# INF 502 – SOFTWARE DEVELOPMENT METHODOLOGIES

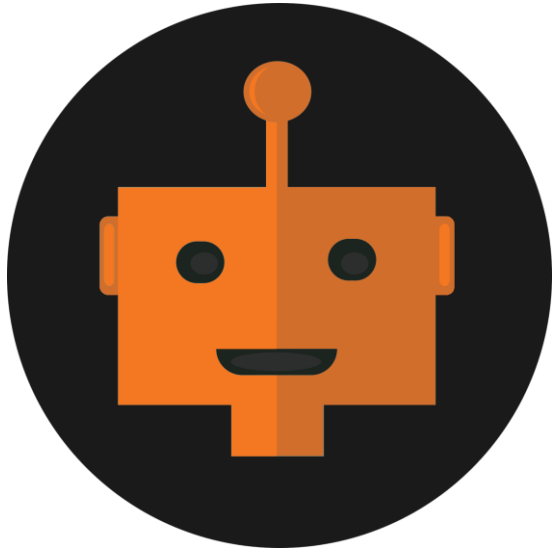
Week 1

NAU NORTHERN ARIZONA UNIVERSITY

# Course instructor

- Dr. Ana Paula Chaves
  - Ph.D. in Informatics and Computing
  - Ms. in Computer Science
- Assistant Teaching Professor, NAU
- Contact:
  - Ana.Chaves@nau.edu
  - MS Teams: link on BBLearn
- Office hours:
  - Available on GitHub

# About me…



# Communication

**MS Teams channels**

Quick questions

Discussions

**MS Teams private message**

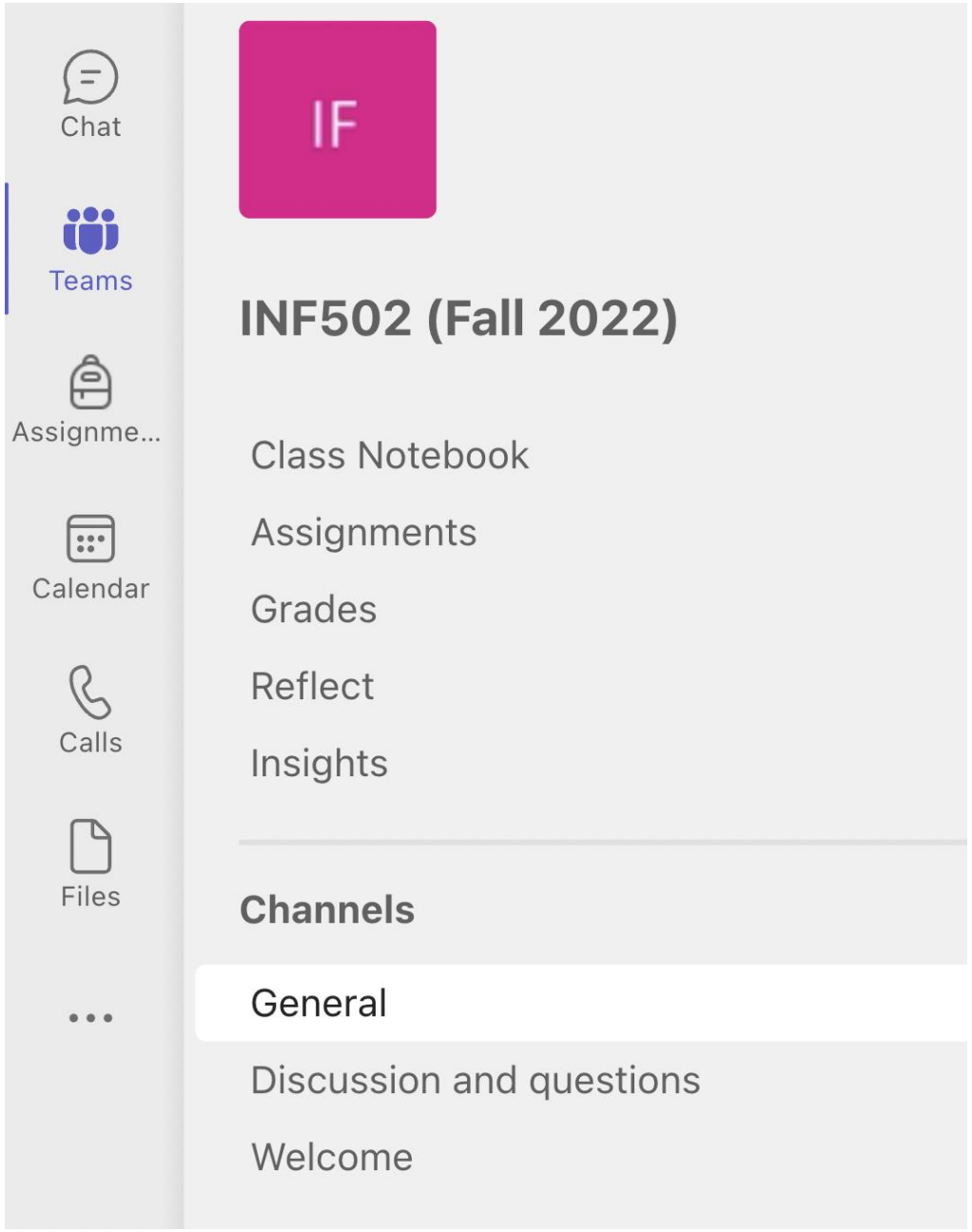
Individual interests

**Email**

Ana.Chaves@nau.edu

**Office hours**

In-person, SICCS building, rm 216



# The course...

## Course page

- https://github.com/chavesana/INF502-Fall22

## What?

- Git/GitHub
- Python
  - With some extras
- Software engineering (Agile)

# About you…

What is your background (BS, MS, etc.)

Knowledge in Programming (if any)

- where did you learn and how much do you know

What is your research topic (which program)

Your expectations about this course

# Syllabus Time

NAU NORTHERN ARIZONA UNIVERSITY

# Be ready for what's next…

Create a GitHub account: www.github.com

NAU NORTHERN ARIZONA UNIVERSITY

# INF 502 – SOFTWARE DEVELOPMENT METHODOLOGIES

Introduction to Programming Languages and source control

NAU NORTHERN ARIZONA UNIVERSITY

# Programming languages

Enable constructing representations of a computational process; well-defined algorithms processing information

Mapping to machine instructions

Syntax and associated semantics

Fundamentally just like human languages and form of expression

Non-functional properties become critical

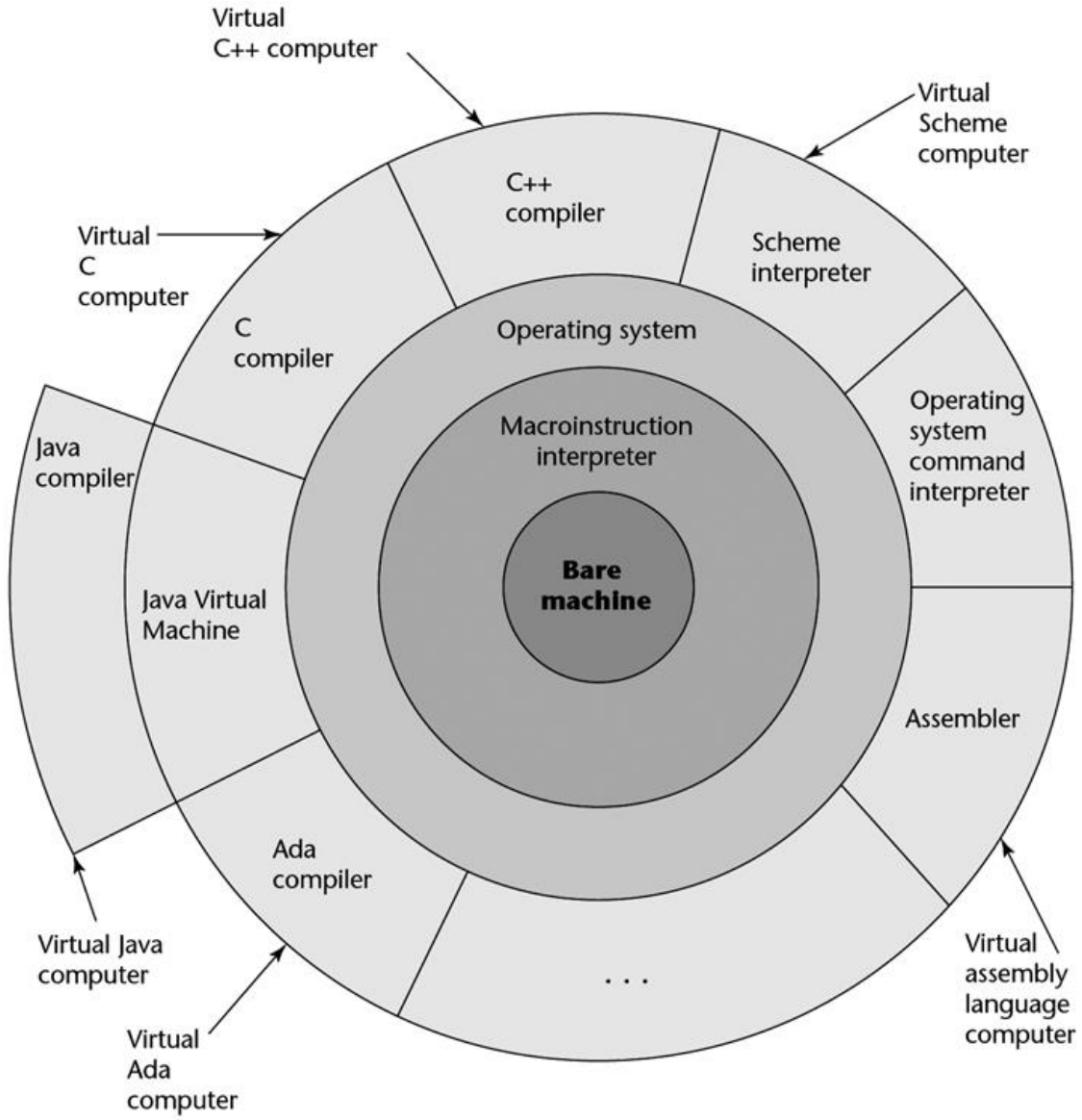
# Language Implementations

Layered architectures

- Mappings from high-lev
  low-level instructions

**Figure 1.2**

Layered interface of virtual computers, provided by a typical computer system



# Compiler-based implementations

Mapping

- High-level syntax to machine code
  - Plus linking of external resources

(Some) Advantages:

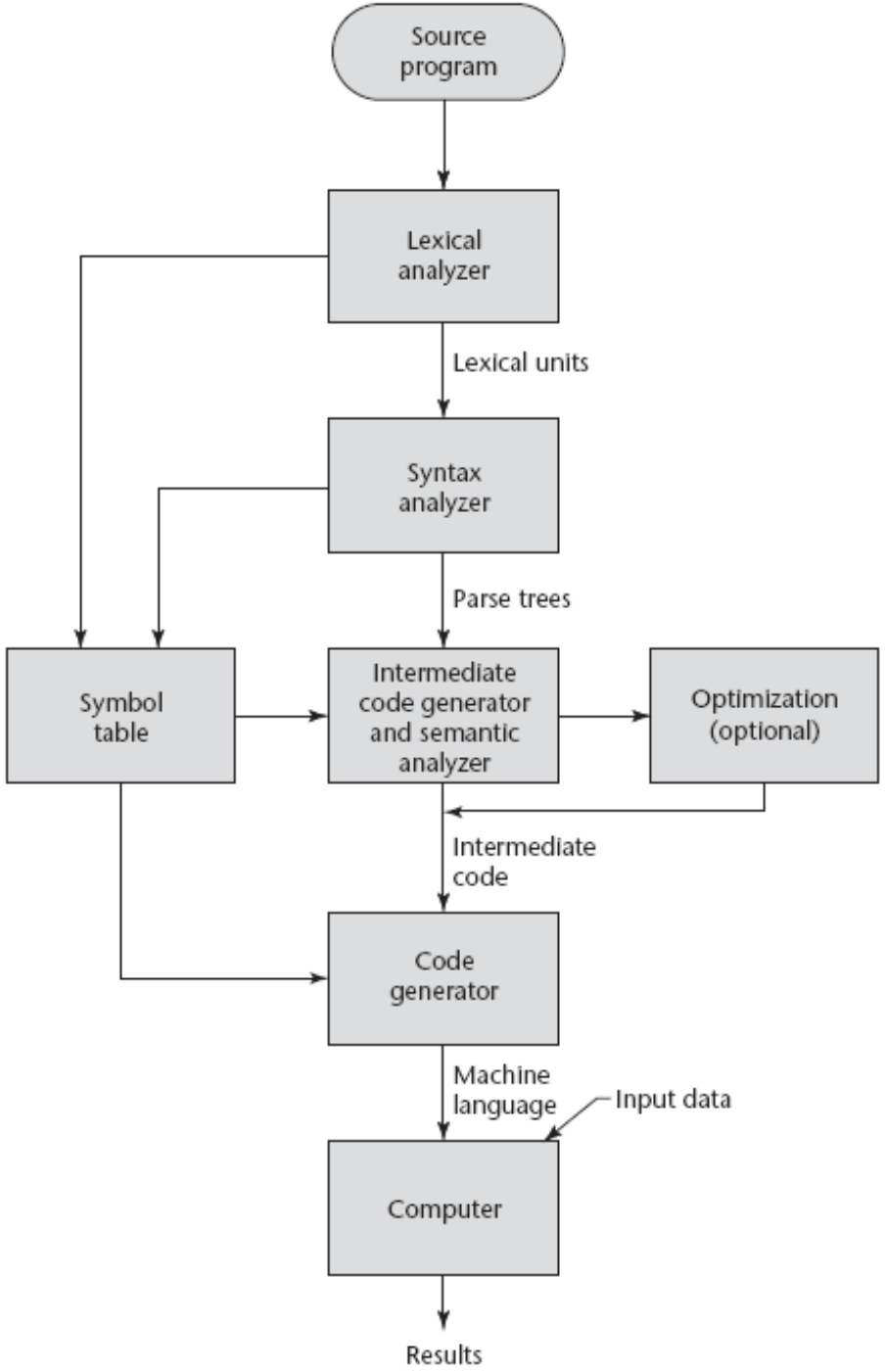
- (Usually) faster execution due to optimizations
  - Both algorithmic and machine-specific

(Some) Disadvantages:

- Compiled code coupled to specific hardware architecture
- Long iterative cycle
- Requires complete program

**Figure 1.3**

The compilation process

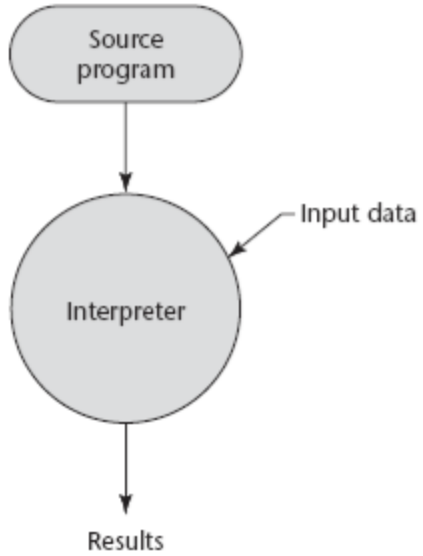


# Interpreter-based implementations

- Mapping
  - High-level syntax executed by interpreter
  - Interpreter “wraps” around machine and maps to machine code
- (Some) Advantages:
  - Higher accessibility
    - Ease of experimentation
  - Portable from machine to machine
    - As long as an interpreter exists for each
  - Dynamic code generation
- (Some) Disadvantages:
  - (Usually) slower due to interpreter layer

**Figure 1.4**

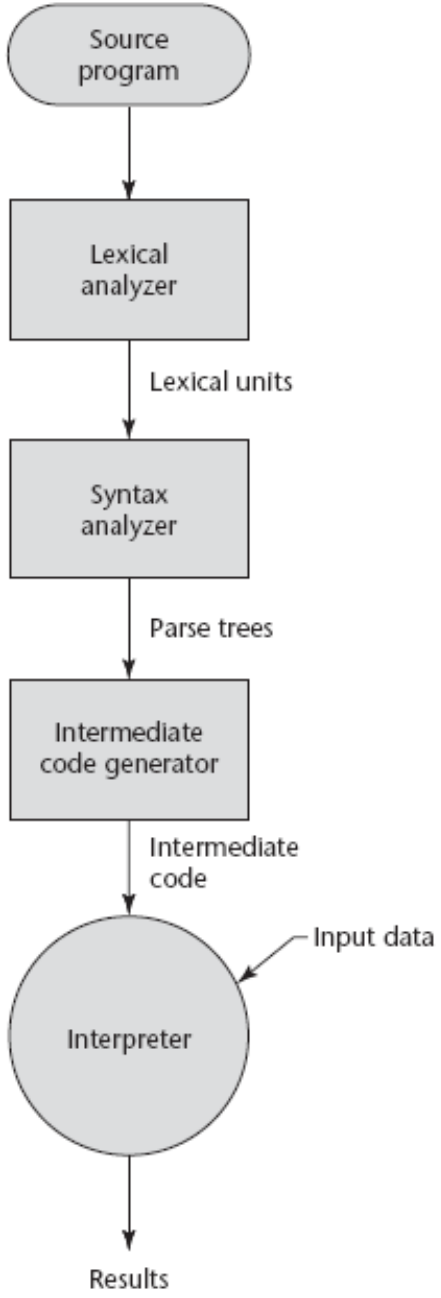
Pure Interpretation



# Hybrid implementations

- Mapping
  - High-level syntax to interpreter instructions (intermedi representation)
    - Or purely interpreted
  - Interpreter still “wraps” around machine and maps inte representation to machine code
- (Some) Advantages:
  - Improved performance (over fully interpreted options)
    - Enabling compiler-type optimizations
  - Higher accessibility
    - Intermediate representation portable from machir machine
- (Some) Disadvantages:
  - Longer iterative cycle than fully interpreted options
  - (Usually) still slower due to interpreter layer

**Figure 1.5**
Hybrid implementation system



# The source control

# Code Management/Versioning

- Team development
  - Code sharing and versioning…


Dropbox


# Code Management/Versioning




CVS


...

# We will focus on:

# Who offers this service


Atlassian Bitbucket

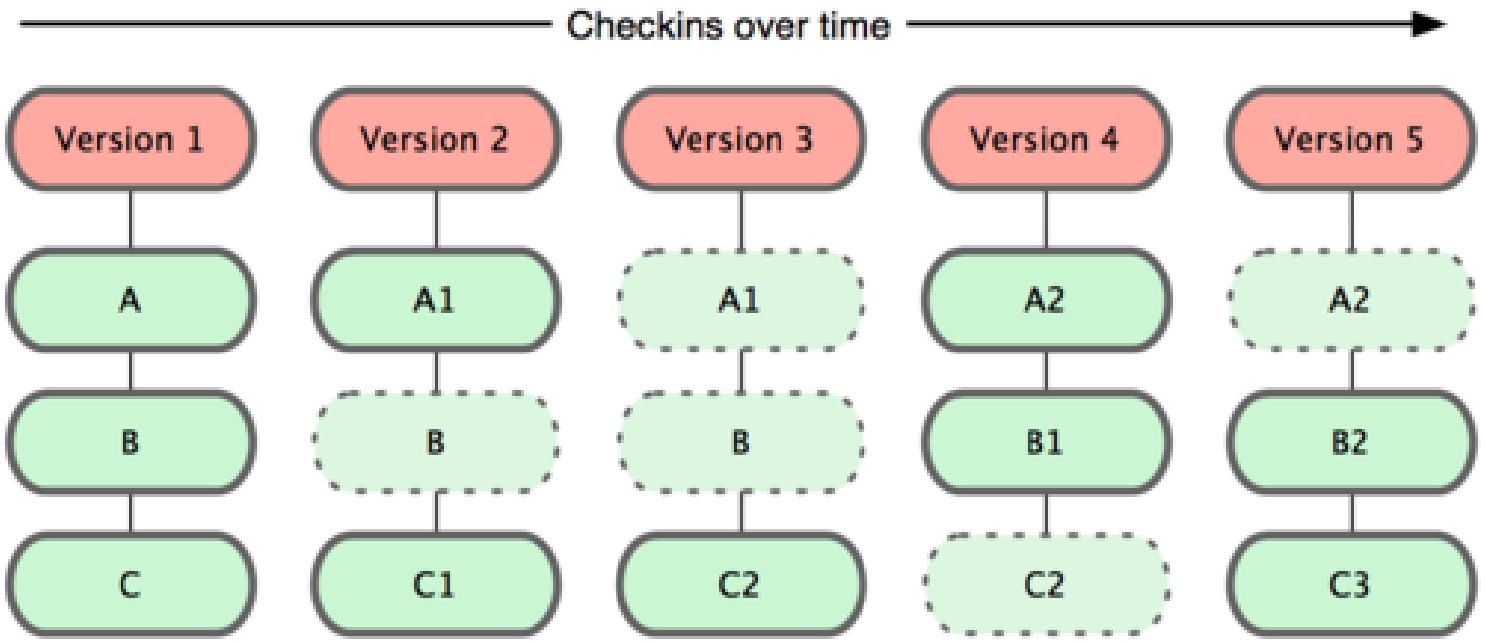

Your machine! (?)

beanstalk

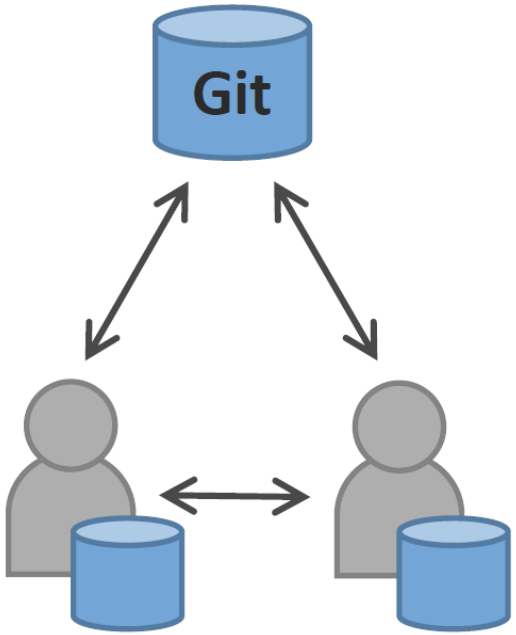

SOURCEFORGE®

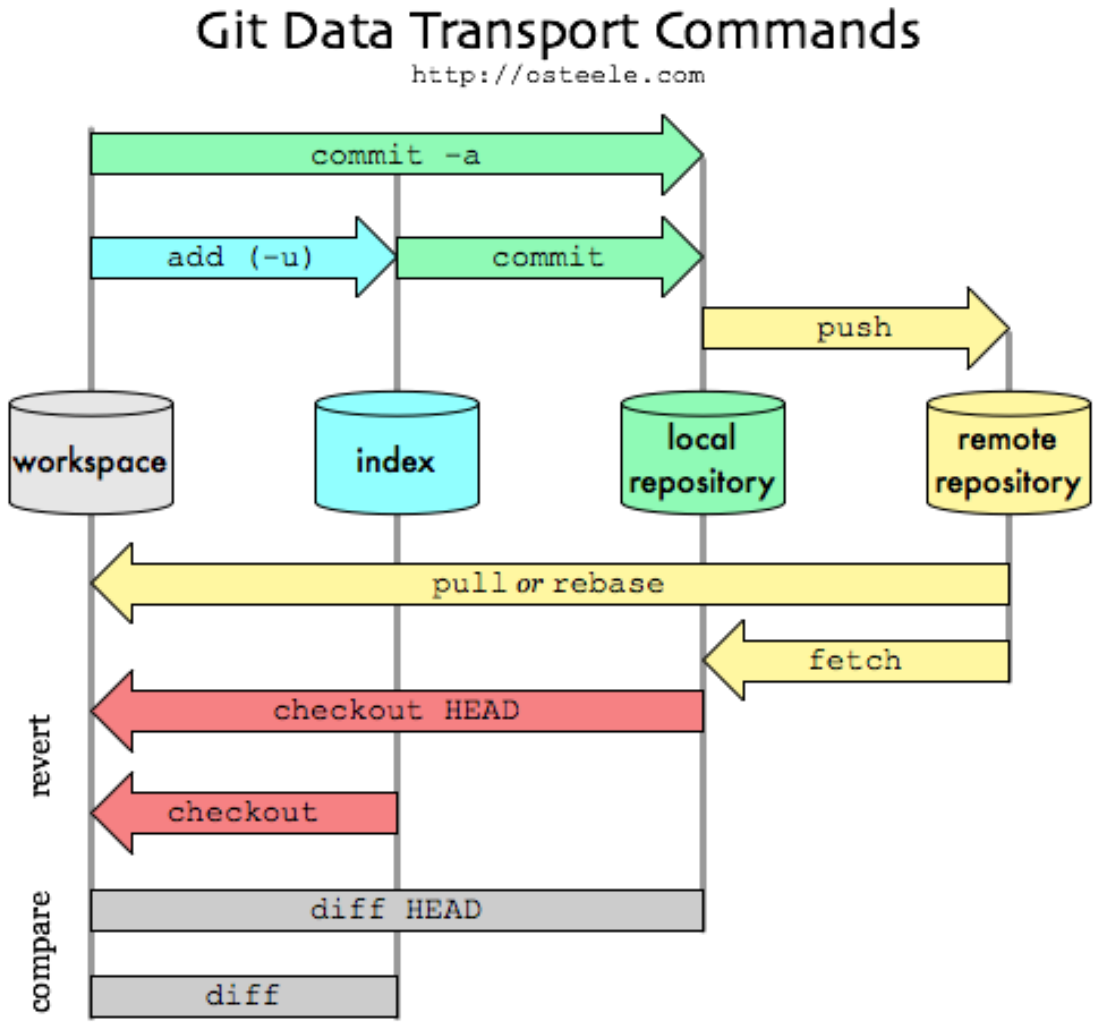
codebase

# How Git Manages Files Over Time

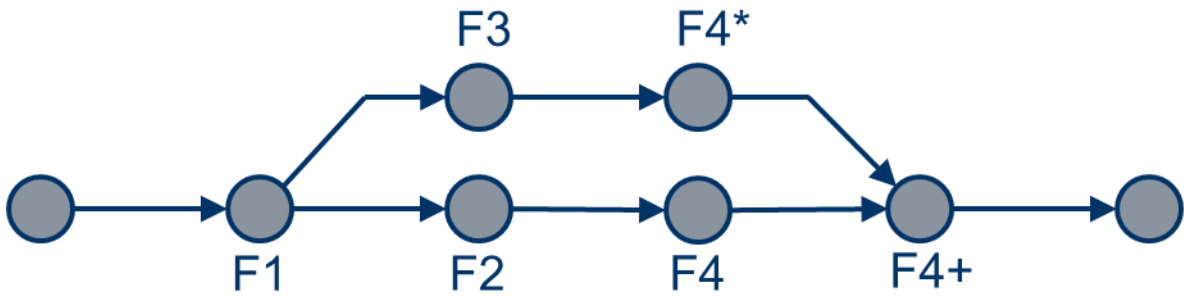


# Git - overview

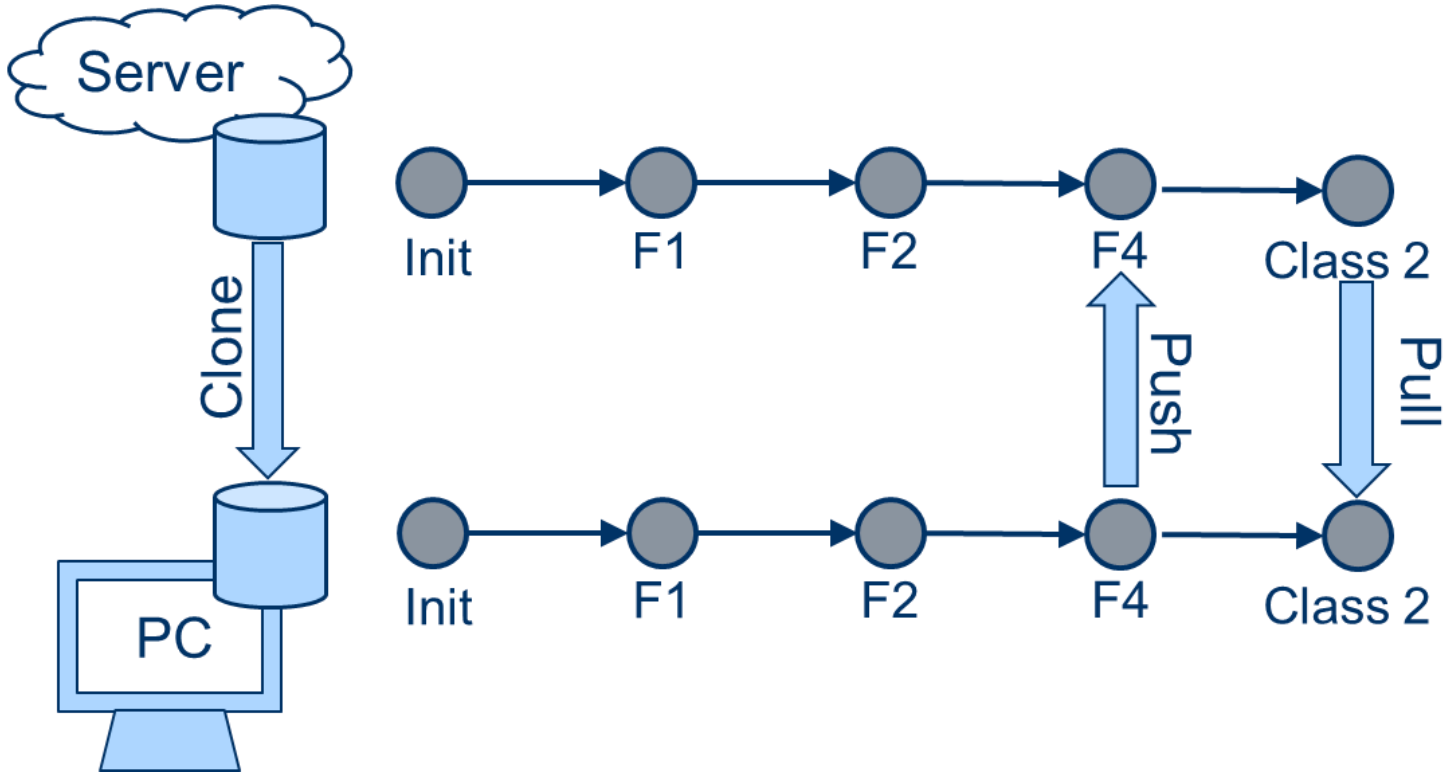


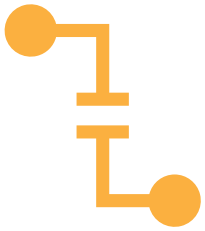
# Git Local Flow - Example




# Git Local Flow - Example



# It’s Hands On Time



## Moment 1:

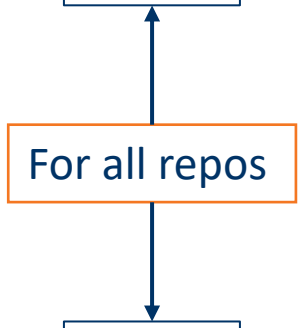
Local commands: add, commit, branch, merge, conflicts..

## Moment 2

Interaction with the remote repo: Push, pull

# Kicking off

- git config --global user.name “Ana Chaves”




- git config --global user.email “ana.chaves@nau.edu”

- Create a folder / access this folder
- git init
  - This folder is now a repo

# Hands On

1

Create a file

2

Check the status of the repo

- git status

3

Add the file to the index

- git add <filename>

4

Check the status

# Hands On

- Our first commit
  - git -a -m “Our first commit!!!”
    - -a: all files
    - -m: will include a commit message
- Check the last commits
  - git log
- Check what has been done in the last commit
  - git show

# Let’s do it again…


And this is the basic flow to put your contributions back to the repo

# Be ready for what's next…

Branching and merging

Dealing with remote repositories