COMP1531

🐶 Software Engineering

3.3 - Testing - Continuous Integration

In this lecture

Why?

- To scale multi-user software projects, we need automated ways to integrate and test code

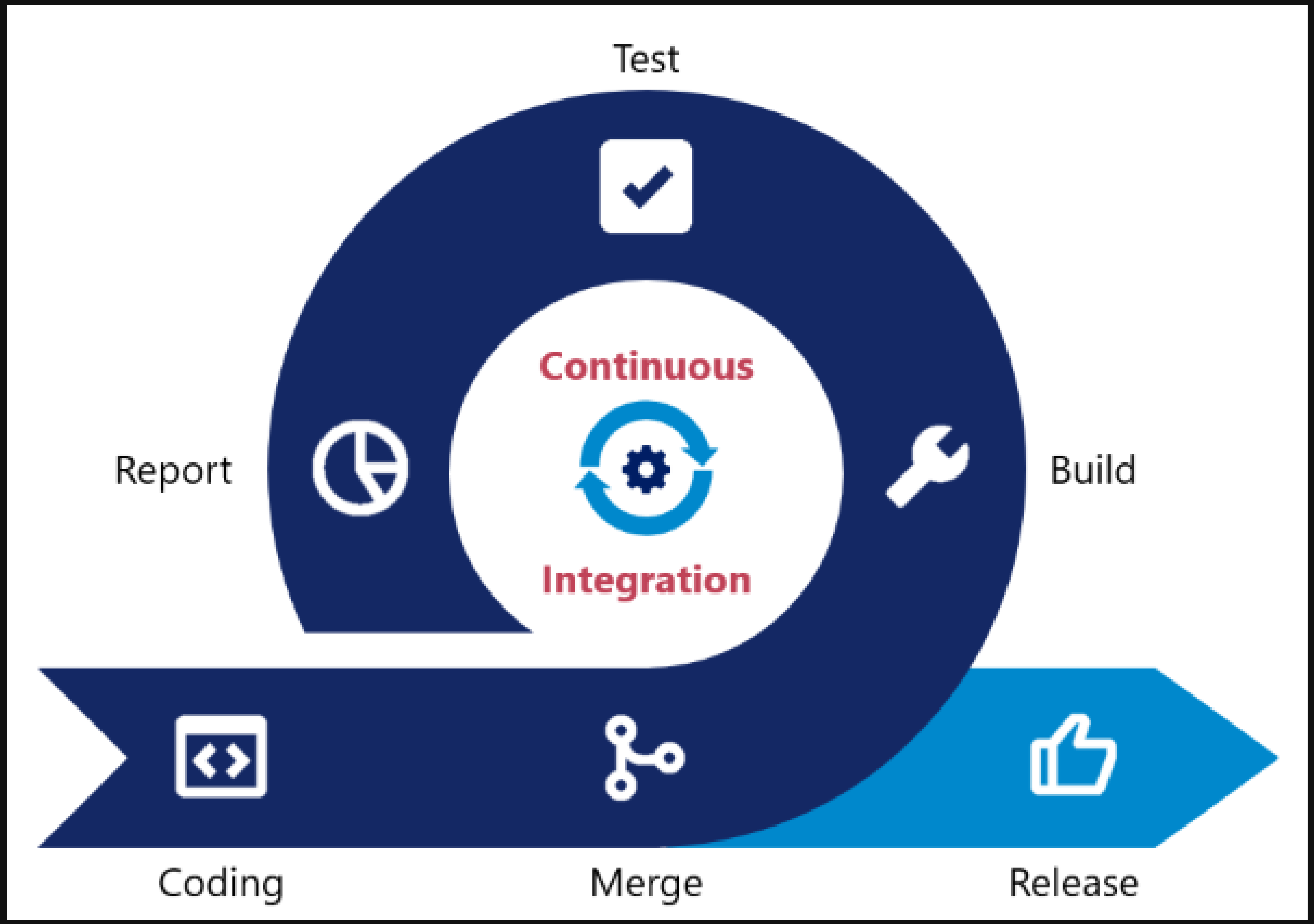
What?

- Continuous Integration
- Pipelines
- Runners

Continuous Integration

Continuous integration: Practice of automating the integration of code changes from multiple contributors into a single software project.

Continuous Integration



Continuous Integration

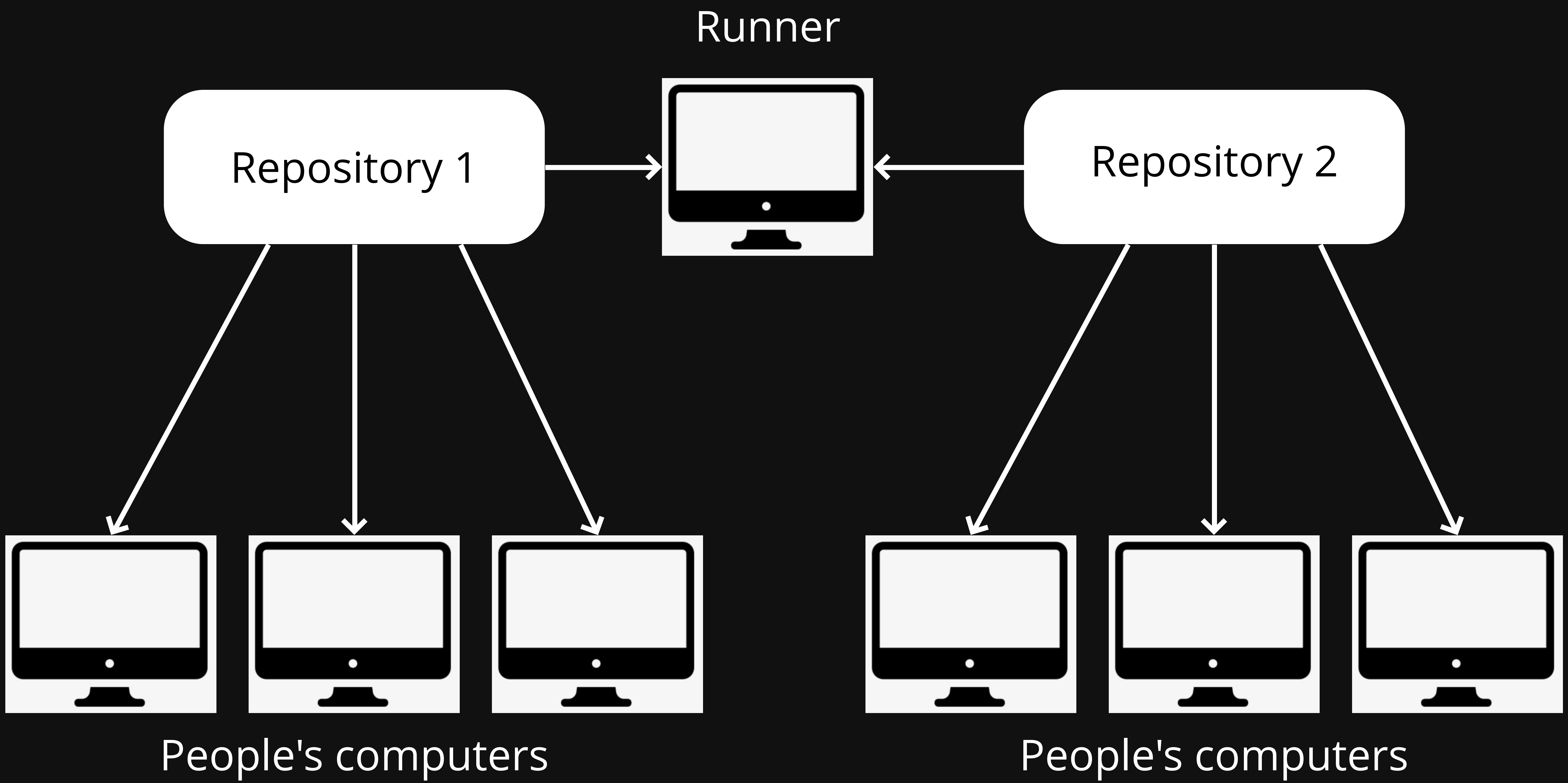
Key principles and processes:

1. Write tests:
    1. Ideally tests for each story
    2. Broad tests: unit, integration, acceptance, UI tests
2. Use code coverage checkers
3. Merge and integrate code as often as possible
4. Ensure the build always works (i.e. is "green")

How it works

- Typically tests will be run by a "runner", which is an application that works with your version control software (git) to execute the tests. This is because tests can require quite resource intensive activities
  - Gitlab: No runners built in
  - Bitbucket: Runners built in

Broad Architecture



Readings on CI

You should definitely read the following:

- https://about.gitlab.com/product/continuous-integration/
- https://www.atlassian.com/continuous-delivery/continuous-integration/how-to-get-to-continuous-integration

Continuous integration, gitlab

Gitlab, like many source control tools, has a way of doing continuous integration. An overview is here and a start guide is here.

There is quite a lot of variance and depth to this, so we will not cover it in any detail besides high level

A simple example can be found here.

Continuous integration, gitlab

In gitlab repos, you can setup your own continuous integration if:

- You connect a runner to your repository
- You setup a .gitlab-ci.yml file
    - (You don't need to understand this file, we will talk about this syntax next week)

Let's add pylint to the pipeline!

Feedback