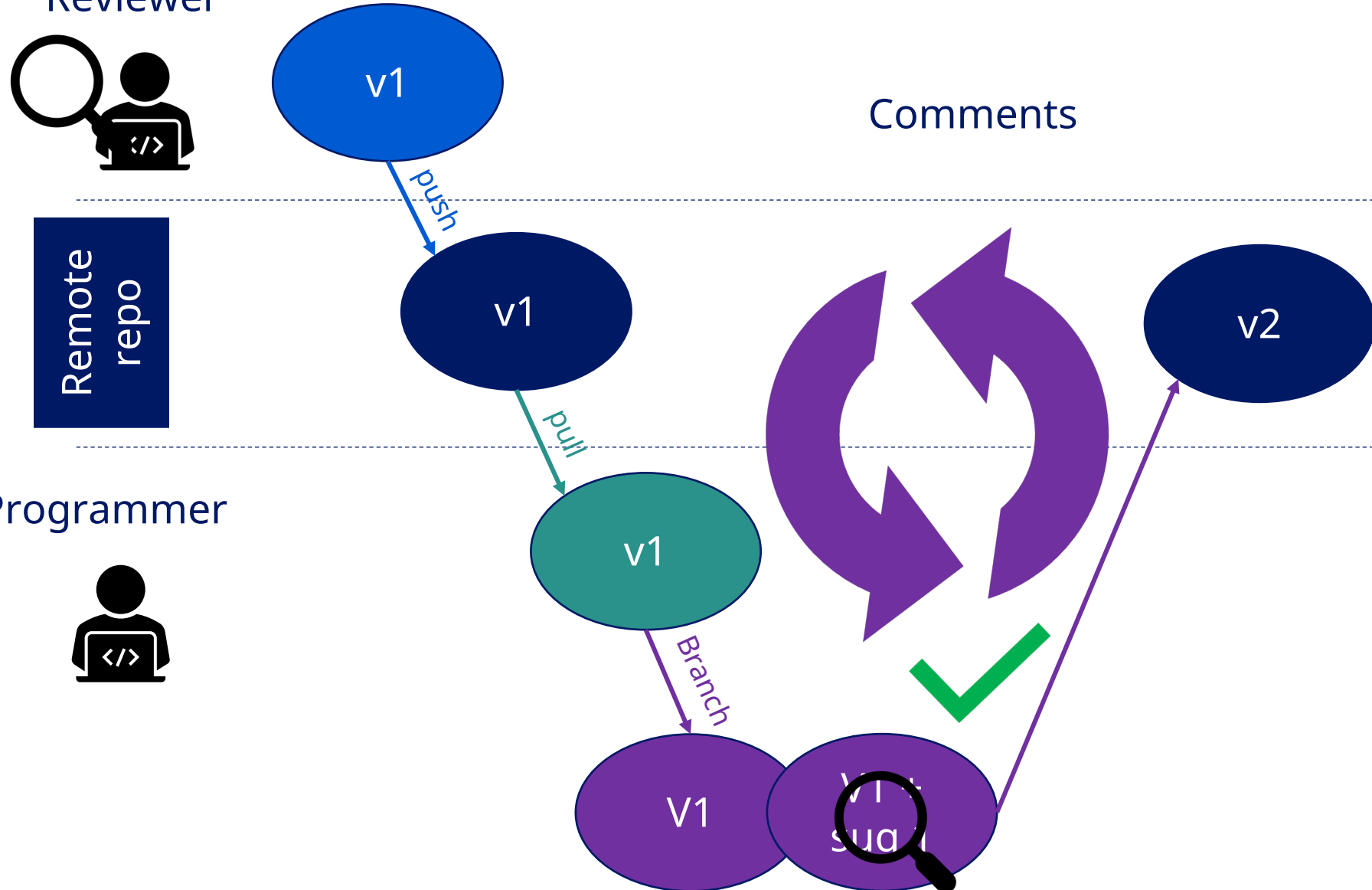
Branch

V1

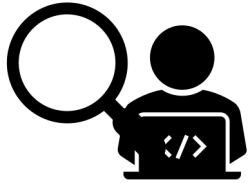
V1
suo

v2

Programmer

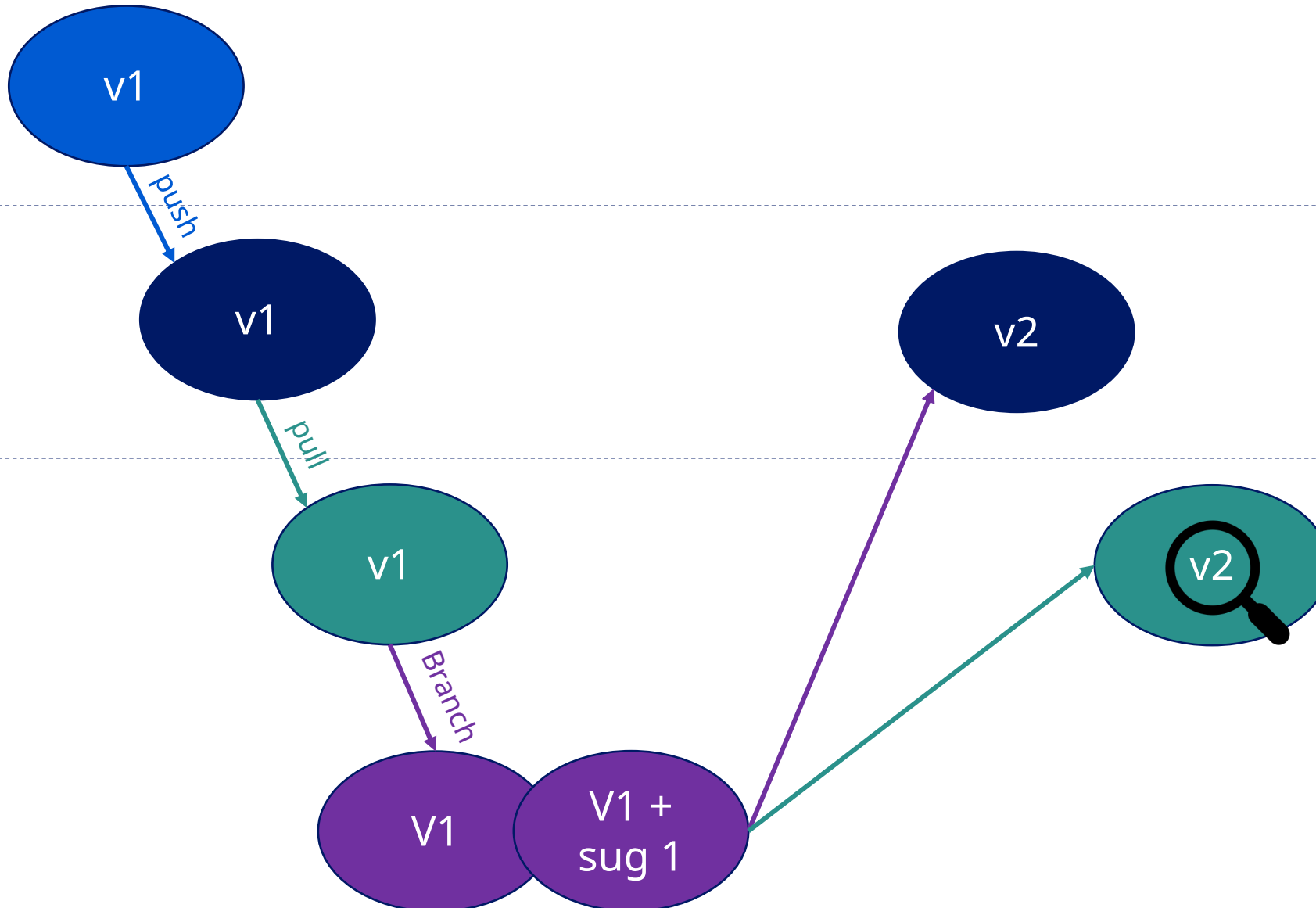


Reviewer



Remote
repo

Programmer



PROS

- Easier to explain and understand an issue in the code
 - Fewer emails, messages, discussions
- Version control software's makes it easier to document and handle collaboration between multiple programmers
 - Decentralization of workload (across team members, less vulnerable to a programmer become sick or leave)
 - Reduce loss of information and untracked changes particularly for large teams participating in code review.

CONS

- Poses the issue how of much of the code can the reviewer suggest changes to without taking over the main responsibility of the program?
 - Should there be an upper limit?
- Learning curve is steep
- All users need to use this setup
- Long-lived branches may be a nightmare to “merge”, lifetime of a branch should be minimal

What did we learn

- We believe version control softwares can:
 - Facilitate collaborative programming and track all changes to a program
 - Ensure transparency of who does what when
 - Ease the reviewer/programmer communication and issue management

Thank you for your attention



Questions → frhe@novonordisk.com