Introductory overview of DevOps

Culture • Practices • Tooling • Examples

Federico Incardona – INAF OACT
Kevin Munari – INAF OACT
Sebastiano Spinello – INAF OACT

USC-C General Assembly, Savoia Excelsior Palace, Trieste (IT) - March 11, 2026

Agenda

- What DevOps is (and is not)
- The DevOps core principles and lifecycle
- Five fundamental practices (version control, Agile, shift left, CI, CD)
- Testing strategies, SDKs & SDEs, containers, and orchestrators basics
- Observability and incident response
- Focus on CI through two examples adopted in scientific collaborations

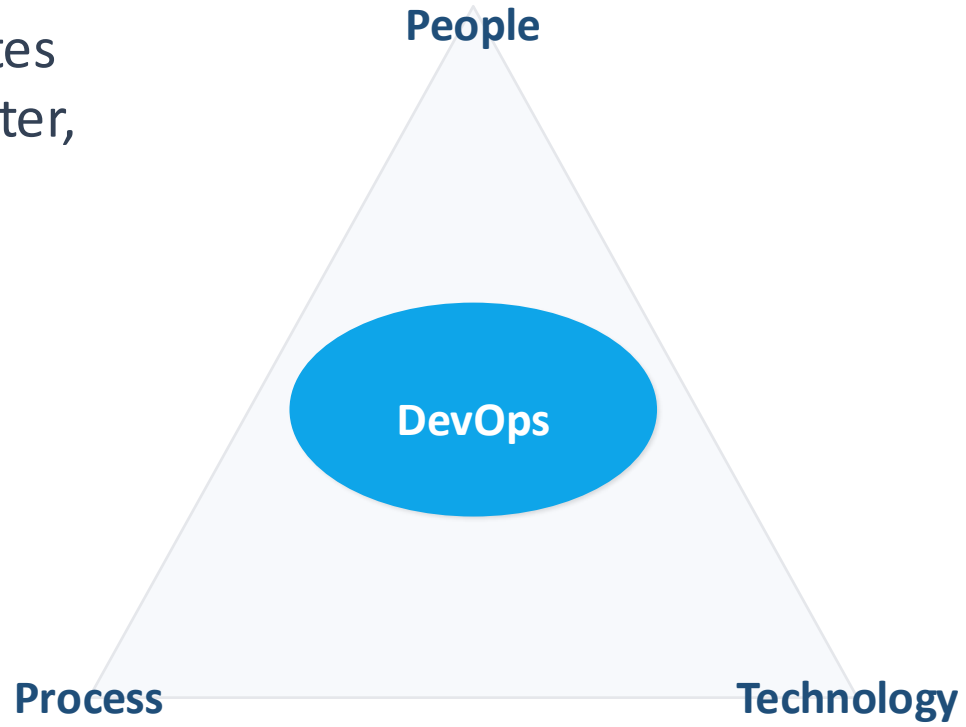
What is DevOps?

DevOps is a **software development methodology** that unites development and operations teams to deliver software faster, more securely, and more efficiently.

Not a single tool or product.

Not a team that does deployments for everyone.

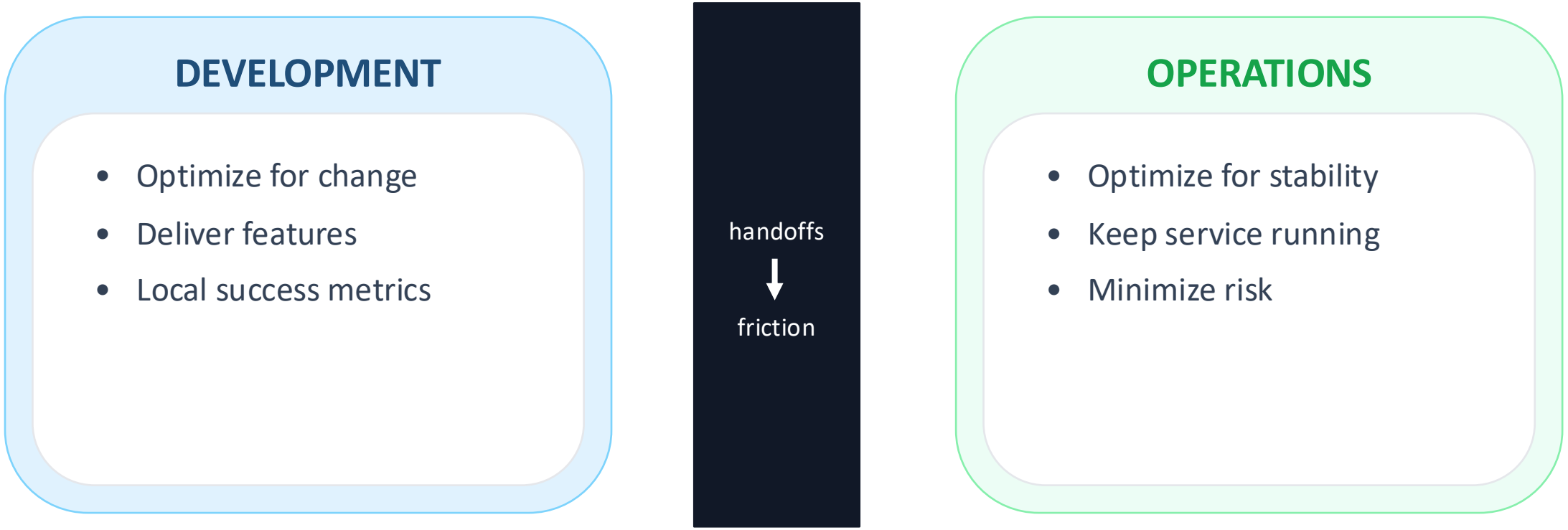
DevOps is a system of people, processes, and technology working together.



Not a tool | Not a team name | It's a culture

Why DevOps? The “wall of confusion”

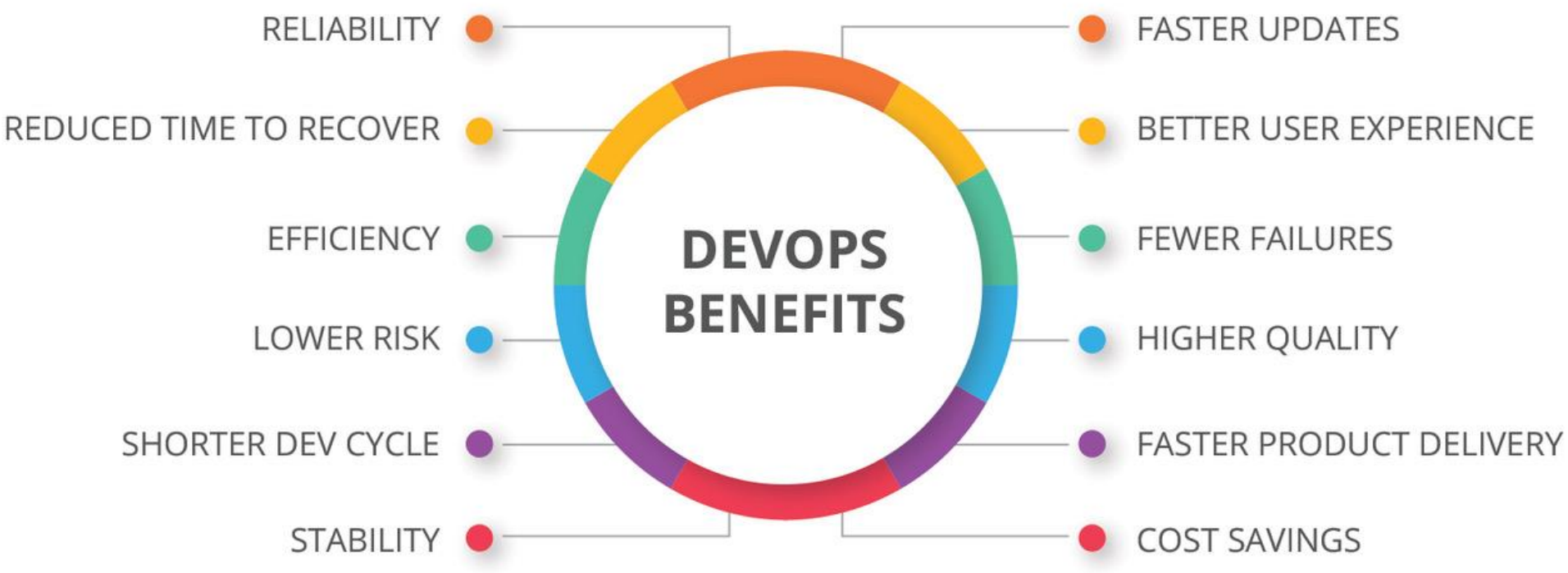
Traditional organizational structure often creates slow, risky handoffs



tickets • approvals • manual steps

DevOps culture aims for better outcomes

High-performing teams improve both speed and stability.



https://devops.com/different-organizations-different-devops-outcomes/

CALMS: a simple DevOps assessment lens

CALMS = Culture, Automation, Lean, Measurement, Sharing

C

Culture

Trust, collaboration, shared ownership

A

Automation

Repeatable tasks, deployments, and workflows

L

Lean

Reduce waste (huge batches, handoffs, waiting time)

M

Measurement

Use data, metrics, and evidence

S

Sharing

Knowledge, best practices, and failures

Use CALMS as a practical checklist.

The “Three Ways” (principles behind DevOps)

1) Flow

Make work move smoothly from idea → production

2) Feedback

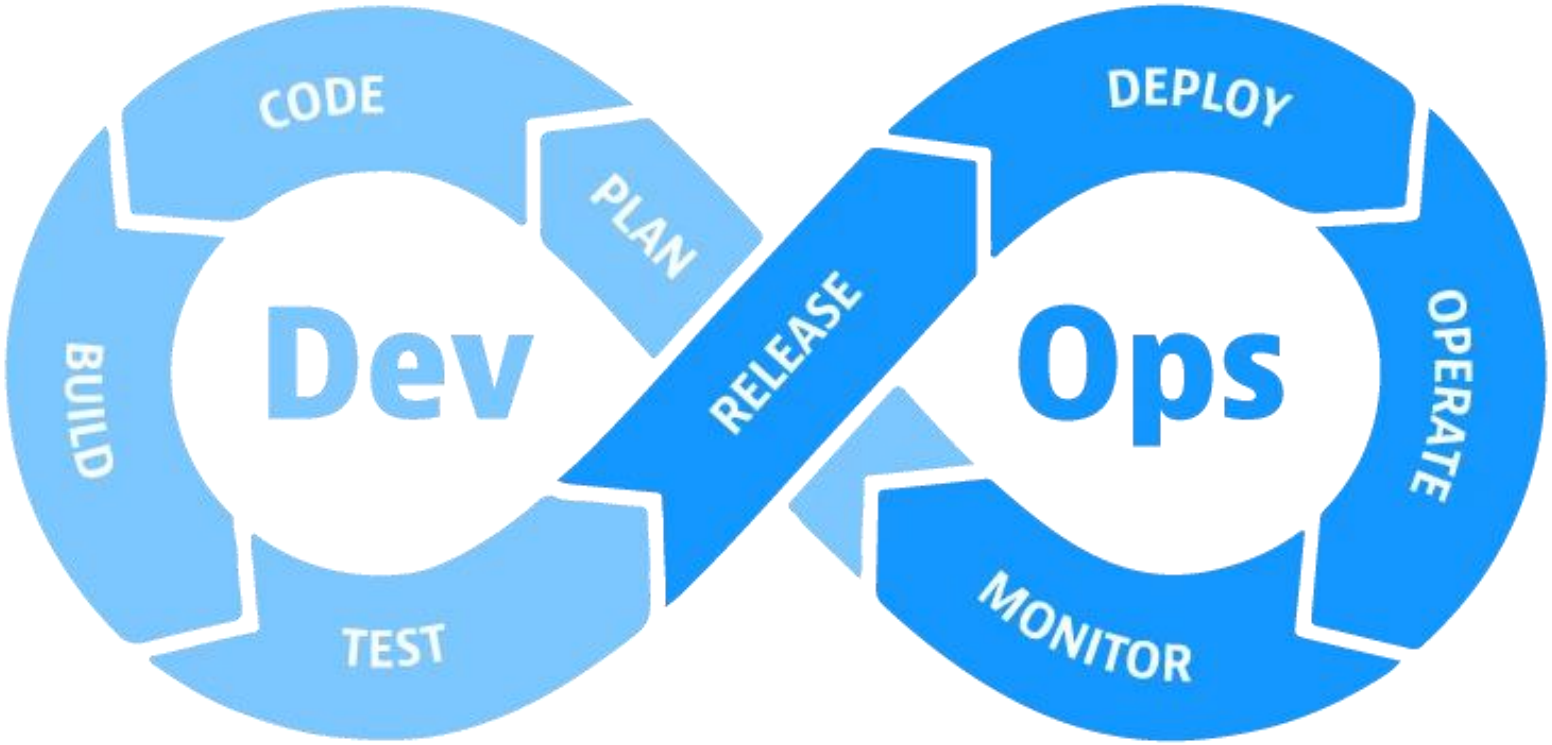
Create fast, high-quality feedback from prod → dev

3) Learning

Continuously learn (experiments, retrospection)

The DevOps lifecycle is a continuous loop

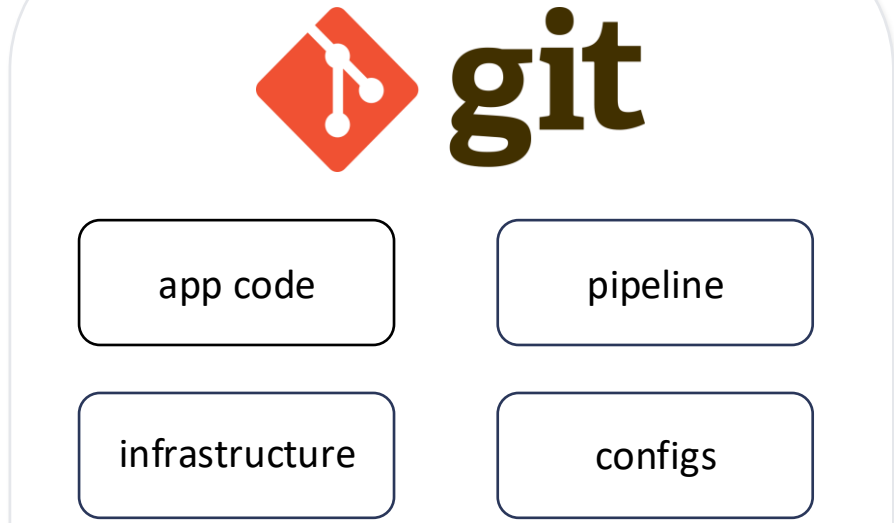
Plan → Code → Build → Test → Release → Deploy → Operate → Monitor → (back to Plan)



https://pagertree.com/learn/devops/what-is-devops

Version control

- Everything important is versioned: code, configs, infrastructure definitions, pipelines
- Small, frequent merges reduce painful integration
- Pull requests + reviews improve quality and knowledge sharing
- Tags/releases create traceability from commit → artifact → deployment



Agile

Individuals and interactions
over
processes and tools

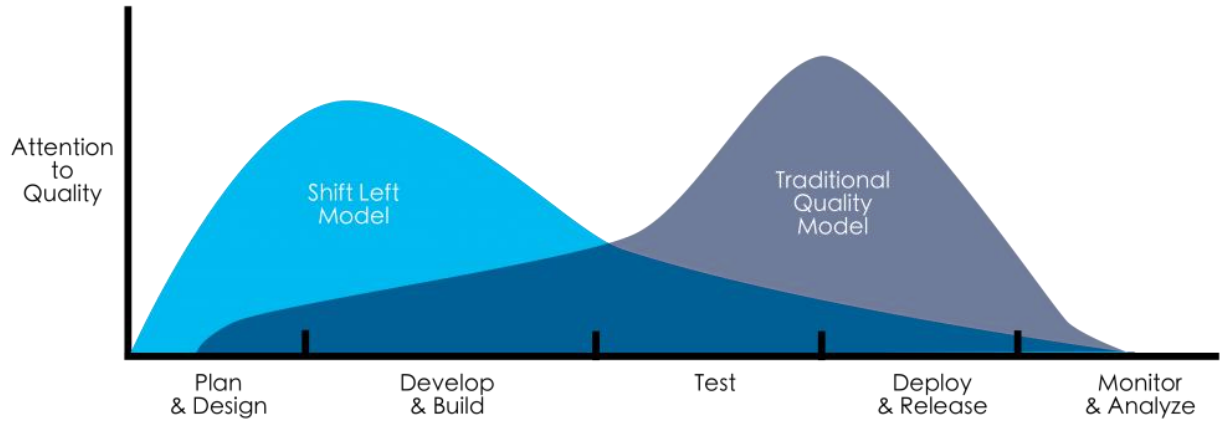
Working software
over
comprehensive documentation

Customer collaboration
over
contract negotiation

Responding to change
over
following a plan

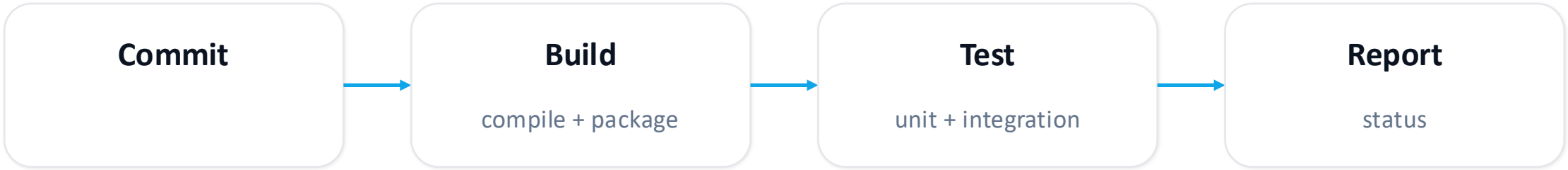
Shift left

- Run unit tests on every change
- Add integration tests where it matters
- Automate code quality checks (static analysis)
- Keep test suites fast and reliable



Continuous Integration (CI)

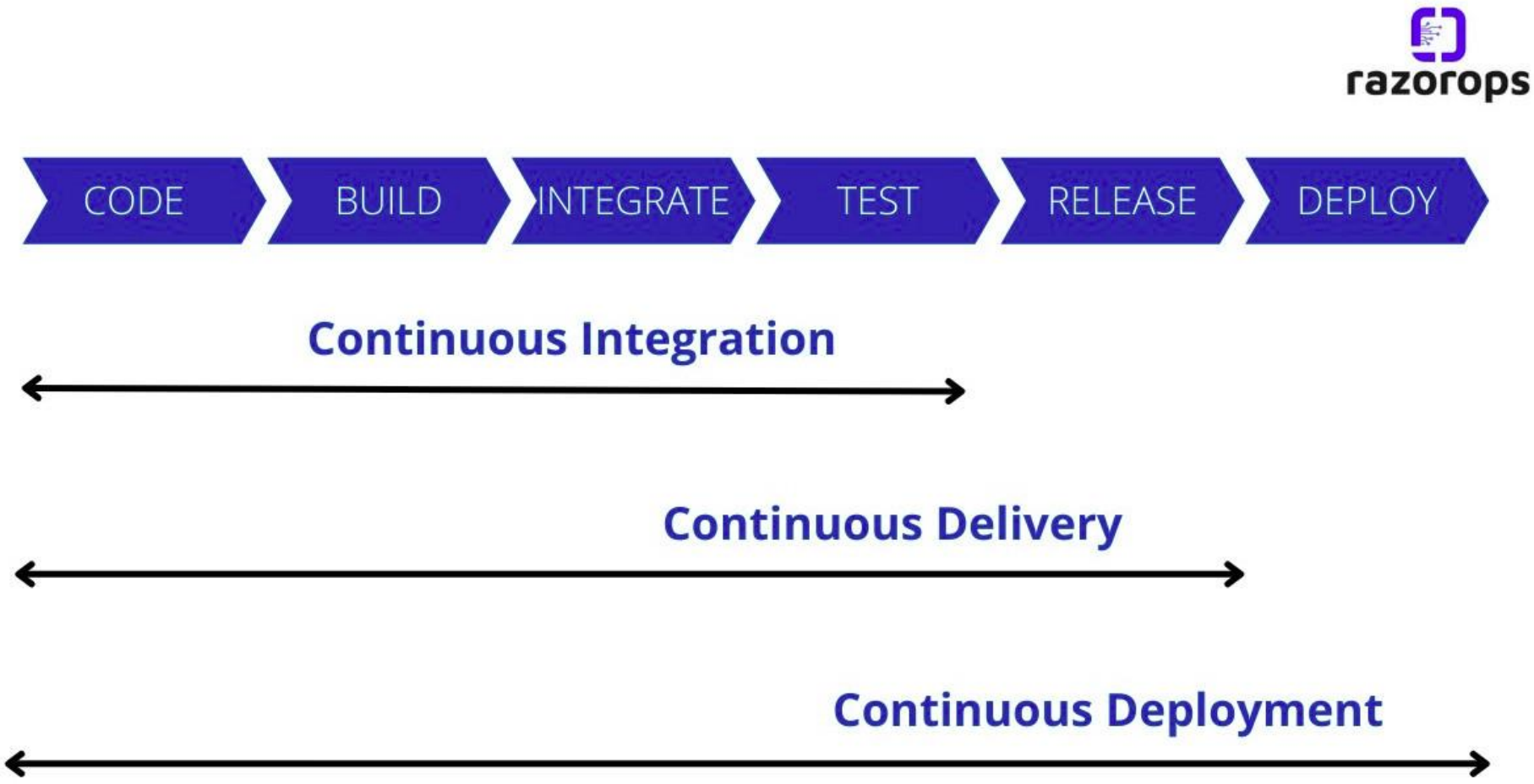
CI means integrating changes frequently and validating them with automated builds and tests.



- Goal: detect integration problems early (minutes, not days).
- Keep the pipeline fast and deterministic (same input → same output).
- Introduce quality gates.

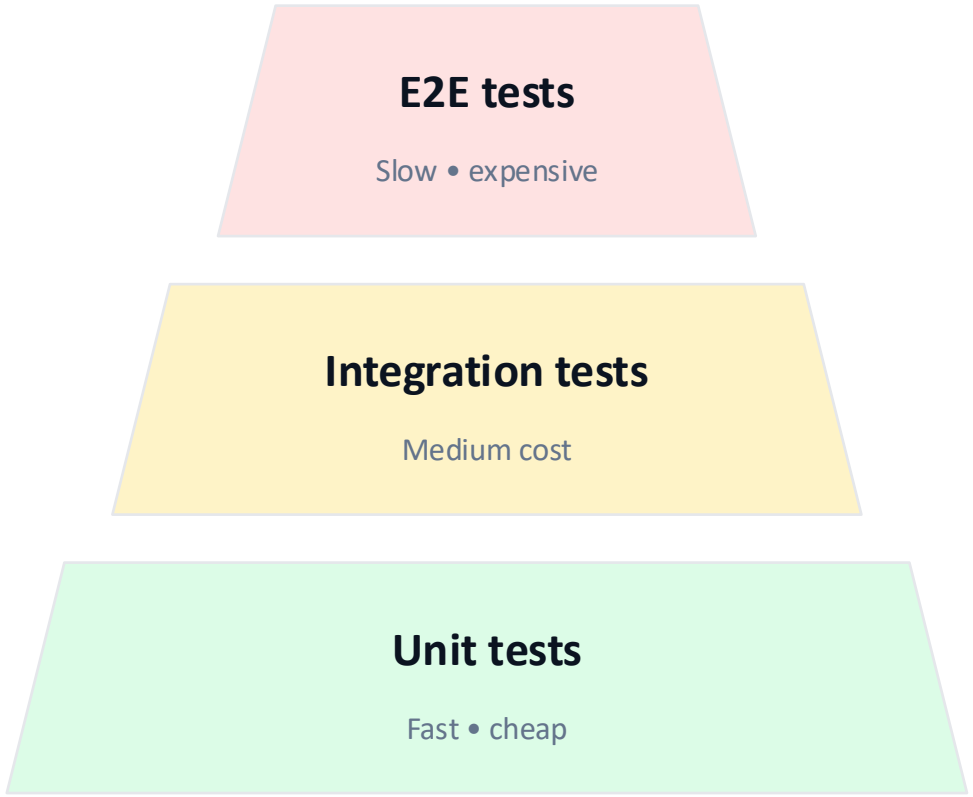
Continuous Delivery vs Continuous Deployment

Both build on CI. The difference is whether “production release” is manual or automatic.



Automated testing strategy (test pyramid)

More fast, cheap tests at lower levels; fewer slow end-to-end tests at the top.



- Very few E2E tests.
- Fewer integration tests.
- Lots of fast unit tests.

SDKs and SDEs

Software Development Kit

- APIs
- Libraries
- Interface files
- Development tools
- Documentation
- Code samples / templates
- Configuration files

Standardized Development Environment

- Operating system & libraries
- Programming language runtime & compiler
- Dependency / package managment
- SDK
- Build system & precompiled binaries
- Version control system
- IDE

Containers: consistent runtime environments

Containers standardize how software is built and run across environments.

Virtual Machines

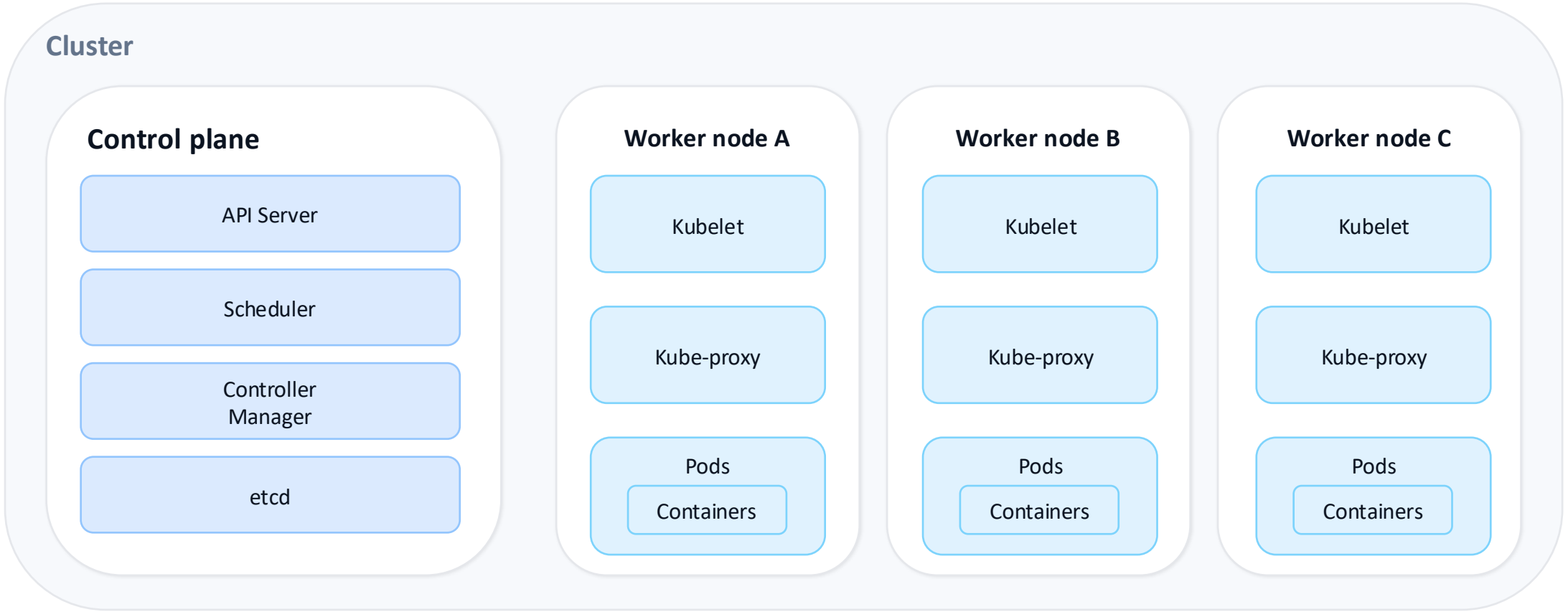
- App
- Libraries
- Guest OS
- Hypervisor
- Host OS

Containers

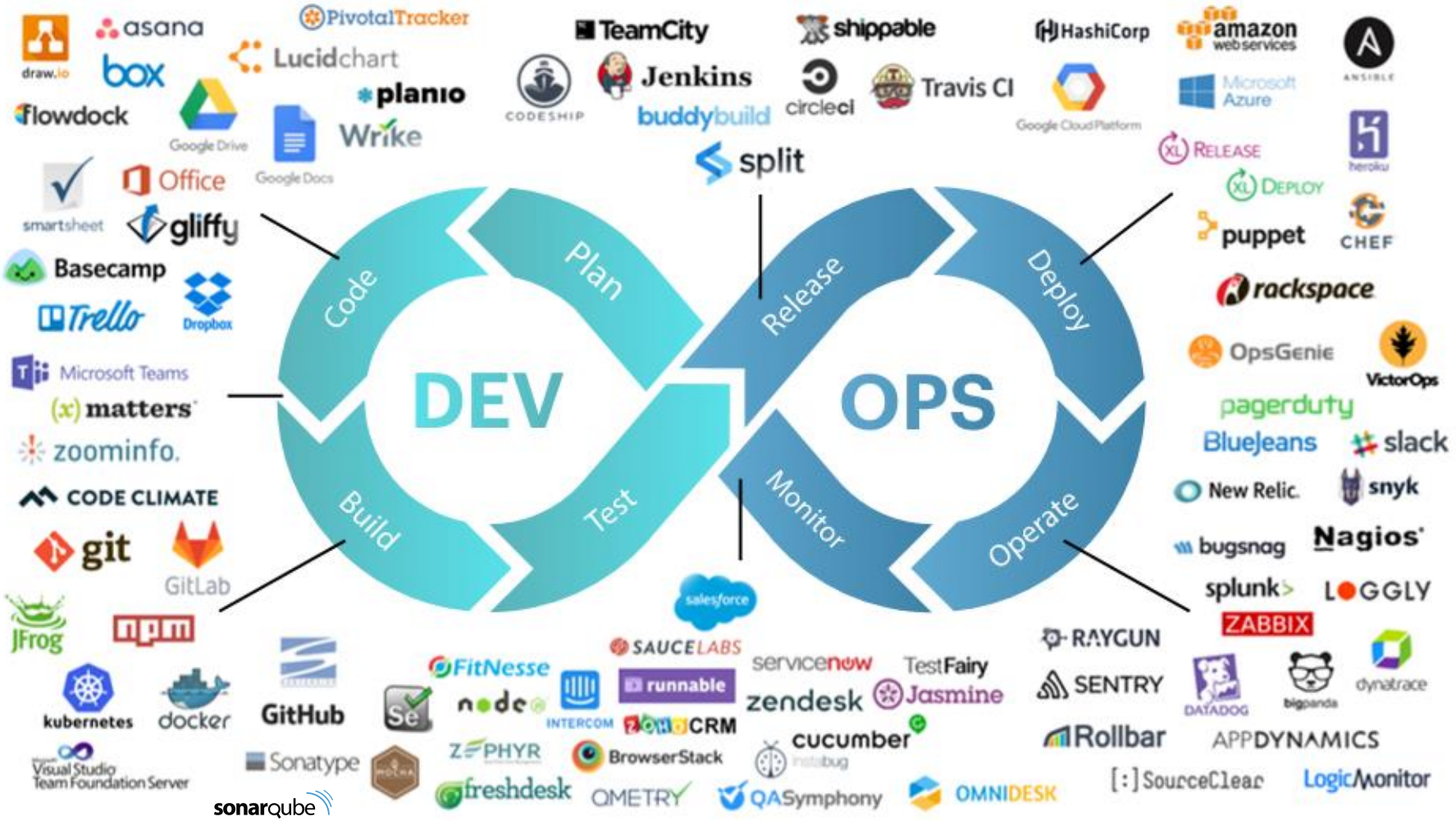
- App
- Libraries
- Container runtime
- Host OS

Kubernetes: orchestrating containers

Kubernetes schedules containers across a cluster and keeps the desired state running.

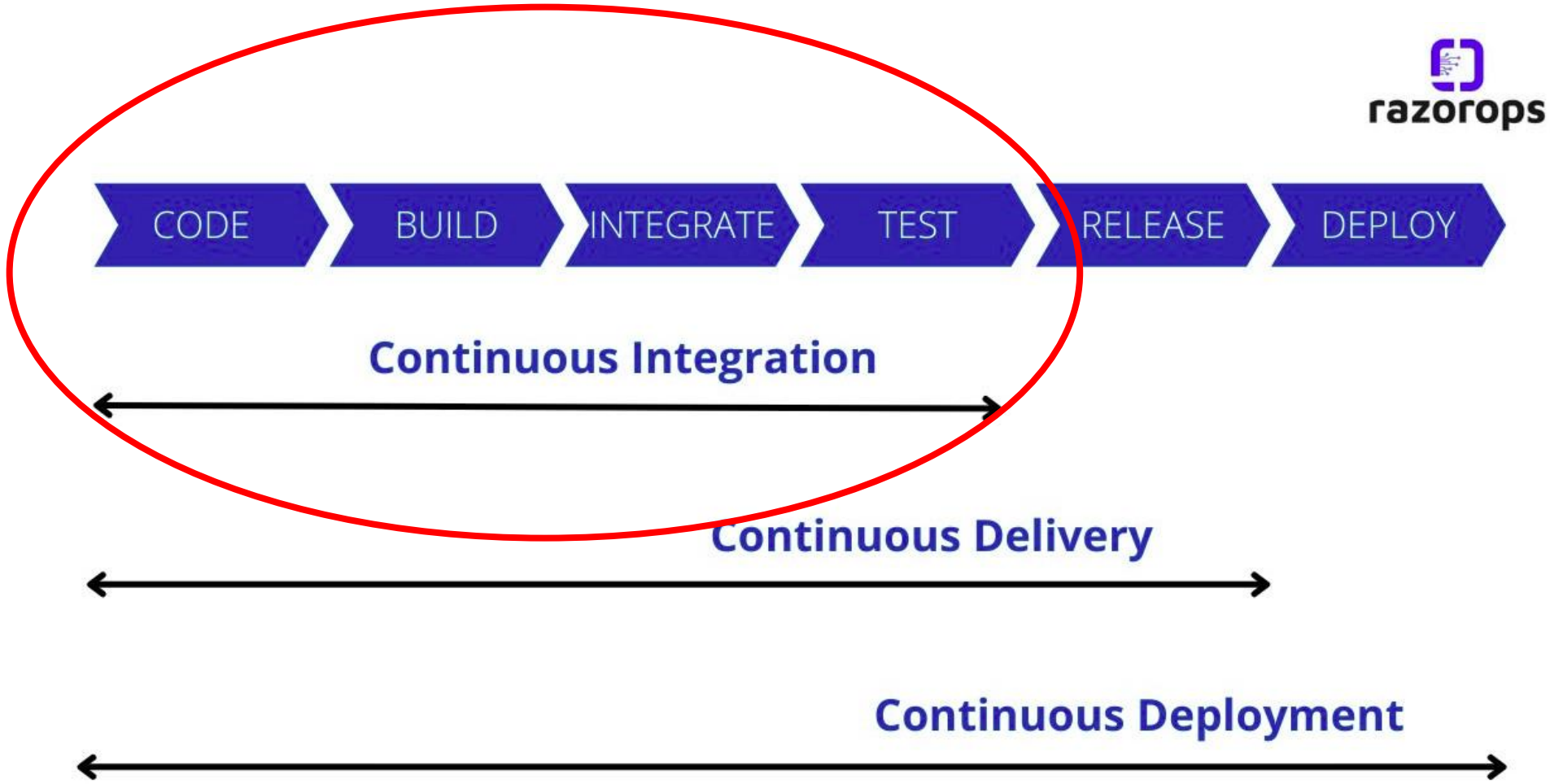


DevOps realm



https://www.openxcell.com/devops/

Focus on Continuous Integration

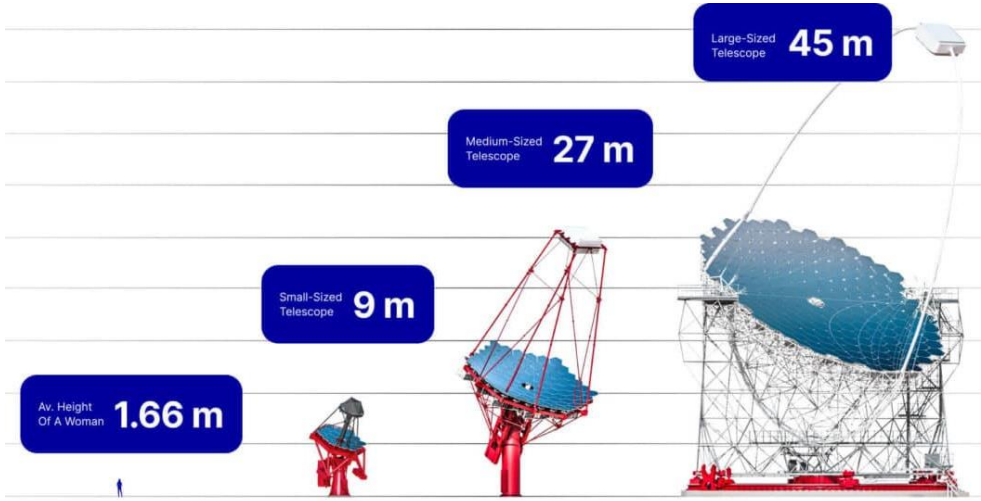


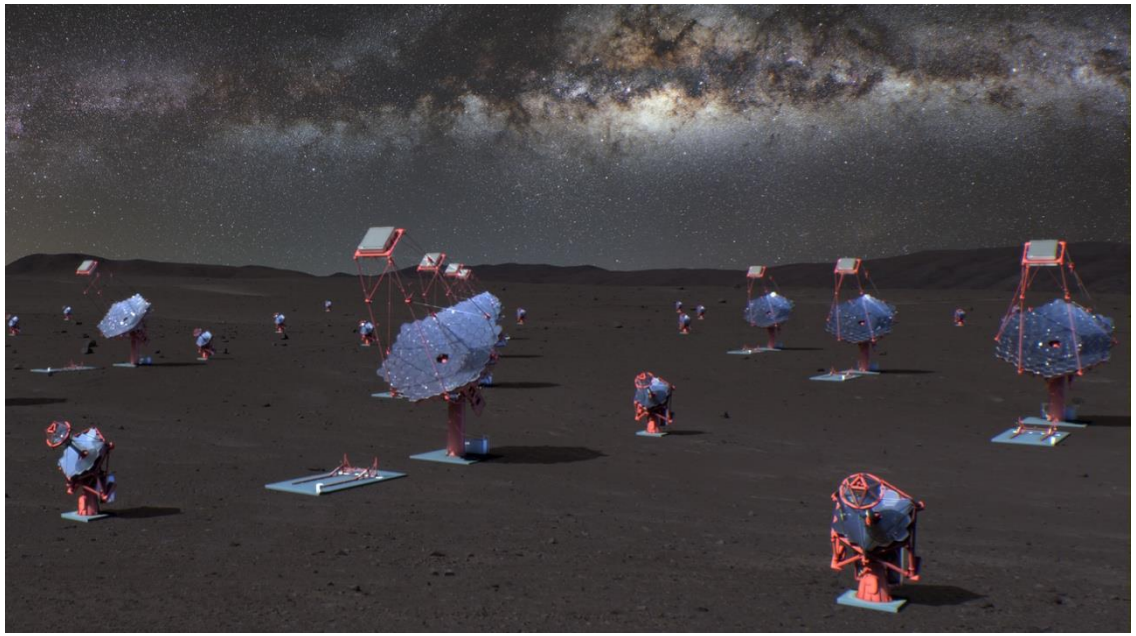
SCADA software for IACTs

The core software infrastructure

➔ **CTAO: ACADA**

➔ **ASTRI Mini-Array: SCADA**





Supervises distributed array elements on-site

Orchestrates telescope **control** and **observation** execution

Manages **data acquisition** and **compression** of raw data

Provides **lifecycle** management and human-machine interface

Continuous Integration in real life

CTAO ACADA

GitLab Jenkins sonarqube

ASTRI Mini-Array SCADA

GitLab CI/CD sonarqube