REPRODUCIBILE WORKFLOWS

Version Control and Computational Notebooks

John Little
Duke University Libraries
Center for Data & Visualization Sciences

2024-02-14

ARTICLE PRODUCTION

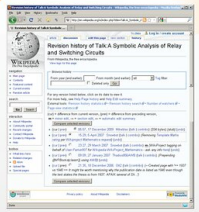
[[A Symbolic Analysis of Relay and Switching Circuits]]

Article Page

Talk Page

Latest revision

Edit

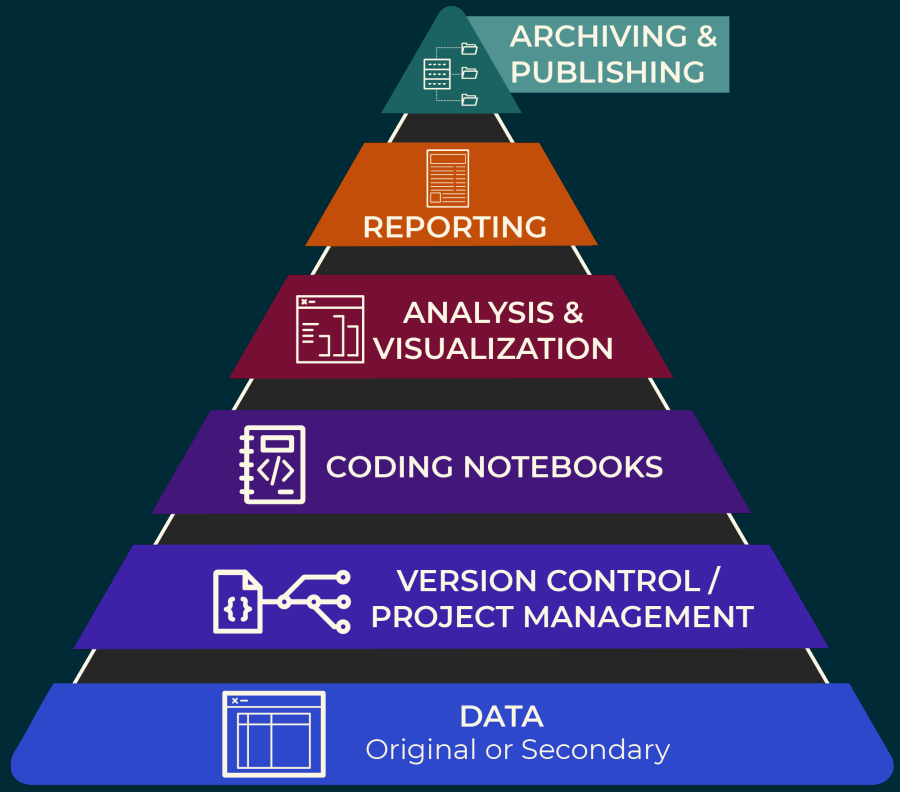


History

Earlier revisions

REPRODUCTION

Authoring and computation environment should enable the articulation of scholarship within a reproducible context



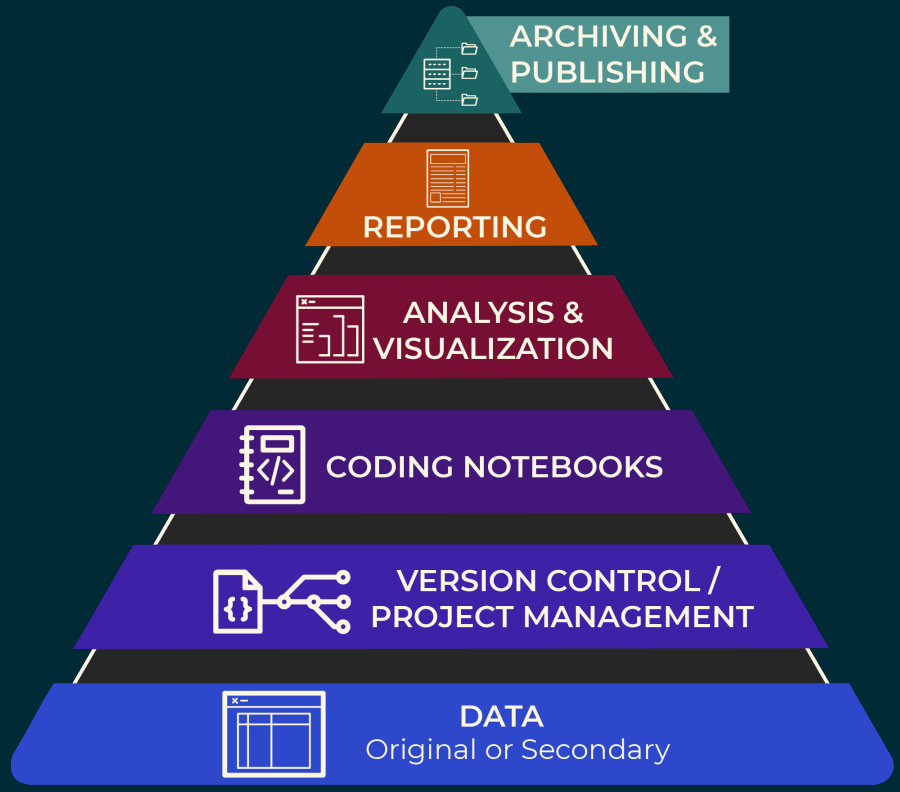
Reproducibility Pyramid • Little & Lafferty-Hess (2020)

FEATURES

- Support composable recombination
- Accommodate multimedia expression
- Provide rich reporting expressions
- Support economical portability and degrade gracefully
- Support extensibility
- Ensure transparency
- Support a documentary-style project history
- Accommodate change and collaboration
- Be citable

THREE POINTS

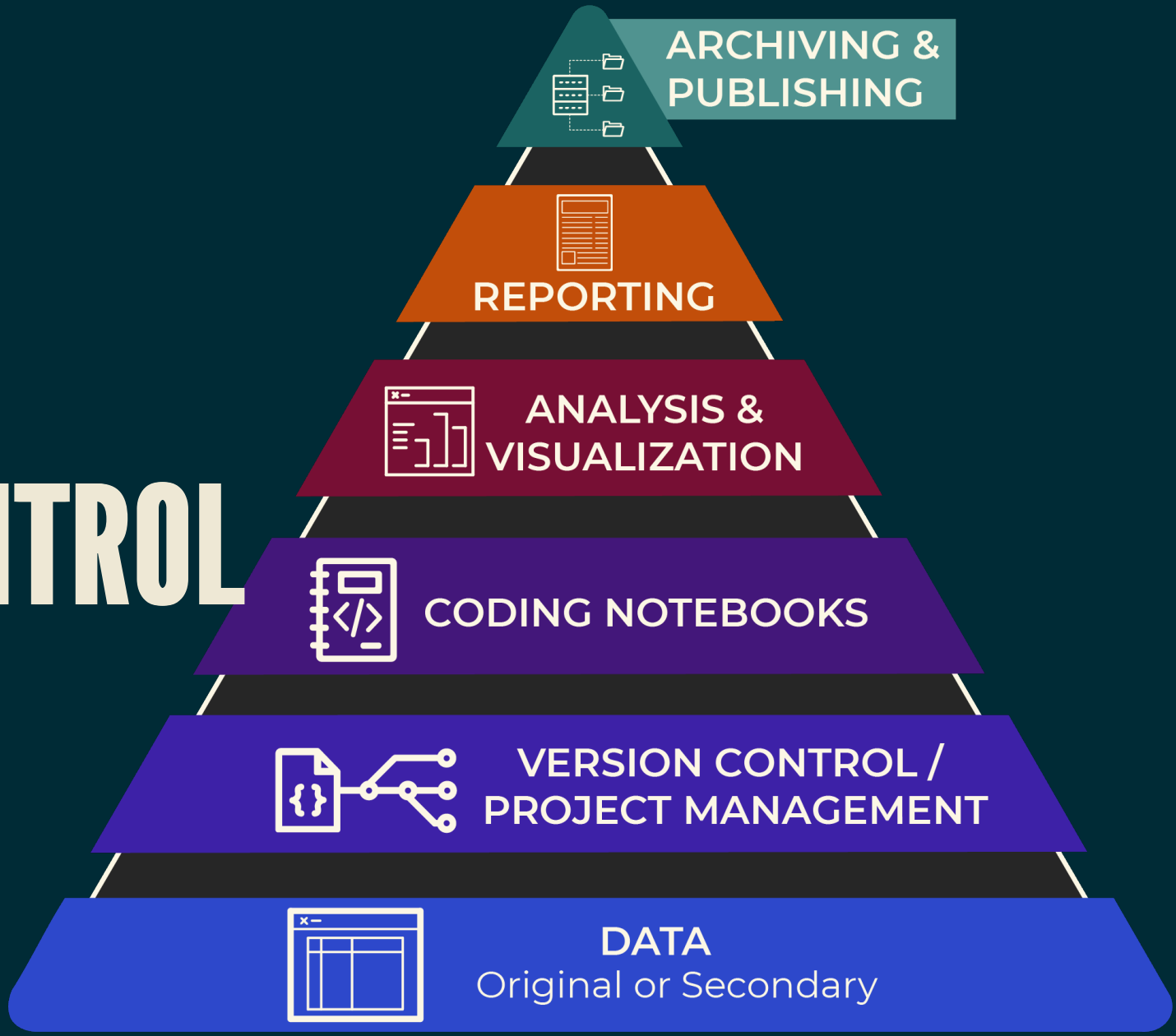
1. Version Control (Git & GitHub)
2. Notebooks (Literate Coding)
3. Archiving & Publishing (Zenodo, Containers)



Reproducibility Pyramid • Little & Lafferty-Hess (2020)

PREVIEW

VERSION CONTROL



CHARACTERISTICS OF VERSION CONTROL

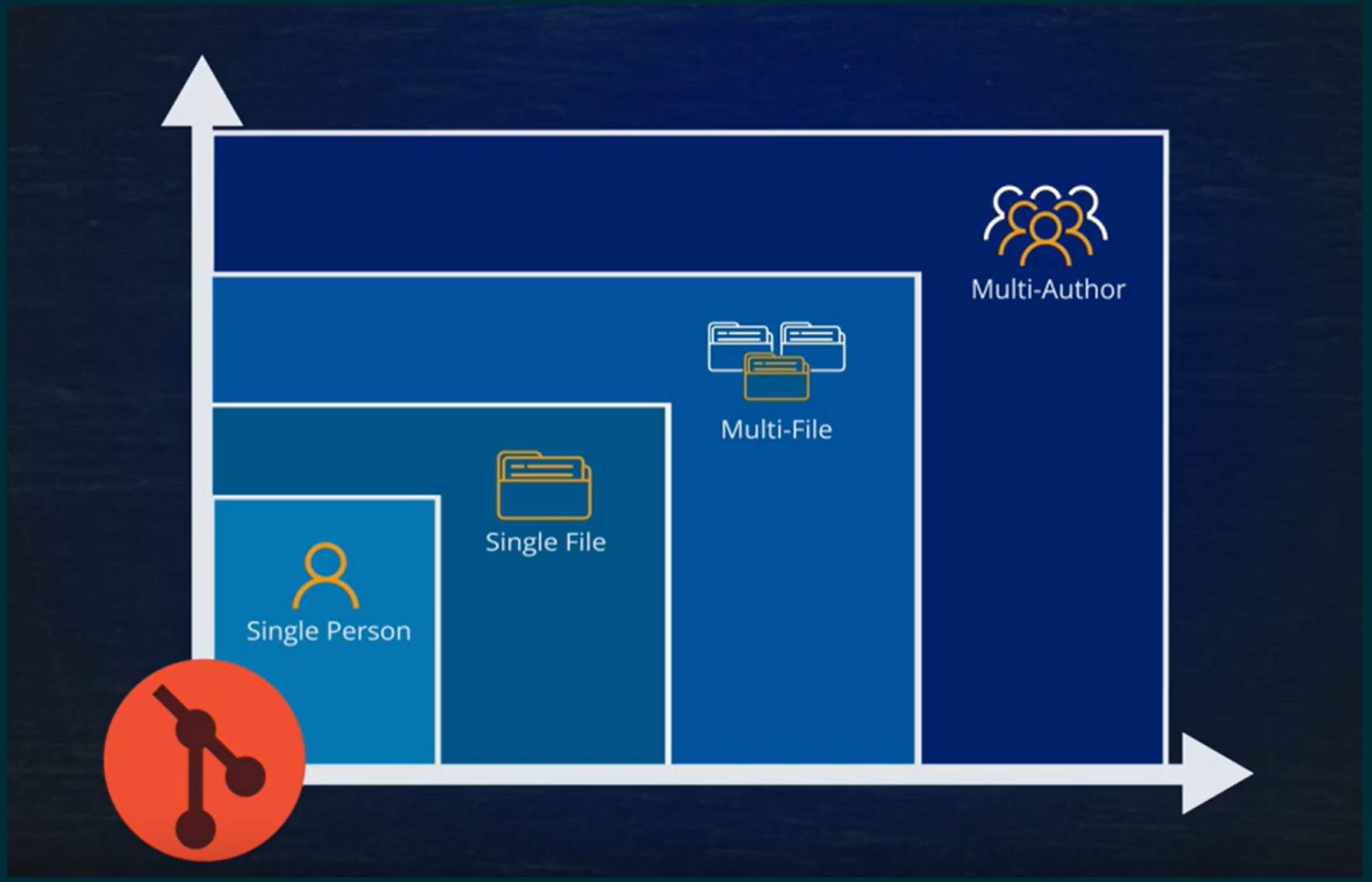
- A system to **manage projects** (repo)
- A system to track how computer files **change over time**
- A system that supports **collaborative revision**

- More than file synchronization
- Assists in project back-ups

GIT

- Free open source
- Wildly successful; most broadly implemented
- In use across the globe
- Use it on any file system
- Track any file
- Use it in any environment

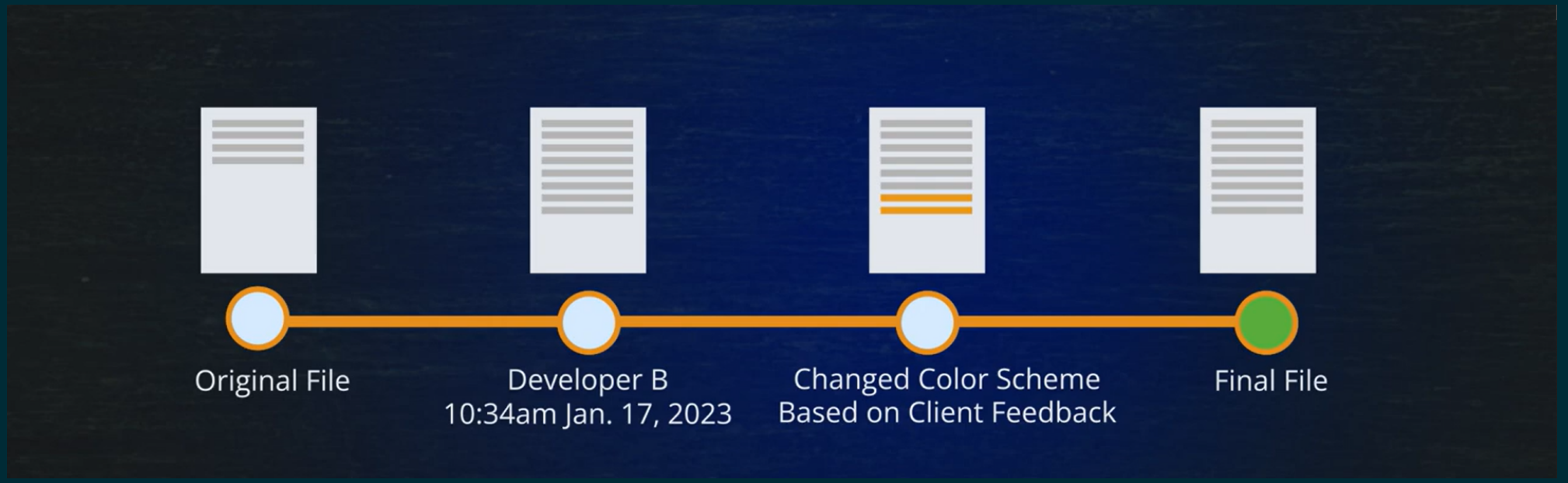
SCALABLE TO PROJECT SIZE



PROJECT REPOSITORIES

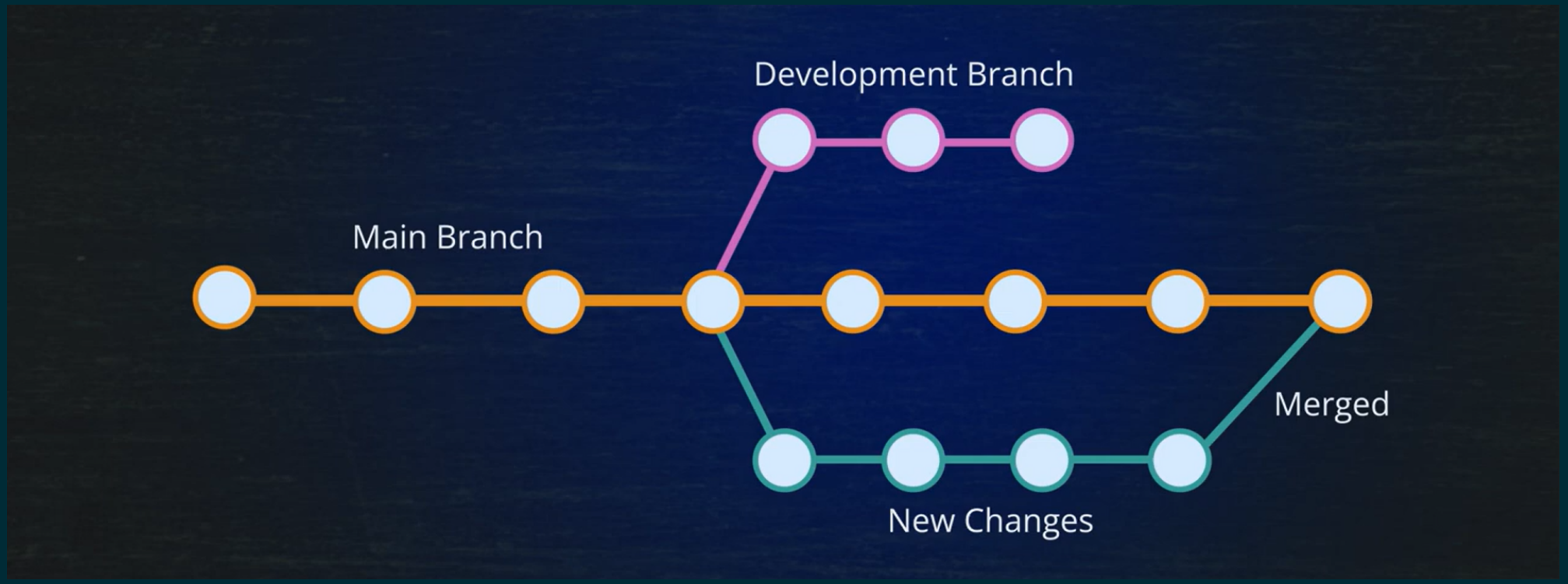
- Work on any file system
- Operates on at the folder level

TRACK CHANGE



When, Who, Why, What

BRANCHES

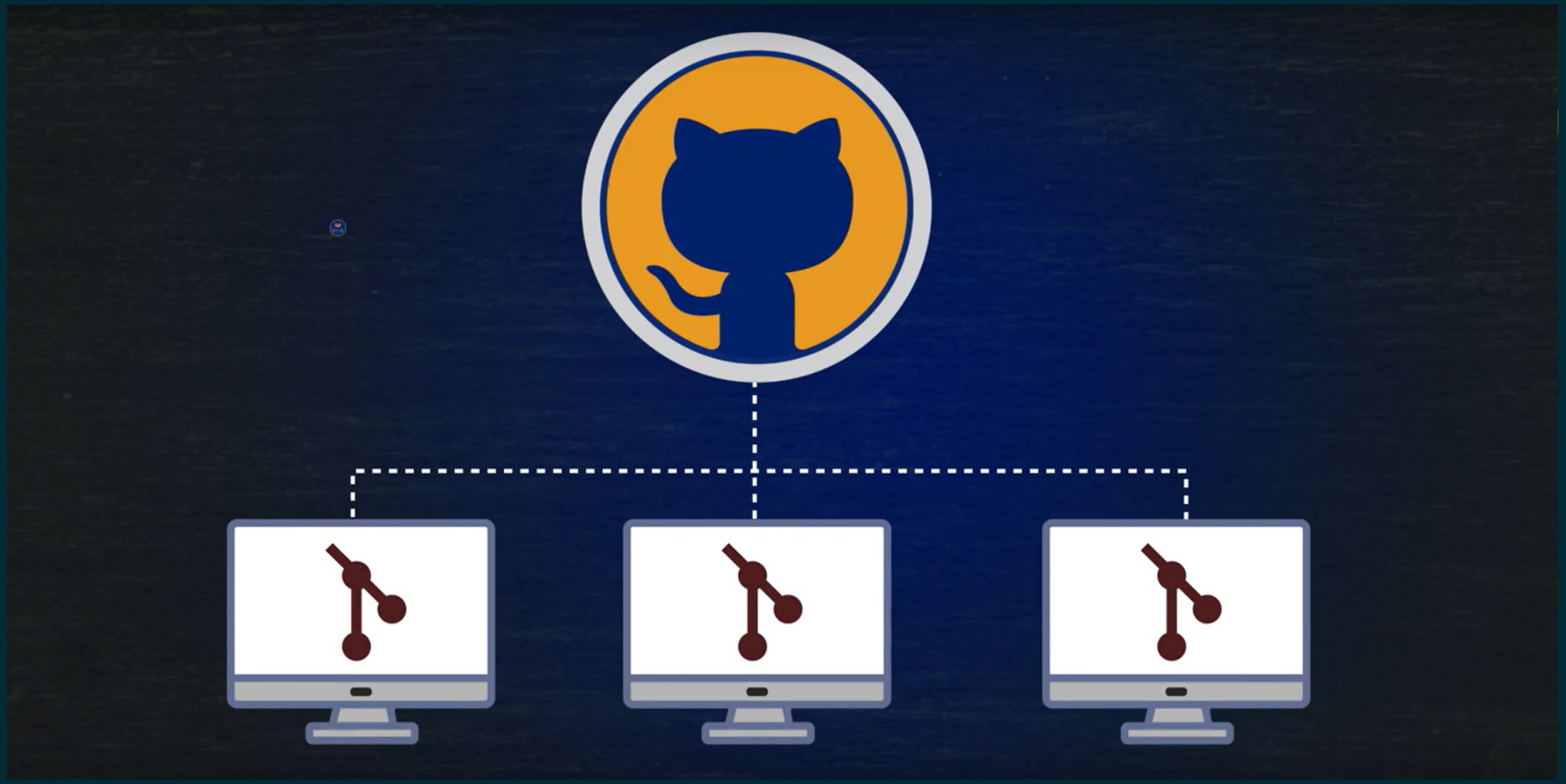


- Main branch
- Experimentation
- Developments
- Merging

GITHUB

- Profile (store and host) git repos
- Enable collaboration across the globe or private
- Editorial and fine-grain control

GIT + GITHUB



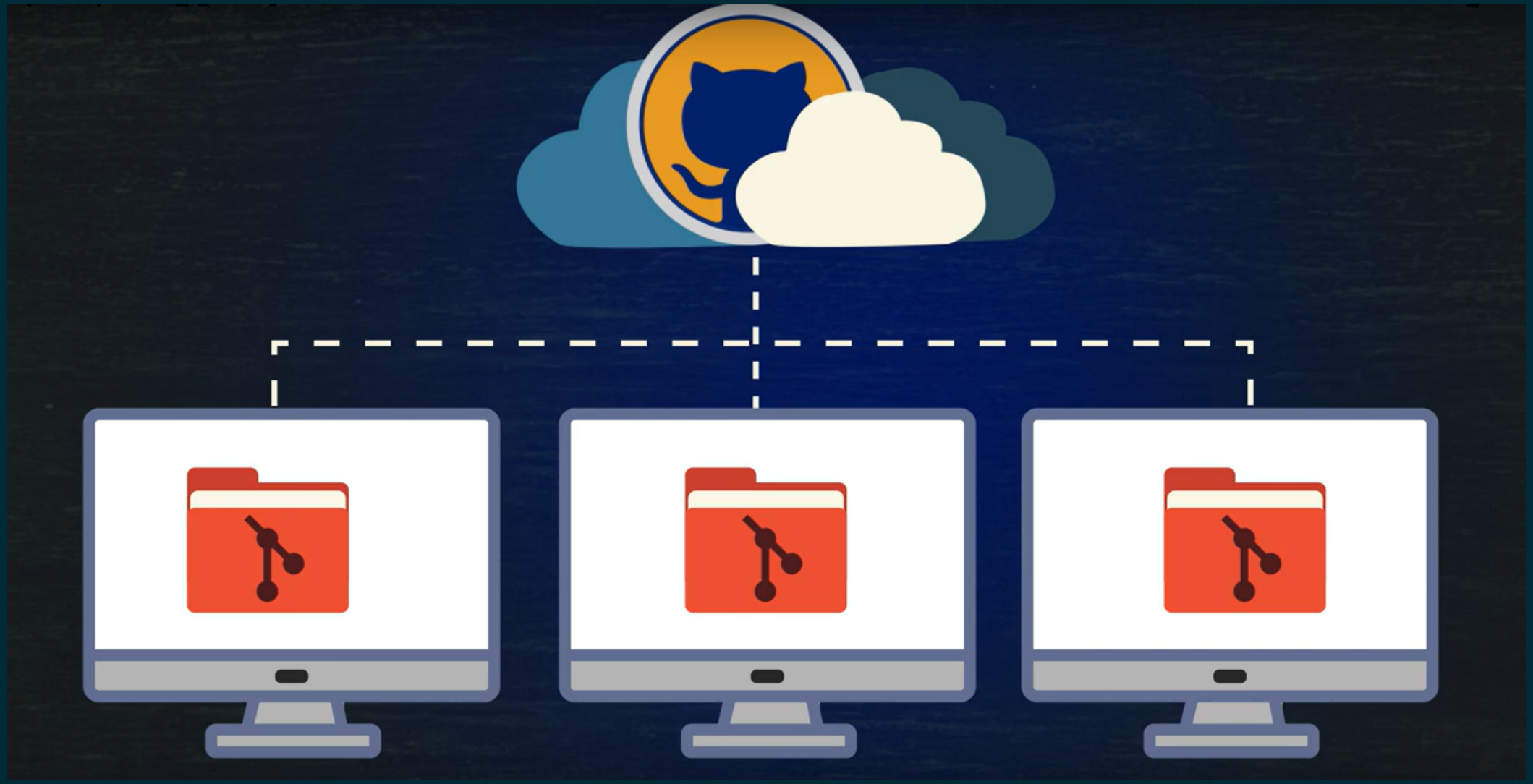
HUBS

- GitHub
- GitLab
- BitBucket

DUKE SPECIFIC HUBS

- gitlab.oit.duke.edu (NetID)
- PACE
- Anywhere that data and coding happens.

FILE DISTRIBUTION AND COLLABORATION



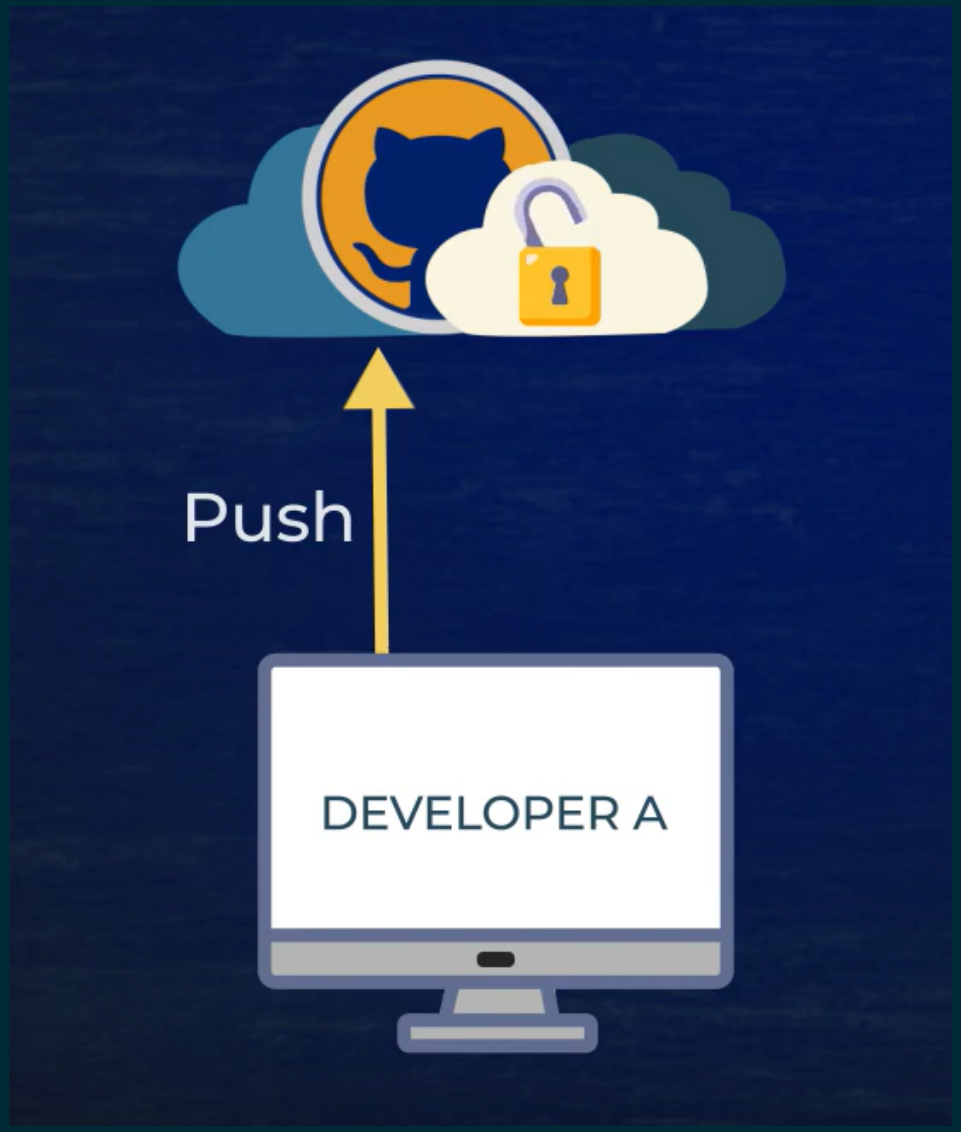
OTHER PROJECT MANAGEMENT FEATURES

BASIC FEATURES

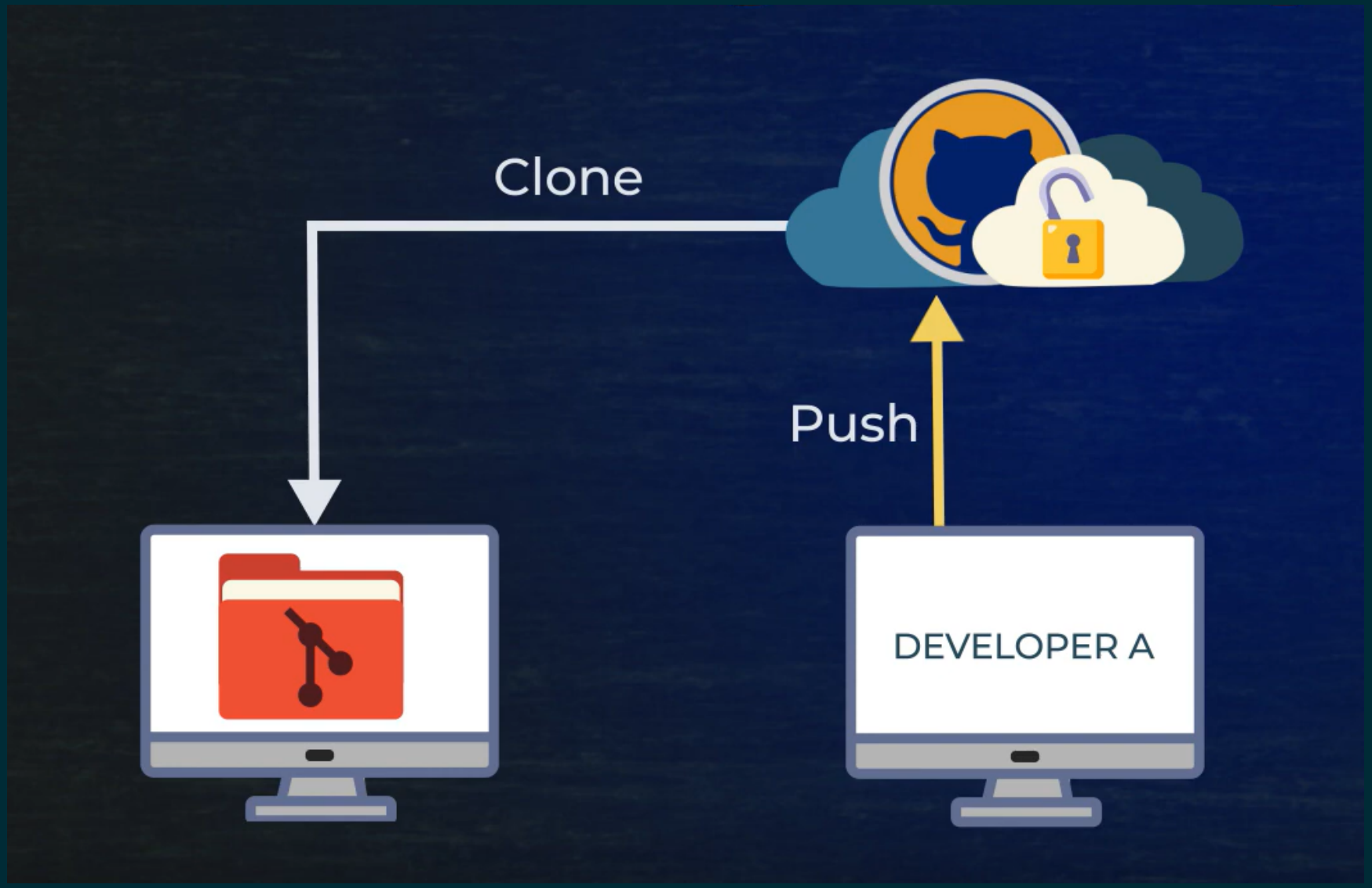
Git features implemented for distribution

- Push
- Public or Private
- Clone / Fork
- Pull Request
- Pull

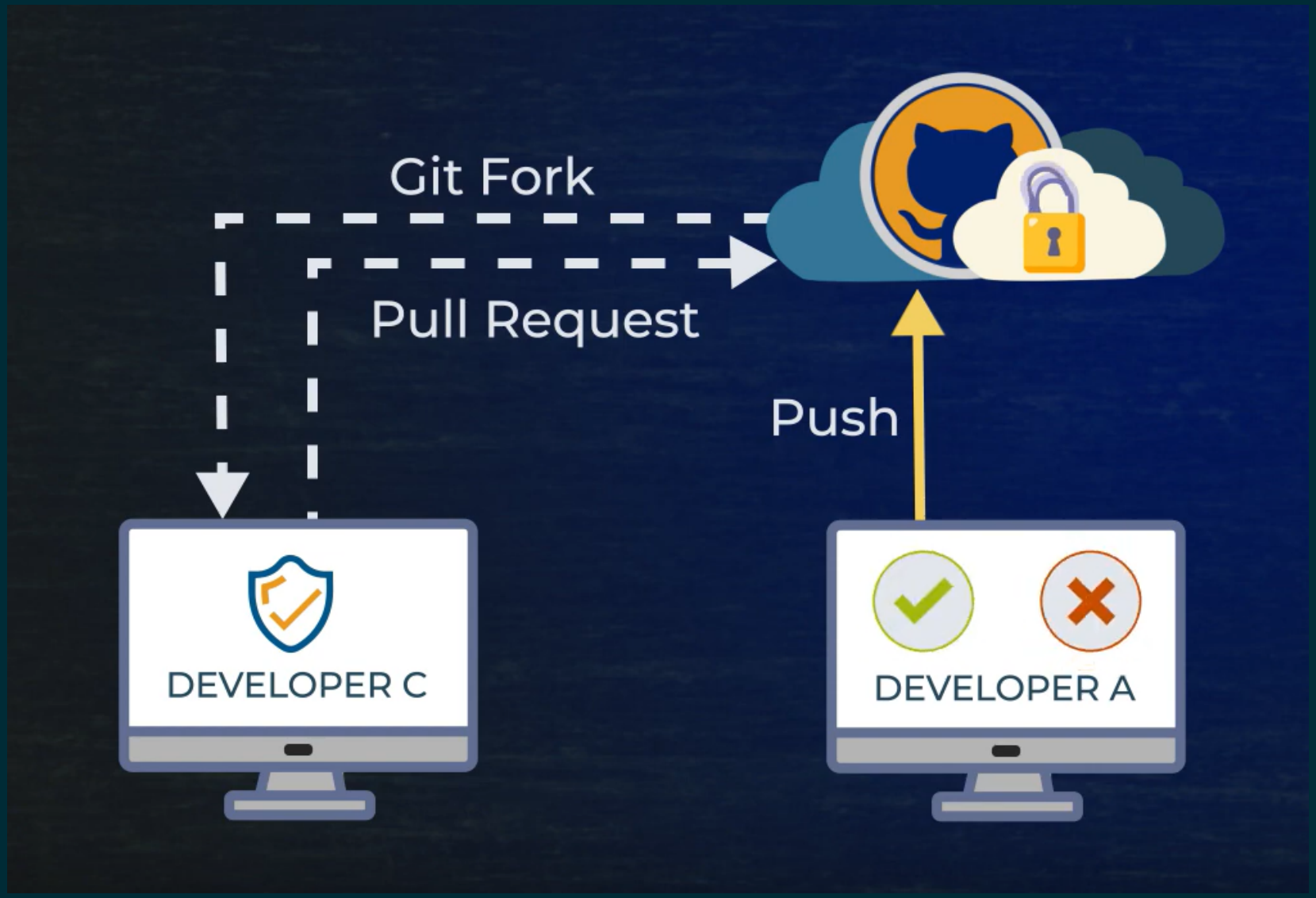
PUSH



CLONE



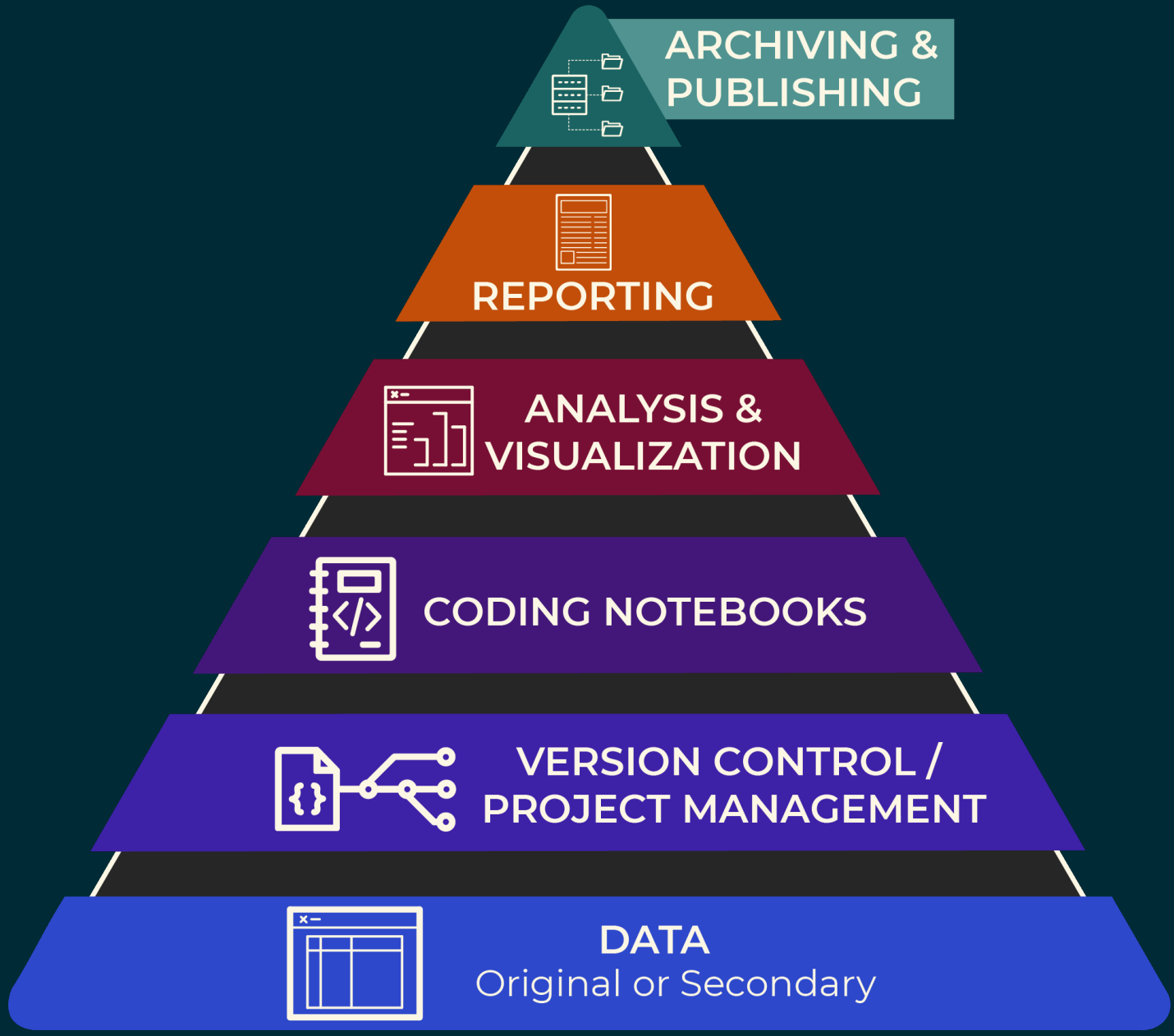
FORK / PR



SUMMARY

- Git is used to track changes to your repo
- GitHub is used to distribute your git repo and facility collaboration

NOTEBOOKS



REPRODUCIBILITY

- **Do everything with code!**
  - Helps reduce repetion errors
  - Helps avoid copy/paste barriers
  - Orchestrate workflows

COMPUTATIONAL NOTEBOOKS

- Authoring environment
  - Code chunks interspersed with natural language
  - aka *Literate Coding*
- Easy to read and compose
- Graceful degradation

REPORTS AND EXPRESSIONS

ort expressions are rendered at code execution

INTERACTIVITY AND WEB APPLICATIONS

- Shiny
- Flask
- WebR
- Plotly Dash
- ObjservableJS

QUARTO NOTEBOOK IN RSTUDIO

```
---
title: "Quarto Computations"
---
```

his dataset contains a subset of the fuel economy data from the EPA. pecifically, we use the `mpg` dataset from the **ggplot2** package.

```{r}
#| label: load-packages

library(ggplot2)
```

he visualization below shows a positive, strong, and linear relationship etween the city and highway mileage of these cars. Additionally, mileage is igher for cars with fewer cylinders.

```{r}
#| label: scatterplot

ggplot(mpg, aes(x = hwy, y = cty, color = cyl)) +
  geom_point(alpha = 0.5, size = 2) +
  scale_color_viridis_c() +
  theme_minimal()
```

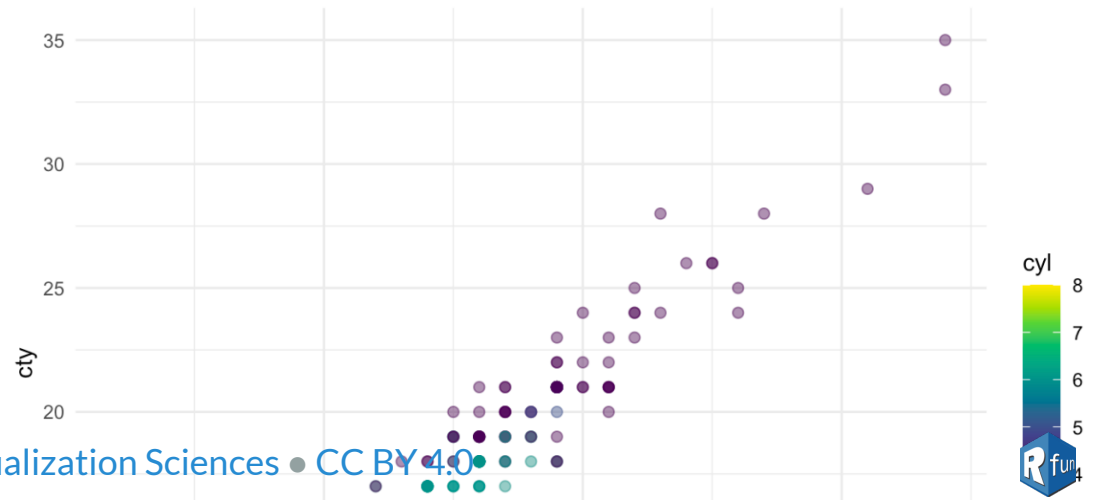
Quarto Computations

This dataset contains a subset of the fuel economy data from the EPA. Specifically, we use the `mpg` dataset from the **ggplot2** package.

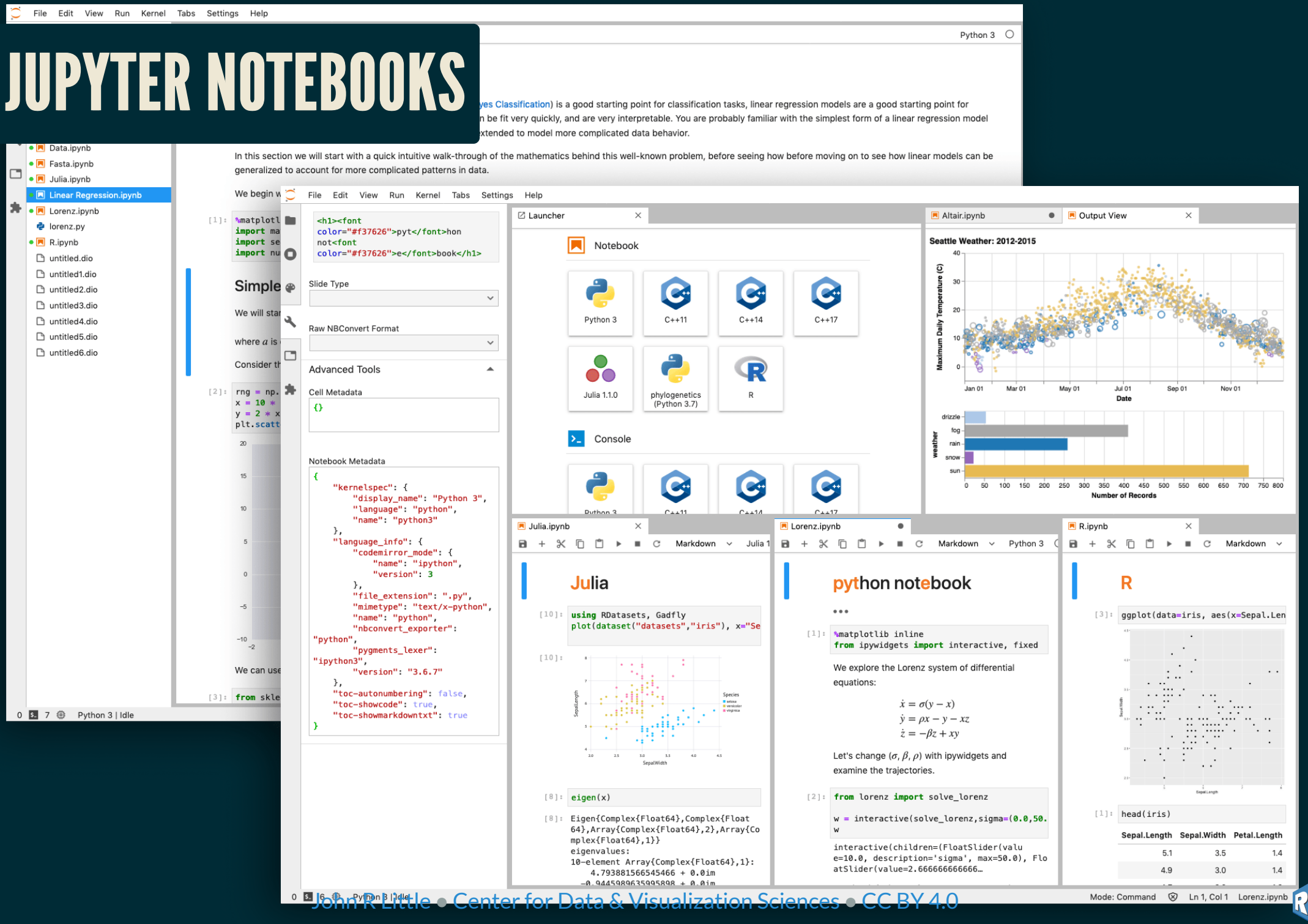
```
library(ggplot2)
```

The visualization below shows a positive, strong, and linear relationship between the city and highway mileage of these cars. Additionally, mileage is higher for cars with fewer cylinders.

```
ggplot(mpg, aes(x = hwy, y = cty, color = cyl)) +
  geom_point(alpha = 0.5, size = 2) +
  scale_color_viridis_c() +
  theme_minimal()
```



JUPYTER NOTEBOOKS



QUARTO

- A scientific publishing system
- R, Python, ObservableJS
- Compose with standard text editors, or basic IDEs
  - IDEs: RStudio, Jupyter, VSCode

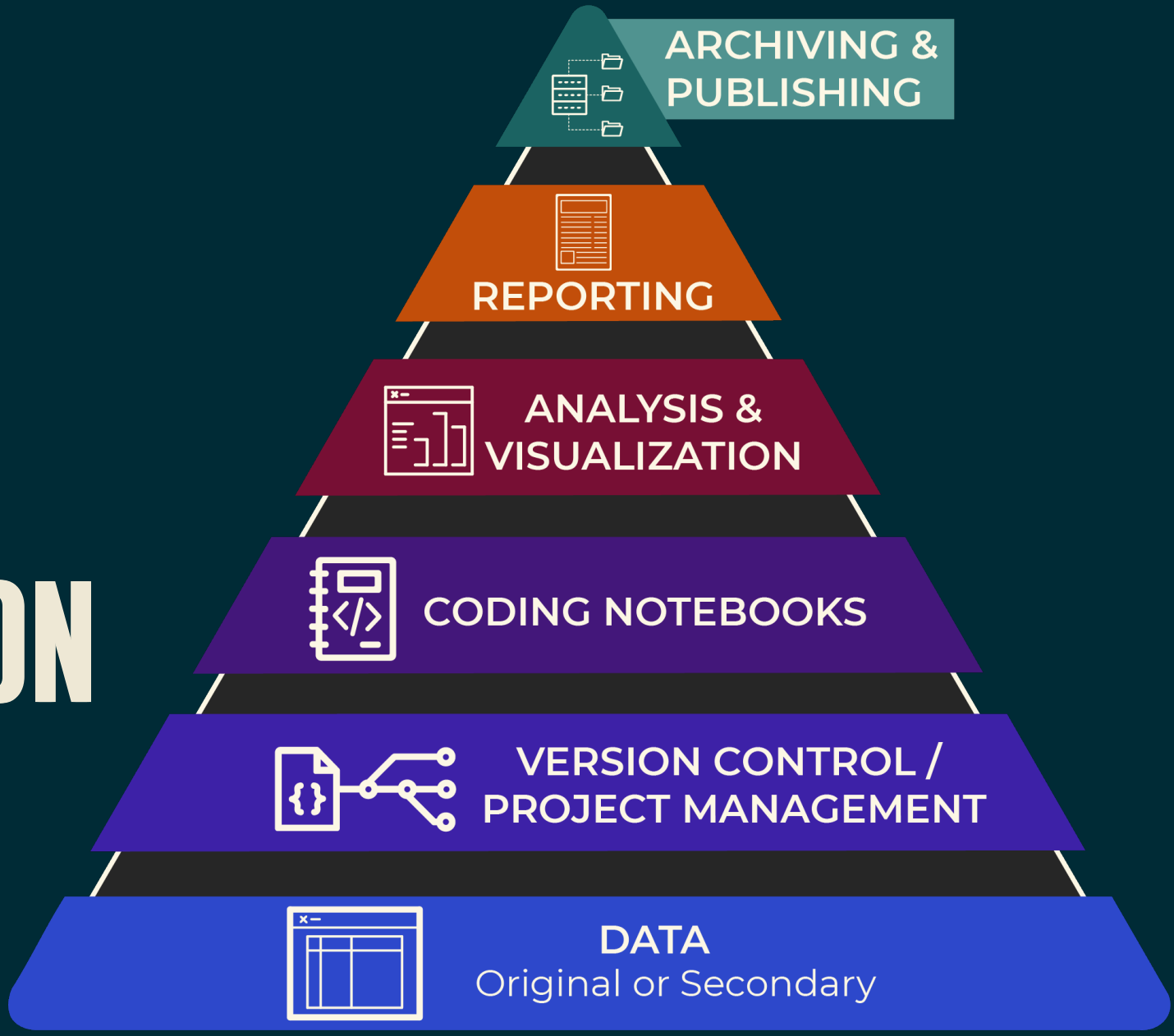
RENDERED OUTPUTS

- Artifacts that document a body of work
- Are reproducible and modifiable when data or techniques change
- Easy to update natural language explanations and re-render outputs
- Schedule emails based on report parameters

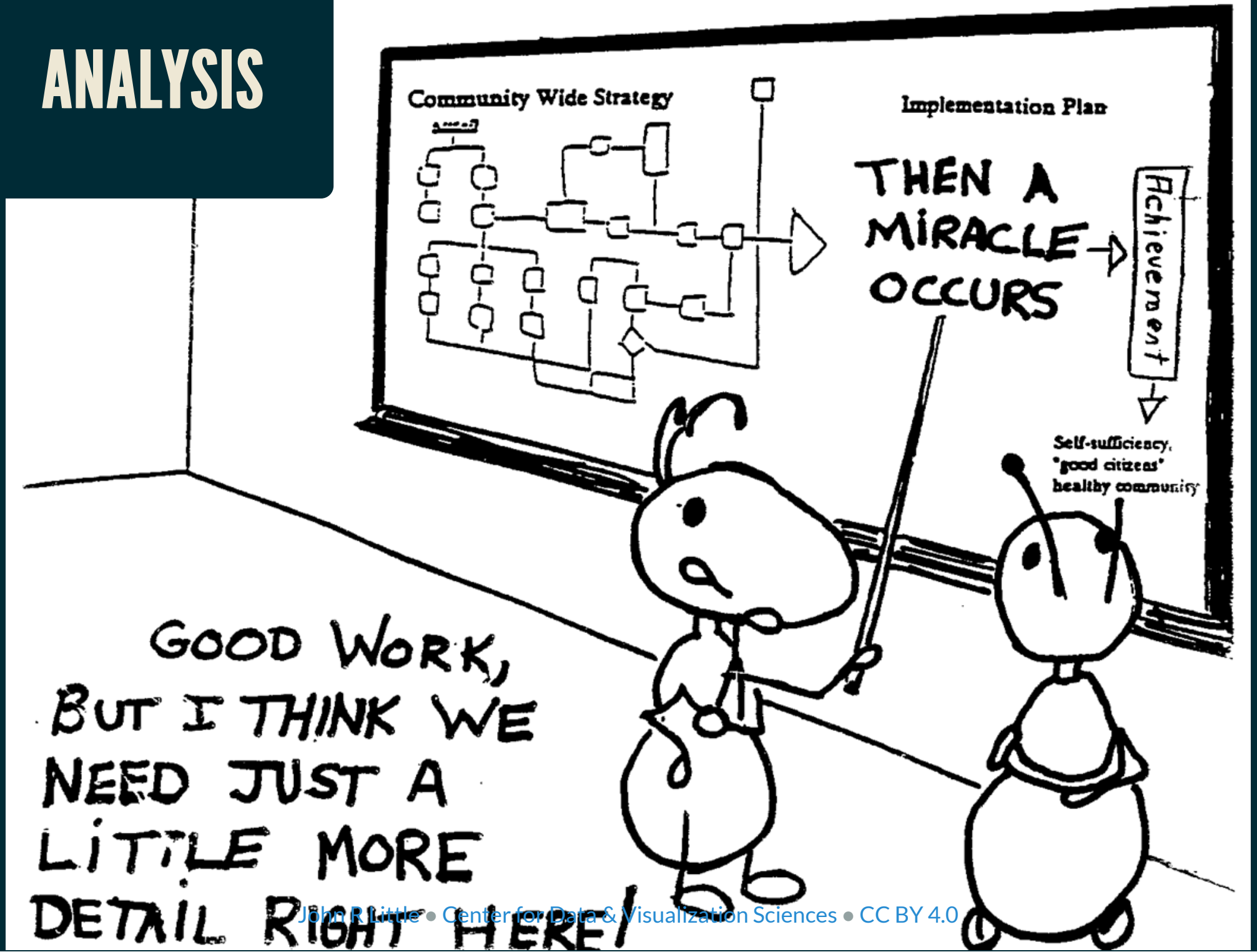
SUMMARY OF BENEFITS

- Using natural language clearly explain data, models, and workflows
- Reduce dependencies on outside and undocumented steps
- Ability to expose technical code chunks depending on audience focus
- State of the art reproducibility
  - 21st century **container** for evidence-based, computationally-processed research

ANALYSIS & VISUALIZATION

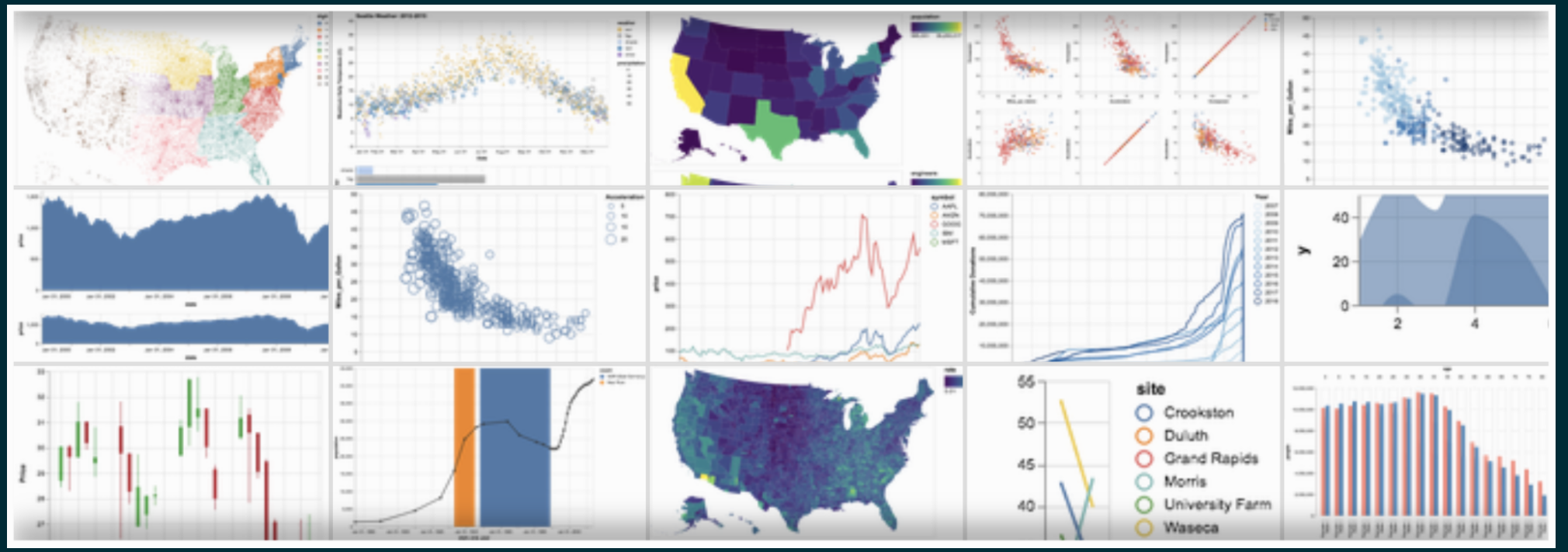


ANALYSIS

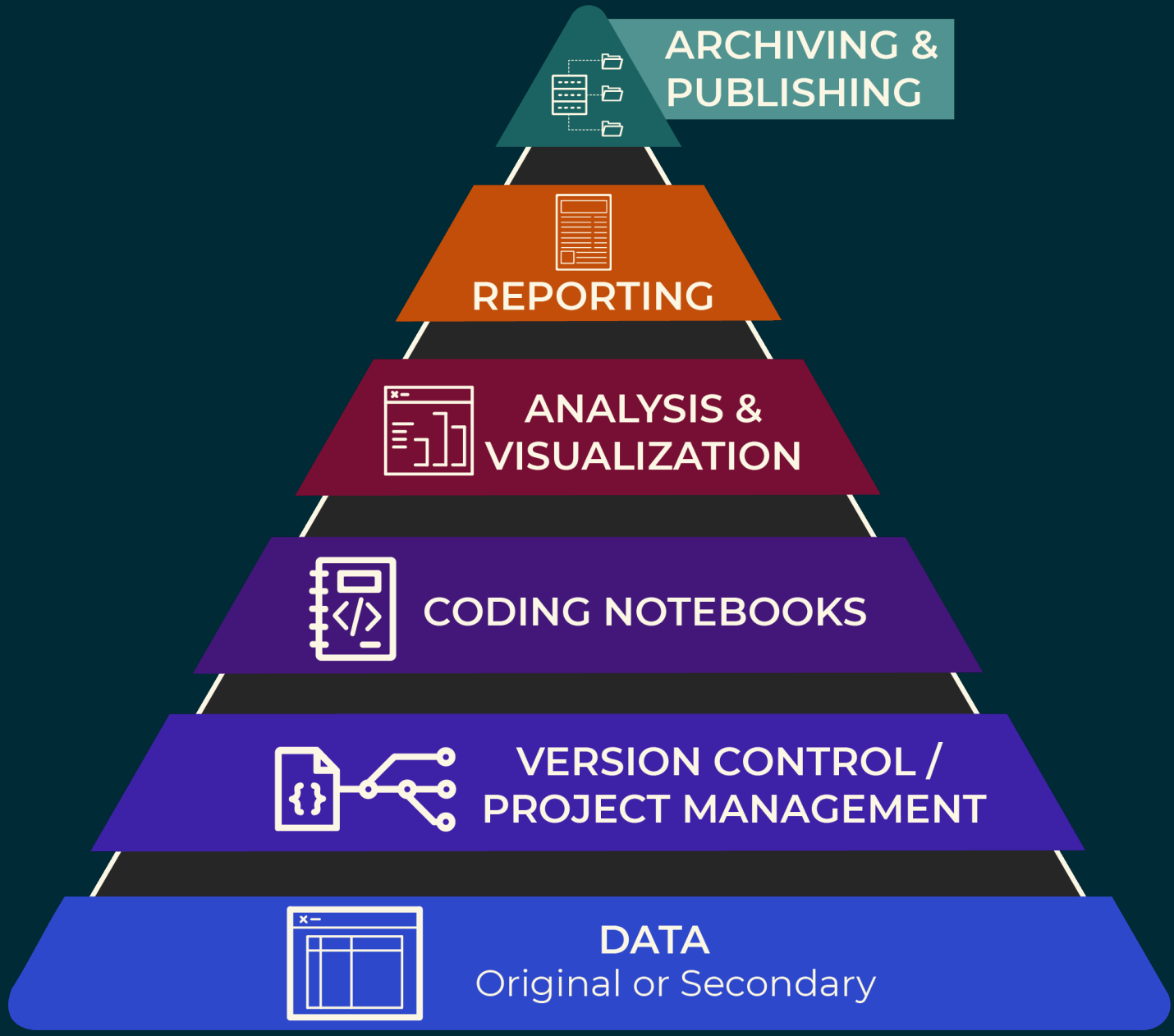

Community Wide Strategy
Implementation Plan
THEN A MIRACLE OCCURS
Achievement
Self-sufficiency,
"good citizens"
healthy community
GOOD WORK, BUT I THINK WE NEED JUST A LITTLE MORE DETAIL RIGHT HERE!

VISUALIZATION

Use graphics tools predicated on the grammar of graphics

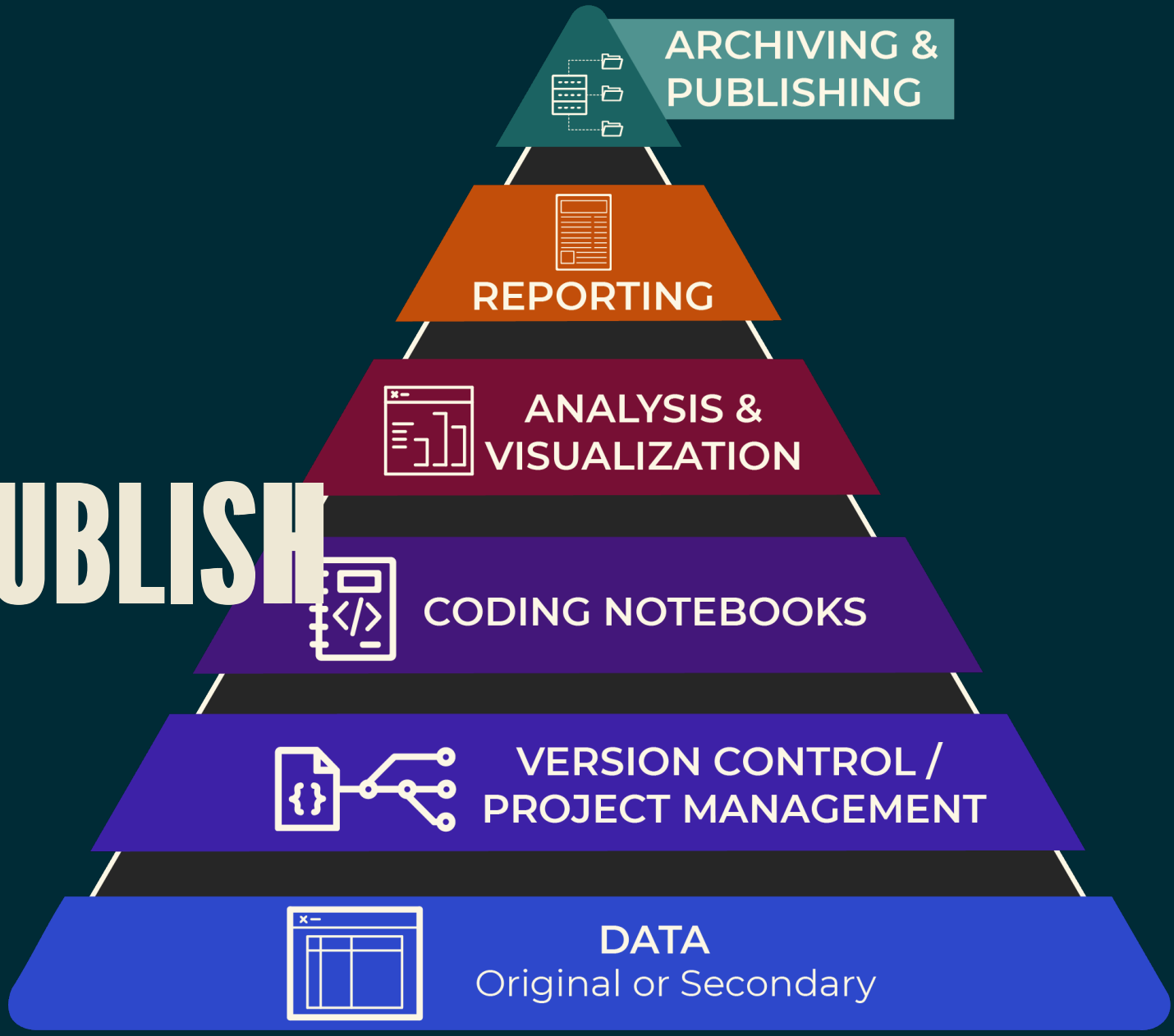


REPORTING



ort expressions are rendered at code execution

ARCHIVE & PUBLISH



TYPES OF REPOSITORIES

Archival

Posterity of milestones

Workflow

Versions / evolution of project

}

HOW

- Generate report expressions from code
- Combine GitHub releases with Zenodo to archive your milestones and share the interactive computation in a binder Hub
- **Zenodo**: general, open repository to deposit research papers, data sets, code, reports and related artifacts and connect to a citable DOI.
- **Binder**: package and share reproducible computational environments
  - mybinder.org (public BinderHub portal)

STEPS

1. Make a GitHub Release at project milestone(s)
2. Connect GitHub to Zenodo
    1. Mint a DOI to a GitHub Release (persistent identifier: citation; milestones)
    2. With DOI, link to ORCID
3. Create a publicly launchable, fully functional computation container of your work

END TO END

1. Project with version control
    - project folder with TIER organization
    - data (raw)
    - version control (git)
2. Coding notebook
    - data cleaning
    - natural language explanations
    - analysis and modeling
    - visualization
    - generate report expressions from code
3. Publish
    - workflow archived ar
    - Milestones linked to
    - **Informal**: web, file sh
        - Whitepapers, slid
    - **Formal**: vis-a-vis pee

SHARING YOUR WORKSPACE

Your computation workspace (i.e. your laptop, desktop, cloud)

Give someone else your laptop so they can play around with your projects

- the code, the data, the settings and configurations?
- Good idea?

Now you can share a copy of your computational environment

BINDER HUB

- Easiest: mybinder.org open and public
  - `quarto use binder`
- Security demands may push you to use singularity

CONTAINER EXAMPLES

- https://github.com/libjohn/workshop_rfun_iterate?tab=readme-ov-file#readme
- https://github.com/libjohn/workshop_webscraping?tab=readme-ov-file#readme

REPEAT FOR THE PDF

END TO END - STEPS

1. Project with version control
    - project folder with TIER organization
    - data (raw)
    - version control (git)
2. Coding notebook
    - data cleaning
    - natural language explanations
    - analysis and modeling
    - visualization
    - generate report expressions from code

END TO END - STEPS (CONTINUED)

3. Publish

- workflow archived and collaboration enabled via **Git**; shared through **GitHub** / **GitLab** etc.
- Milestones linked to GitHub *releases*; DOIs minted; Posterity archiving at archival repositories (e.g. Zenodo)
- **Informal**: web, file sharing, etc.
  - Whitepapers, slides, dashboards, etc.
- **Formal**: vis-a-vis peer reviewed journal arcticles