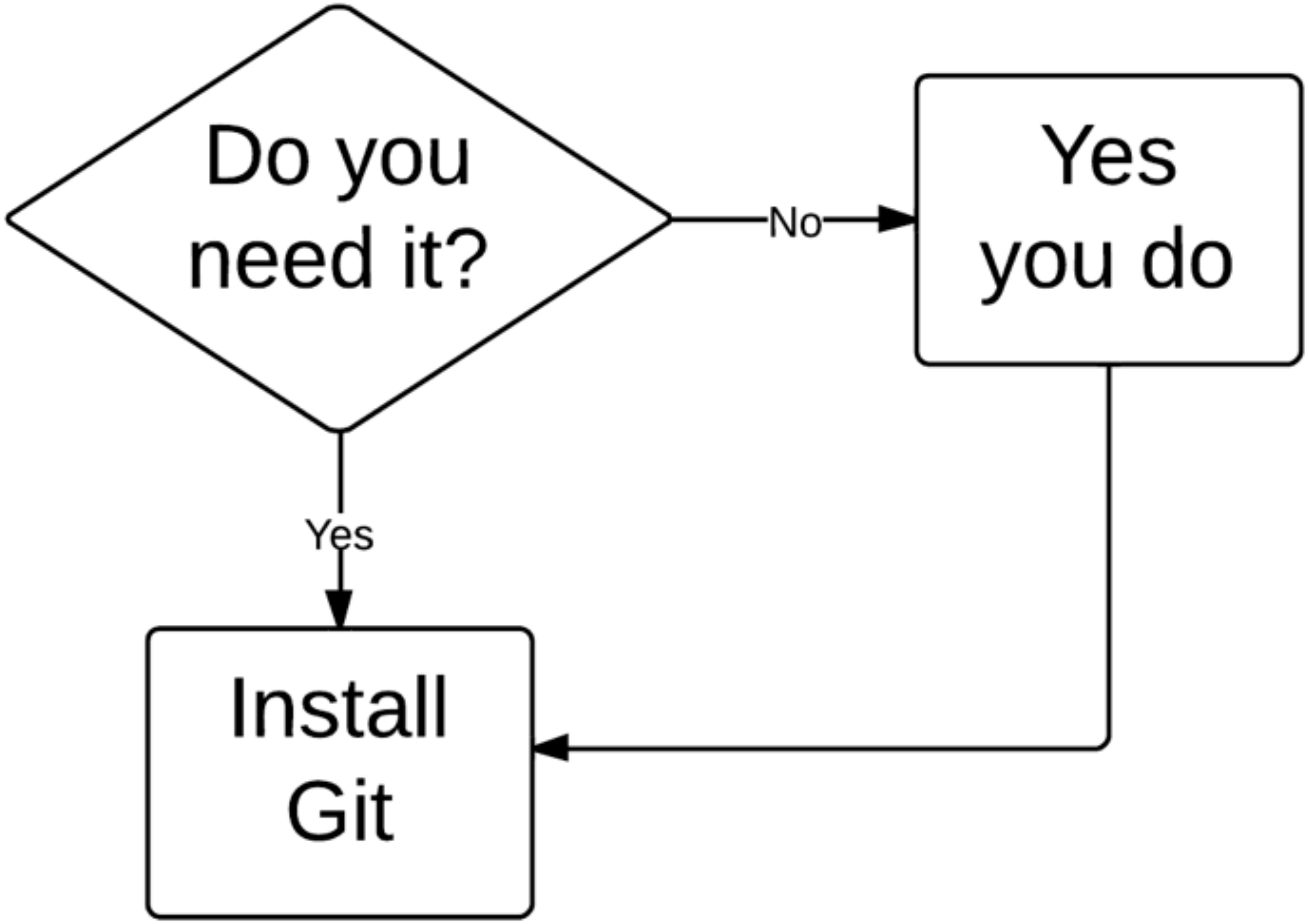
Version Control

RJ LeVeque, UWHPSC

Version Control Flowchart



Originally developed for large software projects with many developers.

Also useful for single user, e.g. to:

- Keep track of history and changes to files,
- Be able to revert to previous versions,
- Keep many different versions of code well organized,
- Easily archive exactly the version used for results in publications,
- Keep work in sync on multiple computers.

Original style, still widely used (e.g. CVS, Subversion)

One central repository on server.

Developers' workflow (simplified!):

- Check out a working copy,
- Make changes, test and debug,
- Check in (commit) changes to repository (with comments). This creates new version number.
- Run an update on working copy to bring in others' changes.

The system keeps track of diffs from one version to the next (and info on who made the changes, when, etc.)

A changeset is a collection of diffs from one commit.

Only the server has the full history.

The working copy has:

- Latest version from repository (from last checkout, commit, or update)
- Your local changes that are not yet committed.

Note:

- You can retrieve older versions from the server.
- Can only *commit* or *update* when connected to server.
- When you *commit*, it will be seen by anyone else who does an *update* from the repository.

Often there are trunk and branches subdirectories.

Git uses a distributed model:

When you clone a repository you get all the history too,

All stored in .git subdirectory of top directory.
  Usually don’t want to mess with this!

```
git clone https://github.com/ComputoCienciasUniandes/MetodosComputacionales
```

This directory has a subdirectory .git with complete history.

Git uses a distributed model:

- git commit commits to your clone's .git directory.
- git push sends your recent changesets to another clone by default: the one you cloned from (e.g. bitbucket), but you can push to any other clone (with write permission).
- git fetch pulls changesets from another clone by default: the one you cloned from (e.g. bitbucket)
- git merge applies changesets to your working copy

Note: pushing, fetching, merging only needed if there are multiple clones.

Advantages of distributed model:

- You can commit changes, revert to earlier versions, examine history, etc. without being connected to server.
- Also without affecting anyone else's version if you're working collaboratively. *Can commit often while debugging.*
- No problem if server dies, every clone has full history.

For collaboration will still need to push or fetch changes eventually and git merge may become more complicated.

GitHub

Many open sources projects use it, including Linux kernal.

(Git was developed by Linus Torvalds for this purpose!)

Many open source scientific computing projects use github, e.g.

- IPython
- NumPy, Scipy, matplotlib