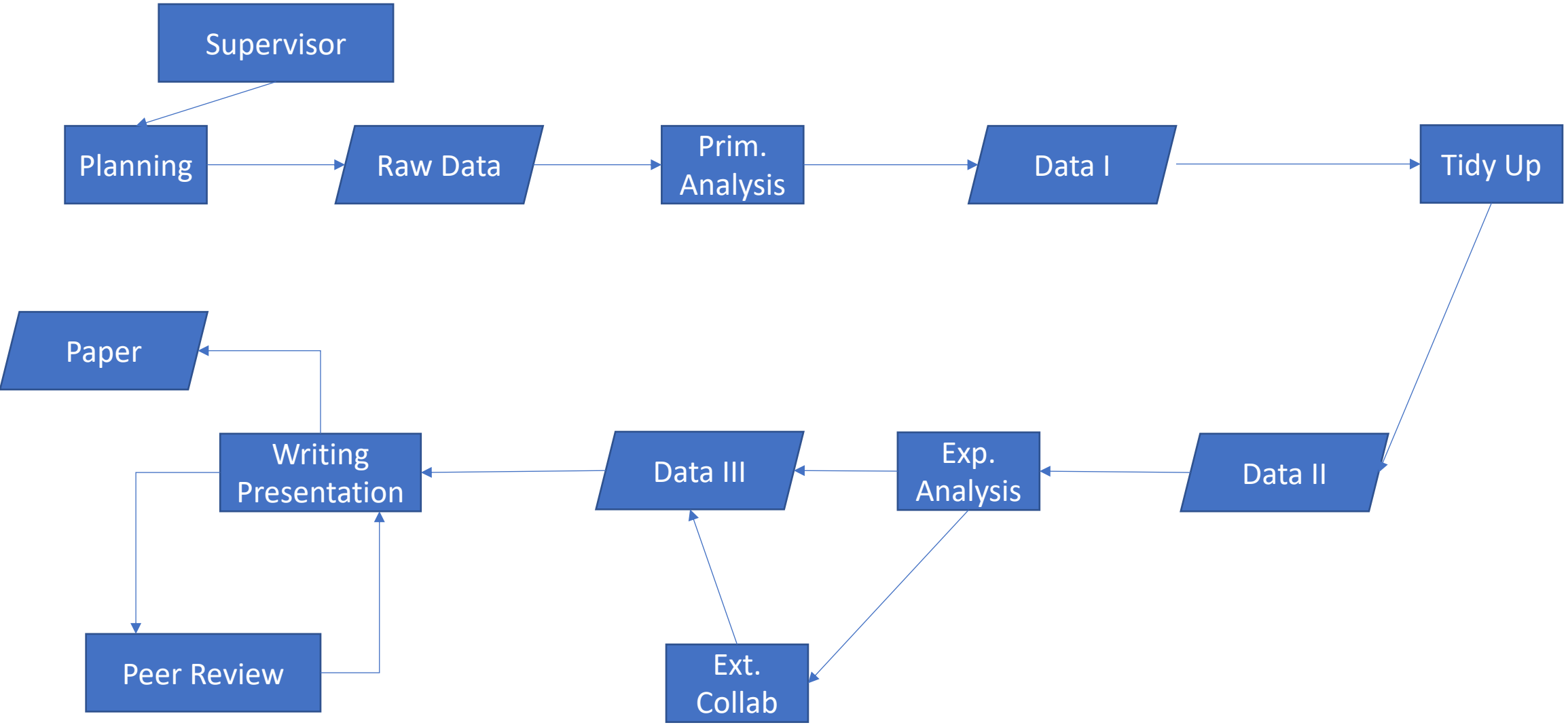
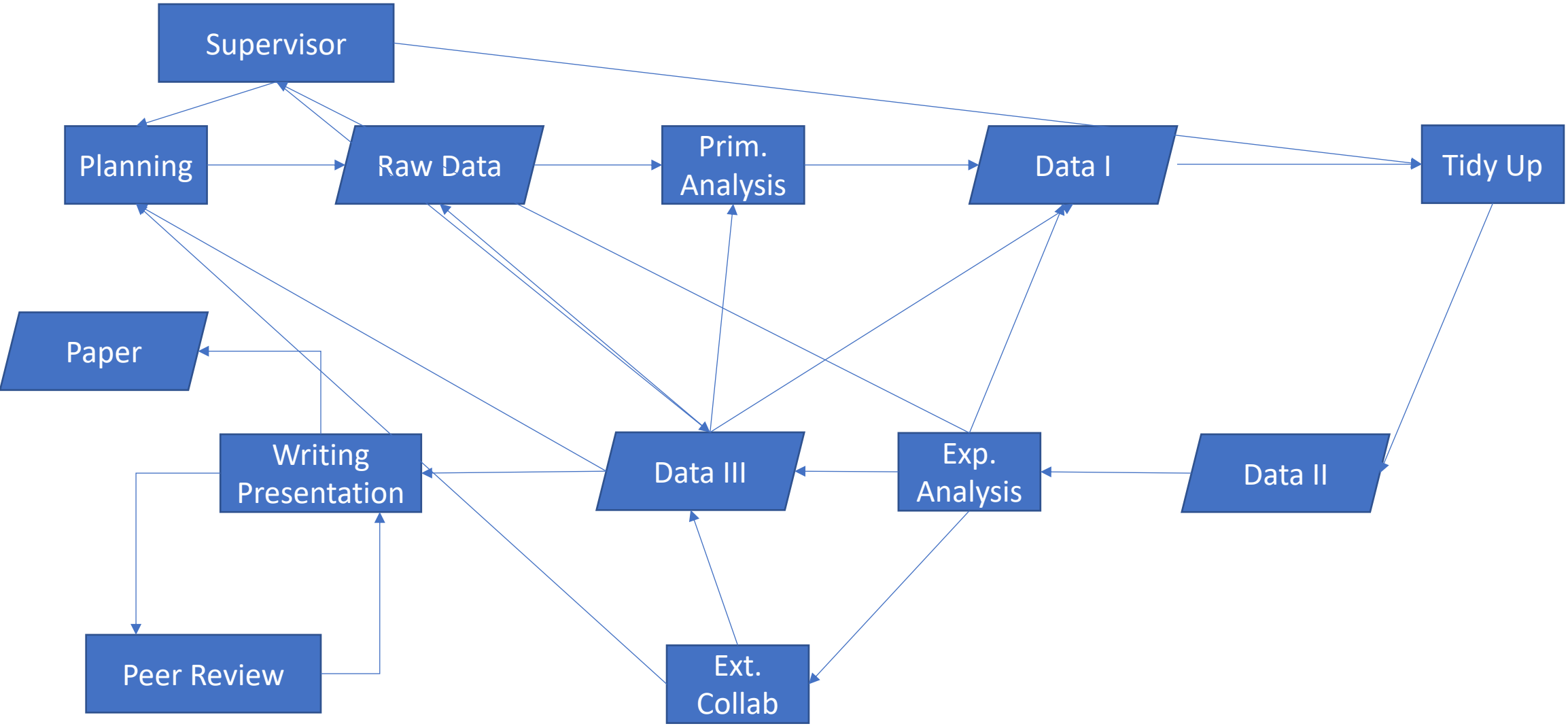
Good Enough Practices In Scientific Computing (And Data Management)

https://doi.org/10.1371/journal.pcbi.1005510

DOI


Supervisor
Planning
Raw Data
Prim. Analysis
Data I
Tidy Up
Paper
Writing Presentation
Data III
Exp. Analysis
Data II
Peer Review
Ext. Collab


Supervisor
Planning
Raw Data
Prim. Analysis
Data I
Tidy Up
Paper
Writing Presentation
Data III
Exp. Analysis
Data II
Peer Review
Ext. Collab

Data Management - Goals

- Data management
  - Data
  - Ideas
  - Decisions - Iterations are part of the scientific process. It’s important to remember why certain decisions were taken
- Make the future you happy ☺

Planning - Guidelines

- Define
  - Goals
  - Milestones
  - Timeline
  - Collaborators
- Be aware
  - Potential bottlenecks (People can be bottlenecks too)
  - Be realistic. You will not get it right on the first try

Data - Guidelines

- Raw Data:
  - Never change raw data (You also don’t change measurements?)
  - Back-up raw data
  - Make it findable (ENA/Zenodo after publication)
  - Make it usable → Describe the data
- Processed Data:
  - Don’t change output of machine generated data (label, document if you have to)
  - Back-up “expensive” data (expensive = time & money)
  - Document all steps taken to get from raw data to here
  - Processed Data will change → Version your data

Data Organisation (High level)

- A project is a self-contained entity defined by goals. Try not to mix projects
- Organise your projects with a standard template →
  - Name
  - Subfolders (Separate Code from Data)
  - Freezes
  - Readme/Changelog → Keep it up to date
  - Most important → Be consistent

Good or bad example?

- Project → project_1
  - Subfolders
    - Data1
    - Data2
  - Data files
    - Metadata.final.txt
    - Metadata.final.final.txt
    - Metadata.final.final.xlxs
    - results.v1-8.2020
    - results.v2-7.2020
    - myCode.R
    - myCode2.R

Good or bad example

- Project → **551-1119-00L_FALL2021_TEA**
  - Subfolders
    - **code** → Commands/Code/Pipeline
    - **data** → Productive data generated in this project, including raw data
    - **scratch** → Non productive data, temp files, sandbox
  - Files:
    - **Readme**
    - **Progress**
    - **Description**

Code/Software

- Project-Level
  - Use language specific guidelines for code/structure
  - Get familiar with git → Versioning
  - (Test Driven Design)
  - (Automated testing)
- Code-Level
  - Comment
  - If the comment is complicated so is the code - simple is always better
  - Semantic naming
  - Don’t repeat yourself

Collaborations - Dependencies

- Make sure that everyone is on the same page
- Goals first, details later
- Delegate duties, set deadlines → People are busy.
- React to questions/requests ASAP. --> Sometimes you cant afford to wait
- Share whatever you can share
  - Versioned Code → Git
  - Versioned Data → Zenodo

Summary

- There will be iterations/Plans will change
- Be conservative with time estimates (Expectation Management)
- Set a standard for data hygiene and follow it
  - E.g. documentation day after reaching milestone
  - Folder structure
  - Naming
- Version/DOI Freezes
  - https://zenodo.org/
  - https://github.com/

Todo

- Group by project
- Decide on
  - Name
  - Structure
  - Goals
  - …