

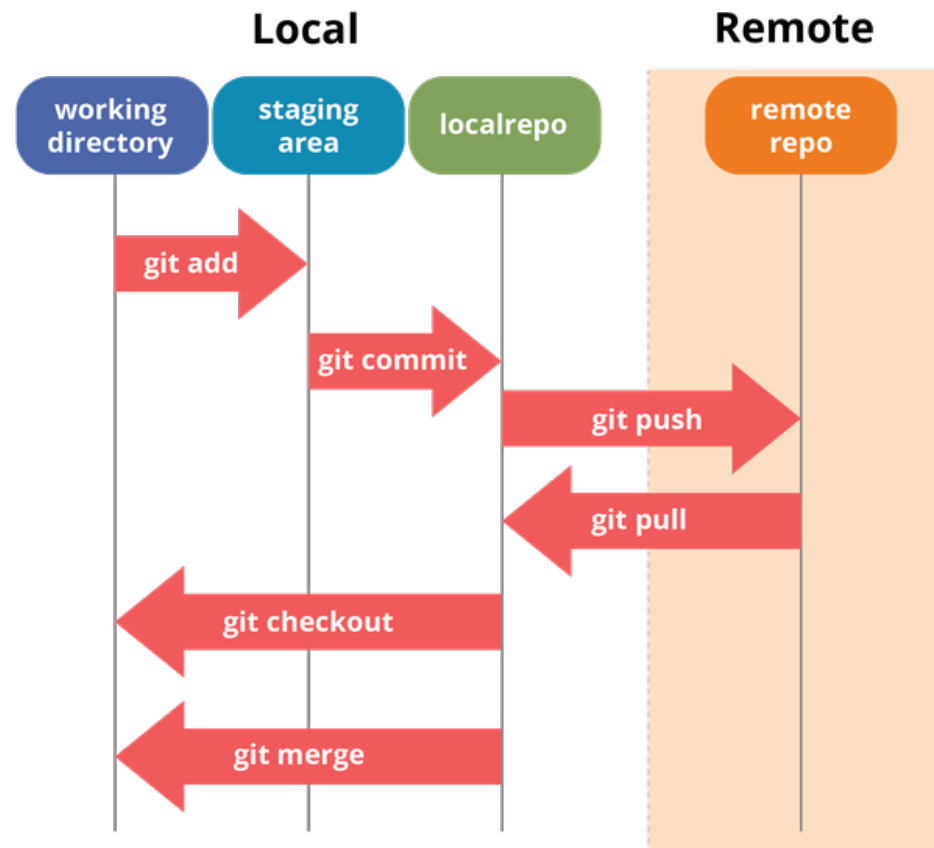
Management and Operations of Networks, Services, and Systems Repositories and CI/CD

Ricardo Morla

FEUP – GORS/M.EEC, GRS/M.EIC

Code Repository Workflow

- Add, commit, push
- Pull
- Checkout, merge



Branches

- Alternative “realities” or different versions of the code
- Always start with master
- Branch to try new concepts, features, different implementations
- Can merge branches into others
 - e.g. bring new features back to master

Continuous Integration / Continuous Deployment



- Version control
 - Found problem with code, can get back to an older version that worked
- Agile
 - Divide in small parts, implement
- Pipeline
 - Workflow of code delivery process

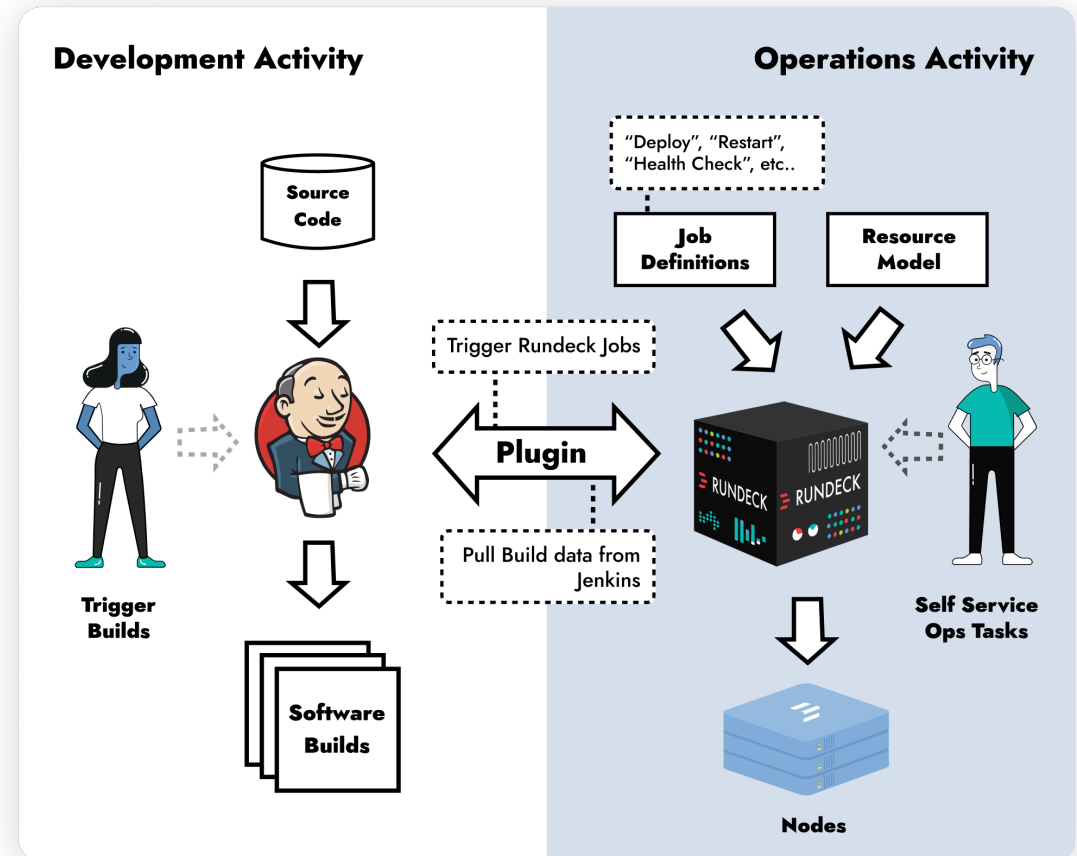
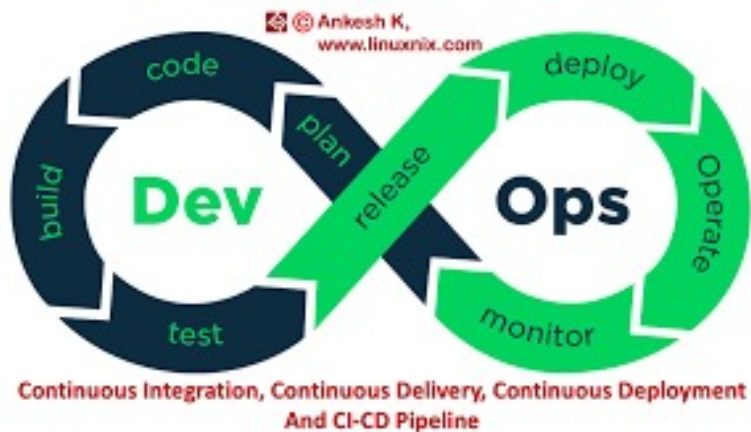
https://gruntwork.io/guides/automations/how-to-configure-a-production-grade-ci-cd-setup-for-apps-and-infrastructure-code/#cicd_workflows

Types of code

- Application code
 - Source tar.gz, jar, exe, docker image, VM image
 - The software you want to run in your infrastructure
- Infrastructure code
 - The code that you write to setup the infrastructure
 - Code that interacts with the docker API, etc
- Live infrastructure config
 - Frontend for infrastructure deployments
 - Parametrized for a particular deployment (e.g. dev vs. deploy1 vs. deploy2)

https://gruntwork.io/guides/automations/how-to-configure-a-production-grade-ci-cd-setup-for-apps-and-infrastructure-code/#cicd_workflows

Continuous Integration / Continuous Deployment





Code vs. build

- Straightforward distinction e.g. in C or Java ... or when you package repositories
 - Source code vs. executable binaries
- Need a build phase for infrastructure?
 - Depends on which language you write the infrastructure code and tools
 - What you're actually deploying are configuration files



Test

- Unit test
 - Sandbox or dry run tests of individual components / configurations
 - Some networking equipment has this option
- System test
 - Deploy the different network components and test their interactions
 - Testing environment? Network emulator, virtualization
- Staging environment
 - Replica of production environment
 - Test specifics of production environment (number of nodes, addresses, etc)



Release vs. deploy

- A release is the software product, ready to be deployed
- A deployment is a release configured for the target environment
- What's the difference for infrastructure?

Abstractions for infrastructure code

- Target: specific deployment of infrastructure
- Abstraction 1:
 - Function that takes parameters in and outputs configuration files
 - How do the configuration files end up in the infrastructure?
- Abstraction 2:
 - Code that takes care of generating and deploying configuration files
 - Based on parameters from the user

Example

https://archive.nanog.org/sites/default/files/monday_general_autobuild_vicente_63.28.pdf

- NETCONF communication with devices
- Ansible with NETCONF module for automation
- Github as repository
- Jenkins for automating CI/CD
- Operations
 - Build: gets facts from deployment, validates configurations
 - Test: dry run in target devices (JunOS, commit-check)
 - Deploy: see if there are changes pending (config vs state of devices), only deploy changes

Management and Operations of Networks, Services, and Systems Repositories and Continuous Integration

Ricardo Morla

FEUP – GORS/M.EEC, GRS/M.EIC