CISCO

You make **possible**

Automate your software delivery process

Artur Cieszkowski – Software Consulting Engineer
Guillaume Mulocher – Consulting Engineer

BRKOPS-1871

Barcelona | January 27-31, 2020

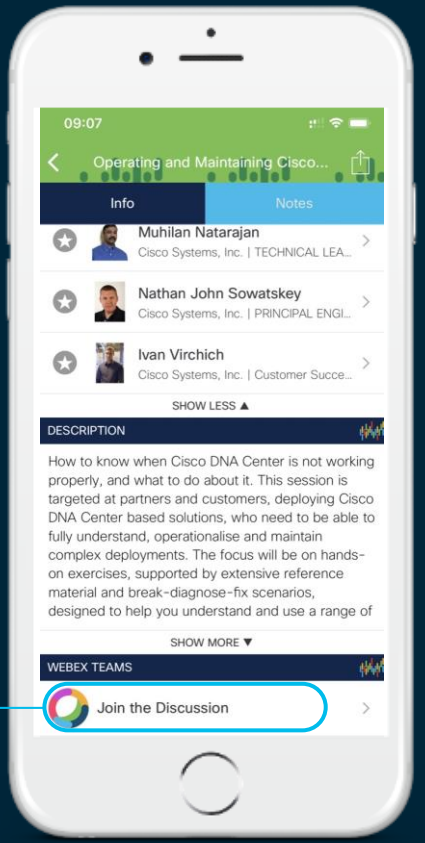
Cisco Webex Teams

Questions?

Use Cisco Webex Teams to chat with the speaker after the session

How

1. Find this session in the Cisco Events Mobile App
2. Click “Join the Discussion”
3. Install Webex Teams or go directly to the team space
4. Enter messages/questions in the team space



What to expect in this session

- Understanding basic concepts around CICD & DevOps
- Understanding the requirements to develop software collectively and efficiently
- See how to automatically build an example of “network related” application
- Automate Unit tests and Functional tests
- An explanation through concrete example

What this session is not

- An advocate for any specific tool
- An in-depth discussion about Agile methodology
- An advanced technical discussion

Agenda

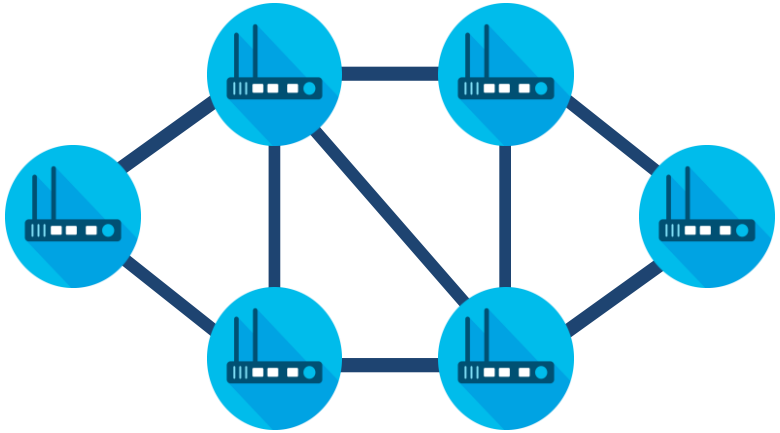
- Short-term thinking in Software Delivery
- CICD overview
- Introducing CICD in Software Development (with demos)
  - Code repository
  - Tests
  - Code quality
  - Deliver / Deploy
  - Pipeline
  - Process
- Conclusions
- Q&A

Short-term thinking in Software Delivery

Company ACME needs

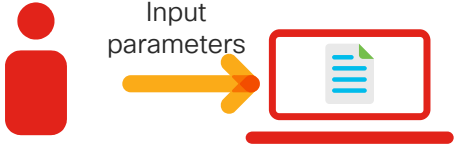
- Small Service Provider
- 6 Core routers with Segment-Routing
- Provides L3VPN for customers

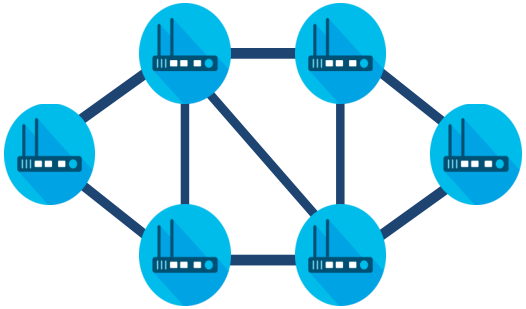
- For each new customer more or less the same configuration
- All configured manually



Birth of an automation tool

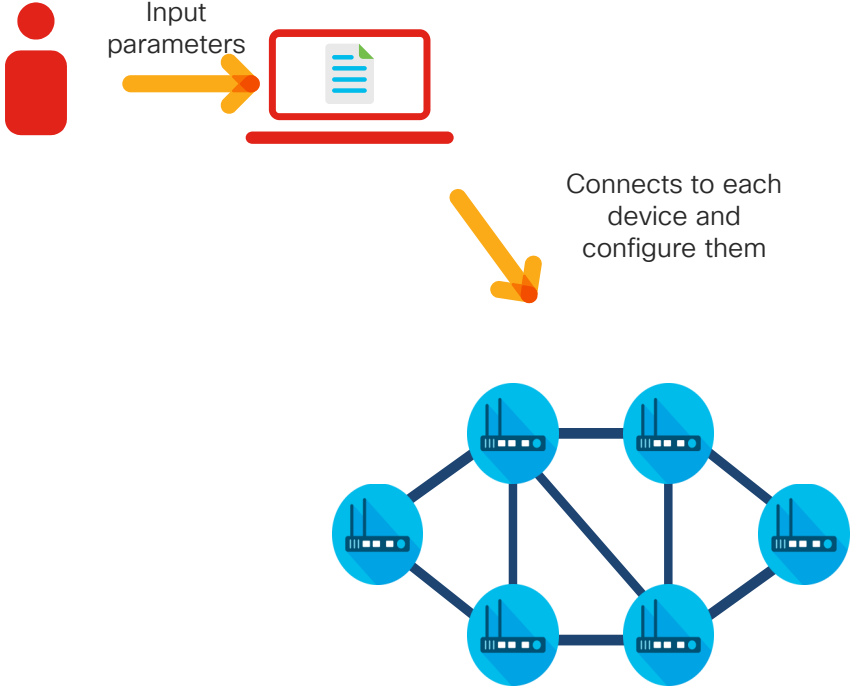
- One person writes a parametrized script to automate repetitive task
- Works perfectly for a given scenario and environment
- Other engineers start to rely on the script
- Limited number of people can maintain the tool





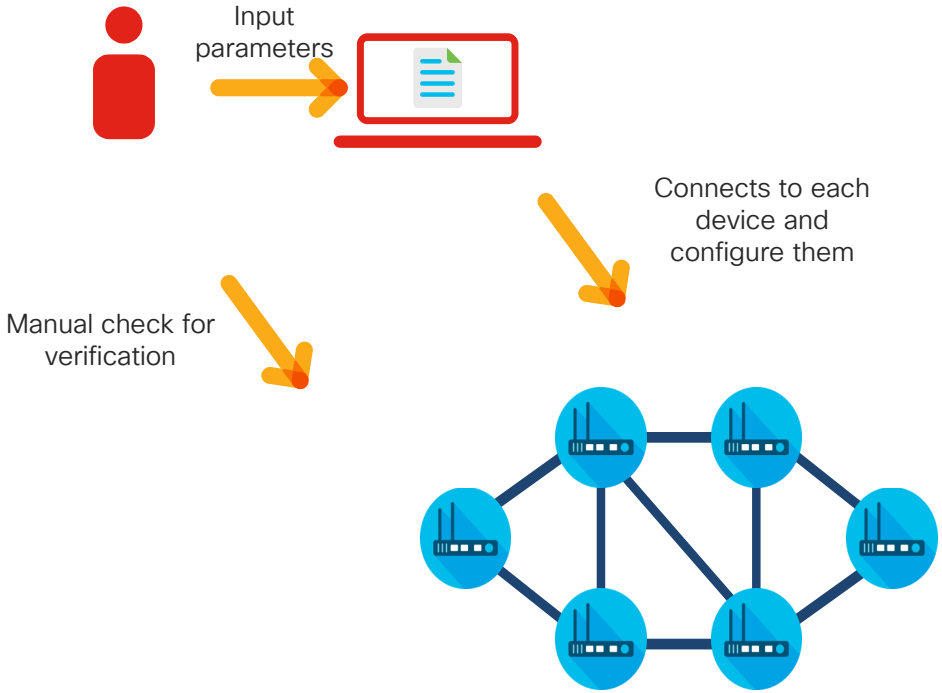
Birth of an automation tool

- One person writes a parametrized script to automate repetitive task
- Works perfectly for a given scenario and environment
- Other engineers start to rely on the script
- Limited number of people can maintain the tool

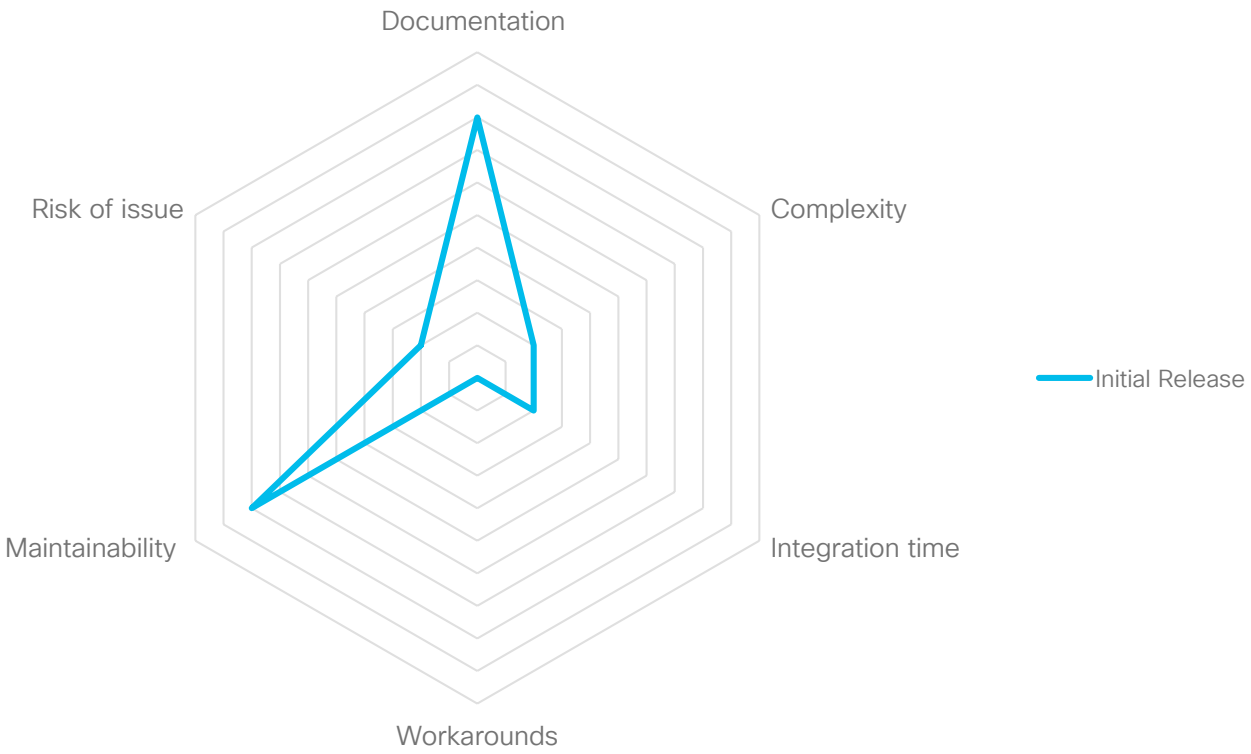


Birth of an automation tool

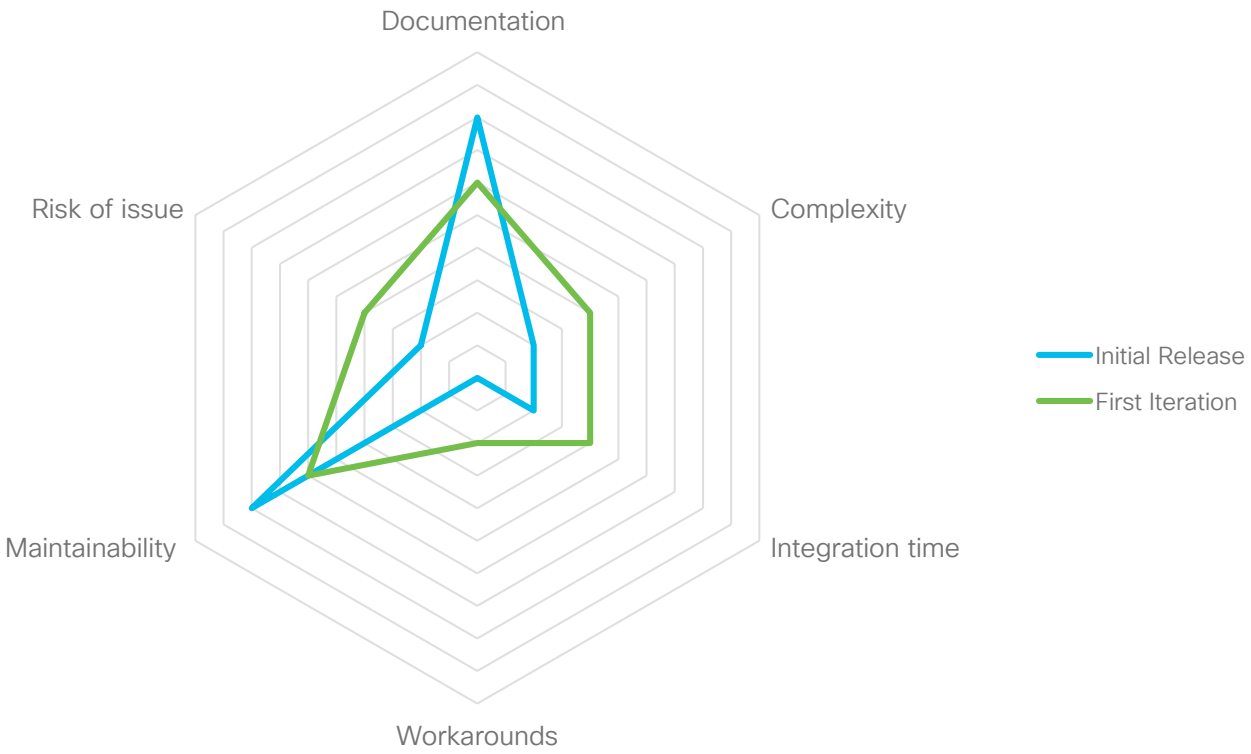
- One person writes a parametrized script to automate repetitive task
- Works perfectly for a given scenario and environment
- Other engineers start to rely on the script
- Limited number of people can maintain the tool



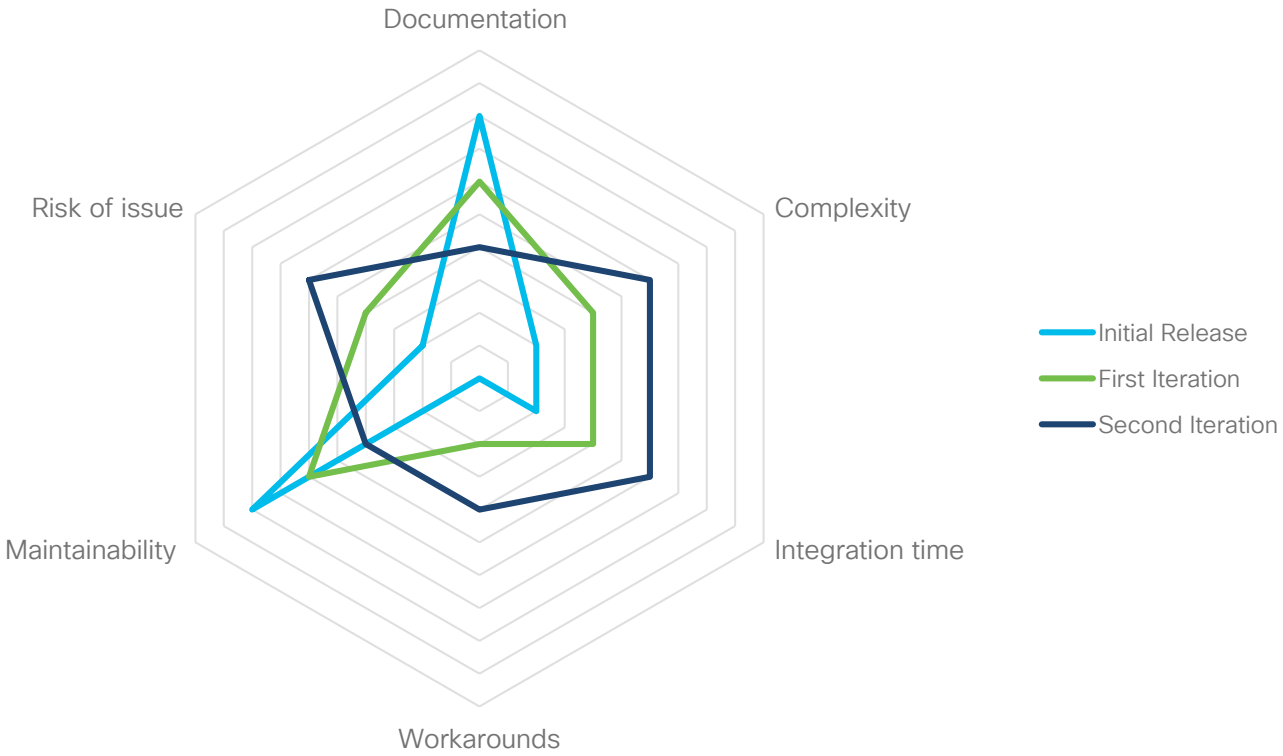
Evolution of the software



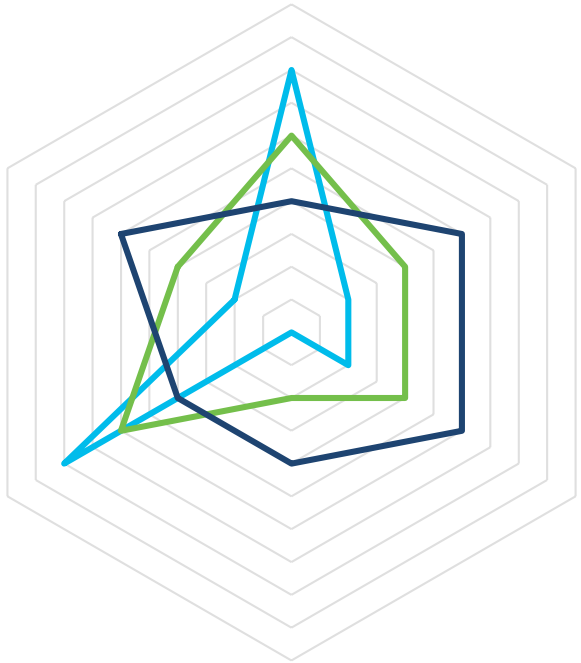
Evolution of the software



Evolution of the software



Evolution of the software



→ Documentation gaps

→ Difficult codebase to work with

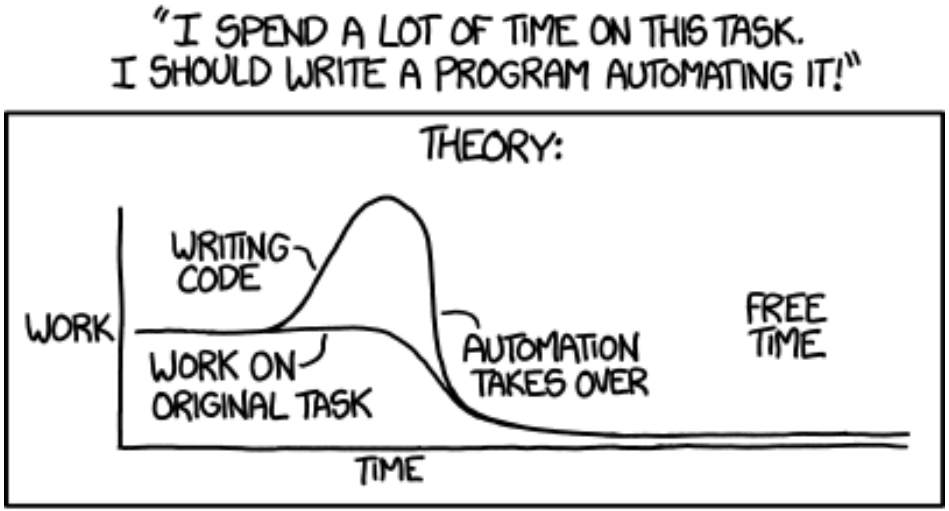
→ New features take longer to get in

→ Long lived temporary hacks

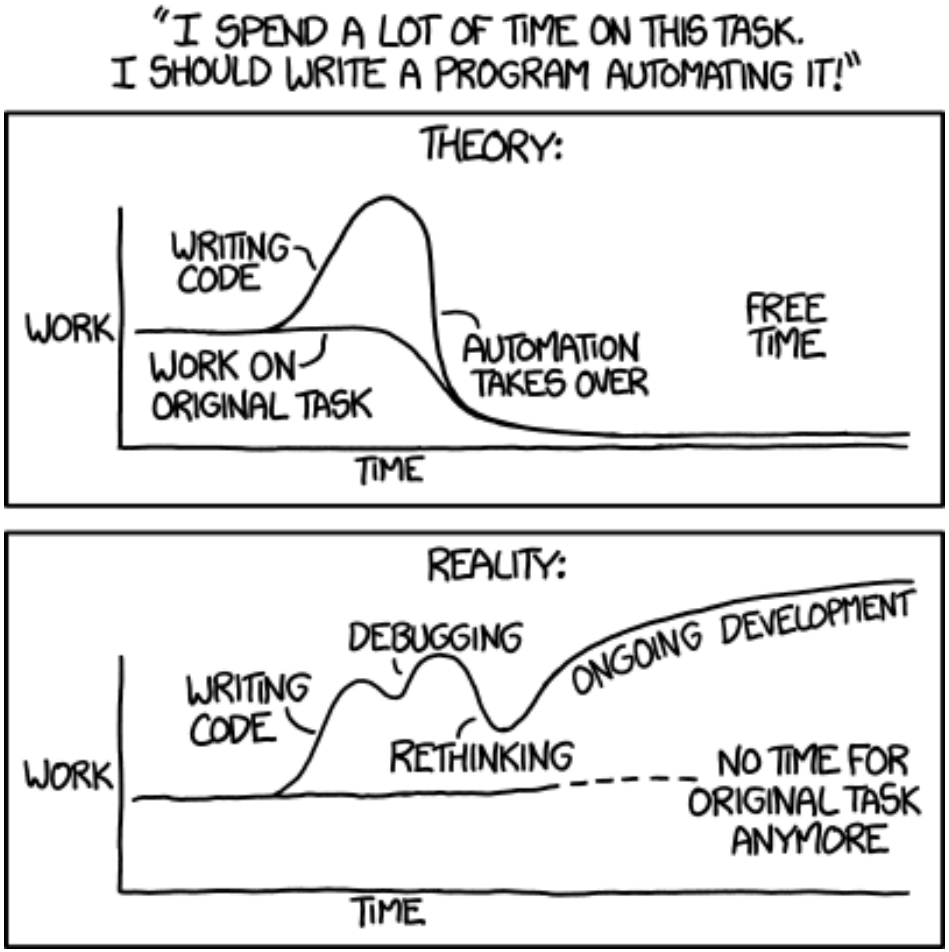
→ Technical debt increases

→ Decrease in test coverage

source: https://xkcd.com/1319/



source: https://xkcd.com/1319/

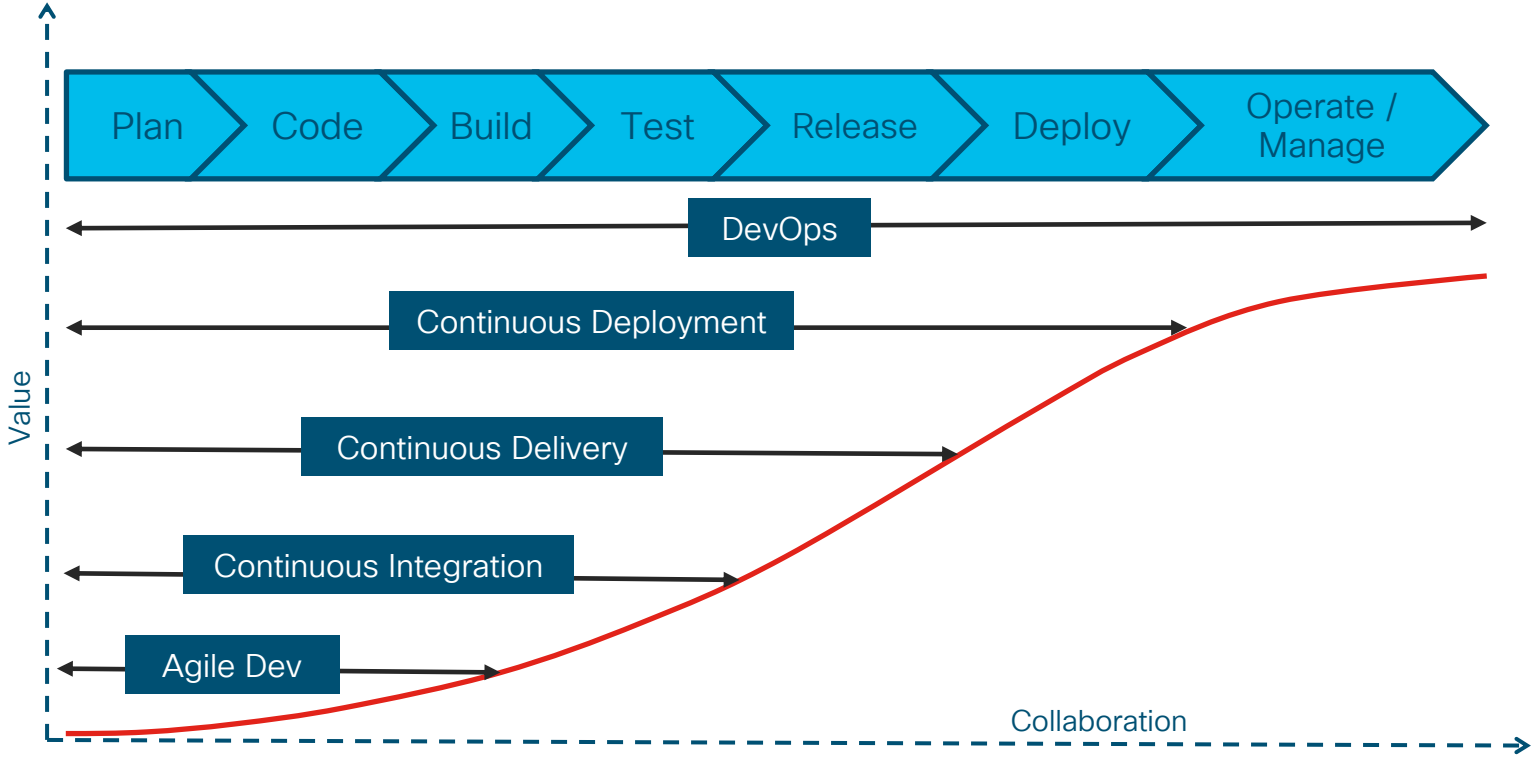


source: https://xkcd.com/1319/

CICD Overview

Continuous ...?

- Continuous Integration / Continuous Delivery (CI/CD)



Objectives

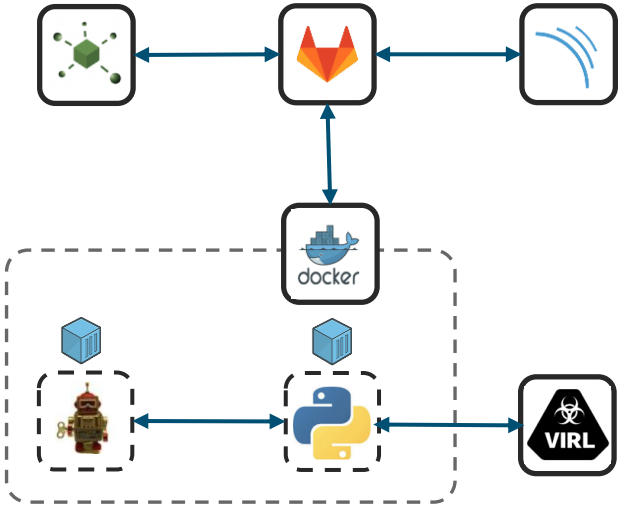
- Create a Continuous Deployment pipeline
- Triggered on git changes
- Perform Automated Unit Tests
- Run Static Analysis
- Perform Automated Functional & Integration Tests
- Notify Interested parties
- Publish Release Candidate
- Tear down Test environment

Introducing CICD in Software Development (with demos)

Agenda

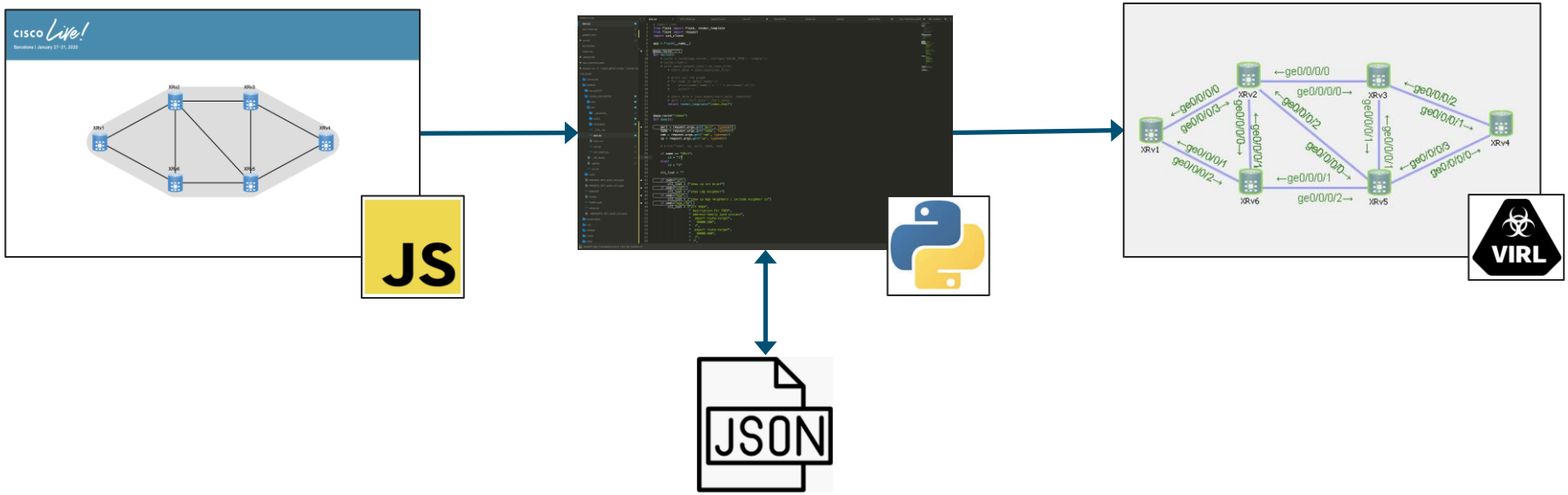
- Short-term thinking in Software Delivery
- CICD overview
- Introducing CICD in Software Development (with demos)
  - Code repository
  - Tests
  - Code quality
  - Deliver / Deploy
  - Pipeline
  - Process
- Conclusions
- Q&A

Demo setup

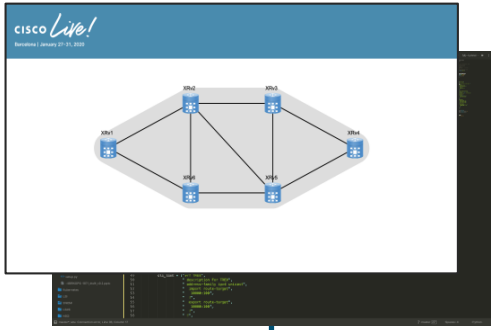


- Gitlab – Git Repo & CICD Server
- Nexus – Binary Repo
- SonarQube – Code Quality
- Docker – Runs Containers
  - On demand TEST environment
  - CXTA – Testing Framework
  - Python App
- VIRL – Cisco Virtual Lab
  - Real device OS!
  - Simulate customer environment
  - Test various topologies
  - Introduce real network issues

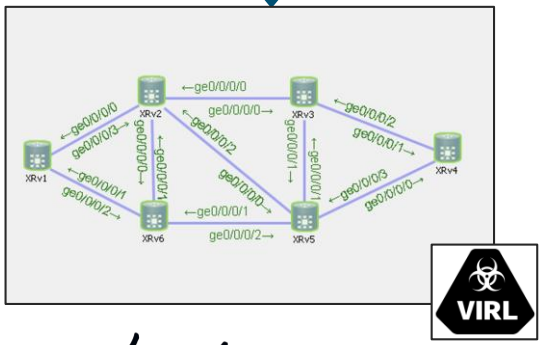
System under test

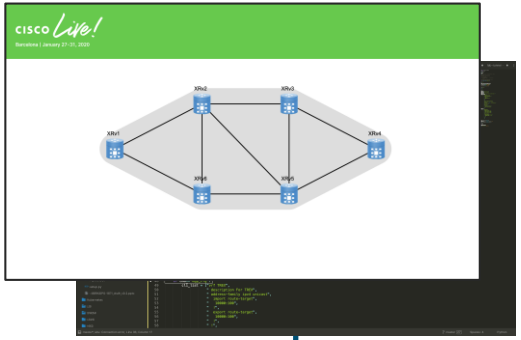


Environments

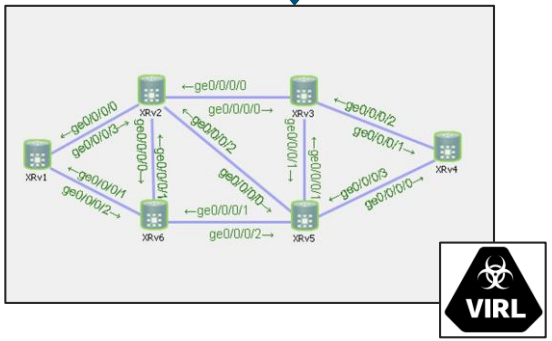


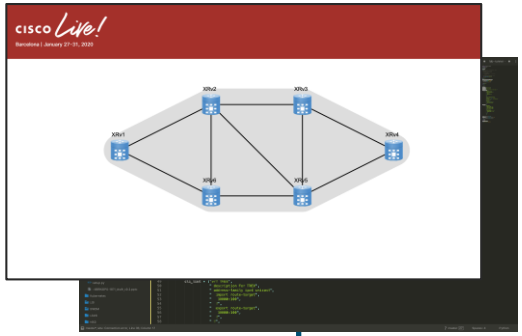
DEV



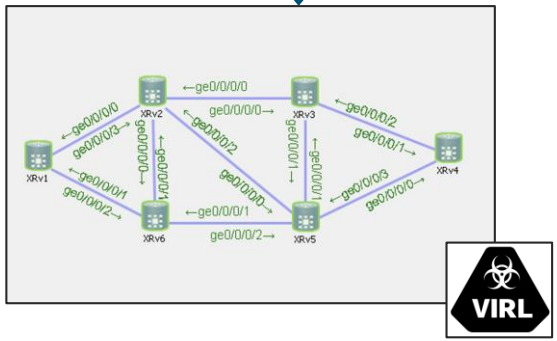


TEST

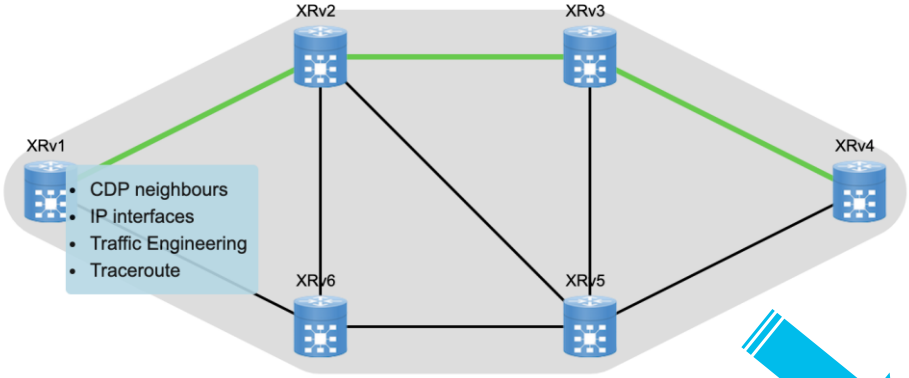


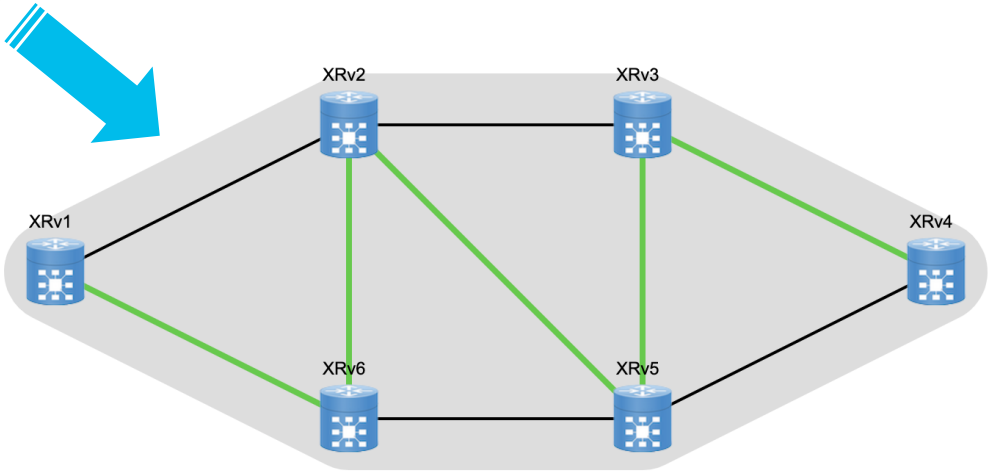


PROD



Introducing new feature





Code repository

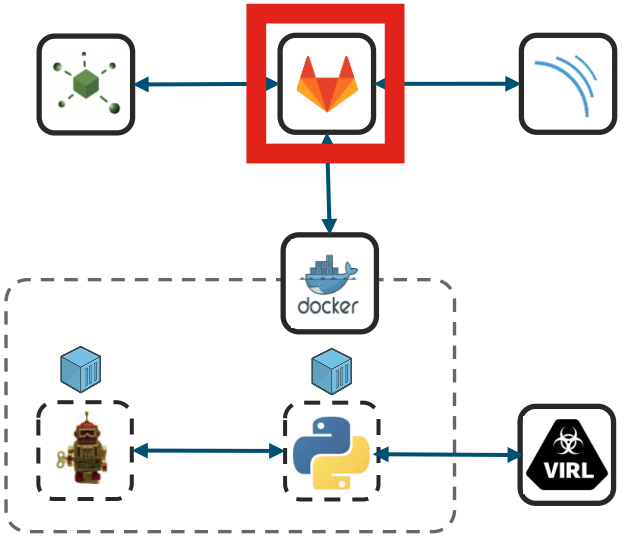
About Version Control System (VCS)

- Versions of the software hosted on a server available to the development team

- Track changes made to documents of the software

- Centralized (e.g. CVS, Subversion, ...) vs Distributed (e.g. Git, Mercurial, ...)

Code repository

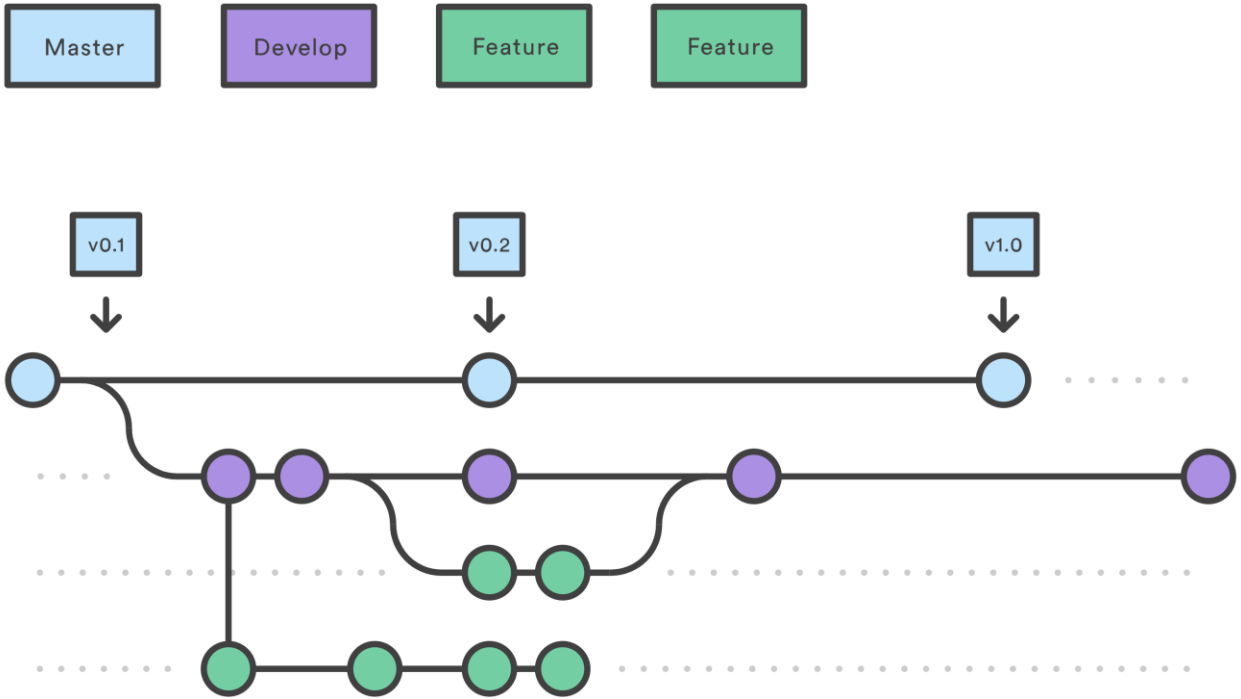


- Version Control System contains:
  - Source code
  - Environment specific configuration
  - Code of the pipeline
  - Tests
- Pipeline builds
  triggered on every change
- Feedback sent to dev teams

A branching strategy is needed

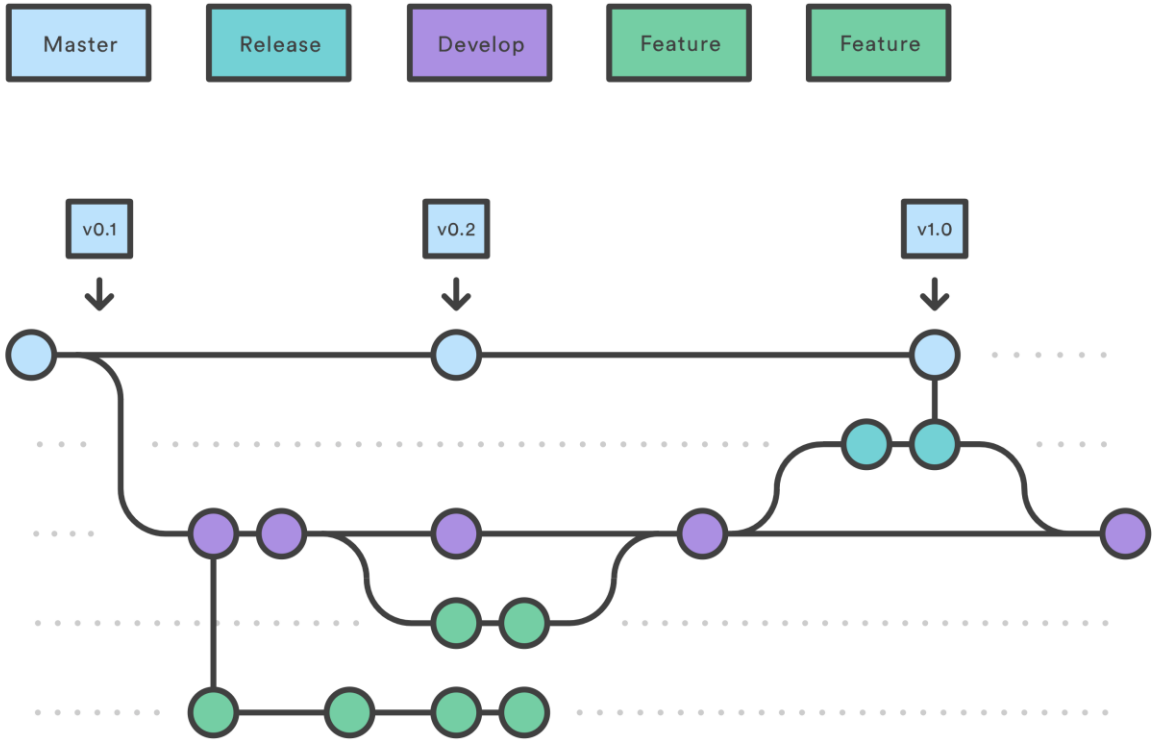
- VCS is a tool
- No one size fits all
- Various strategies:
  - Trunk-Based Development
  - Git Flow
  - Gitlab Flow
- Communicate it

Branching strategy for the demo



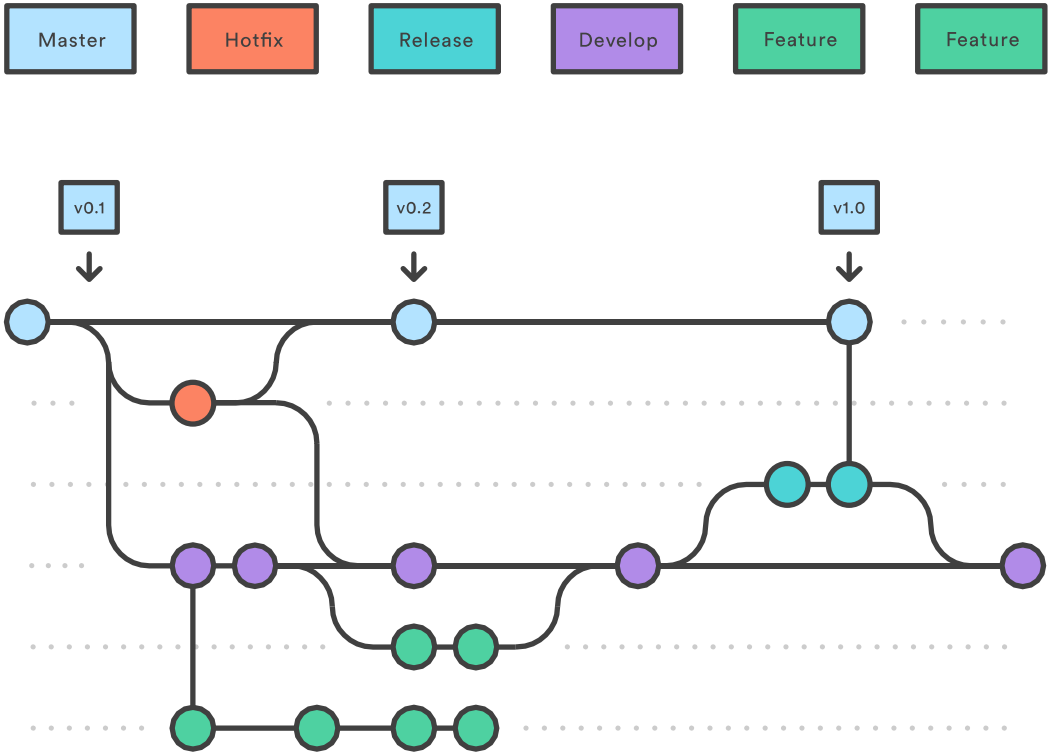
source: https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Branching strategy for the demo



source: https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Branching strategy for the demo



source: https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Testing

Testing

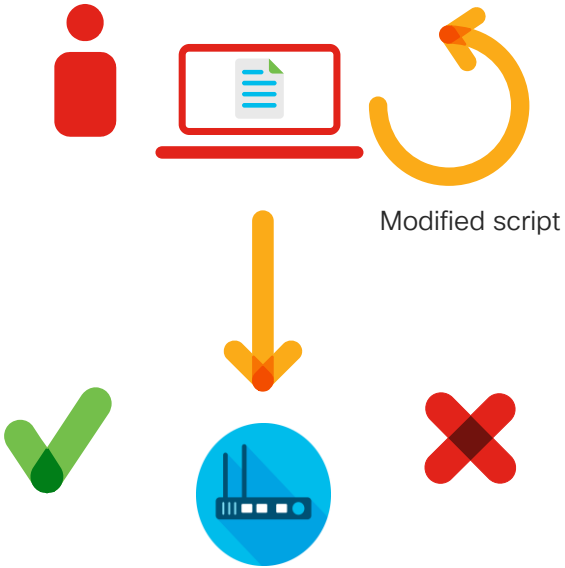
- Verify code does what it is supposed to do
- Verify changes do not break (regression)

- Running tests manually takes time
- The more the software grows, the bigger the test suites, the longer it takes

- Types of tests:

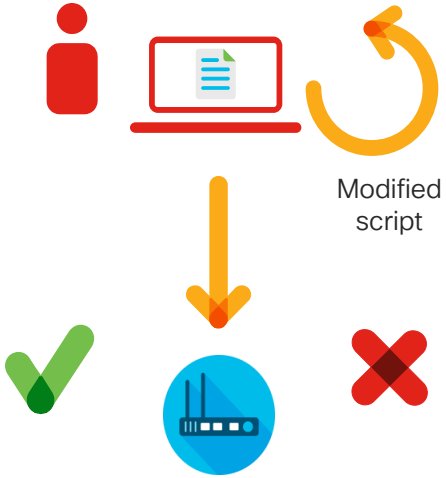
  Unit tests, Functional tests, Integration tests, Platform tests, Load tests, Security tests, E2E tests

- Testing strategy!



Unit tests

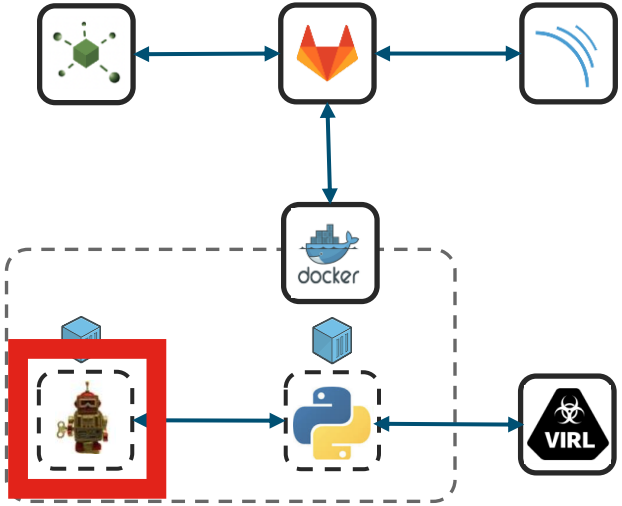
- Language specific
- Low level
- Fast tests
- At the function / method / class level



Code coverage

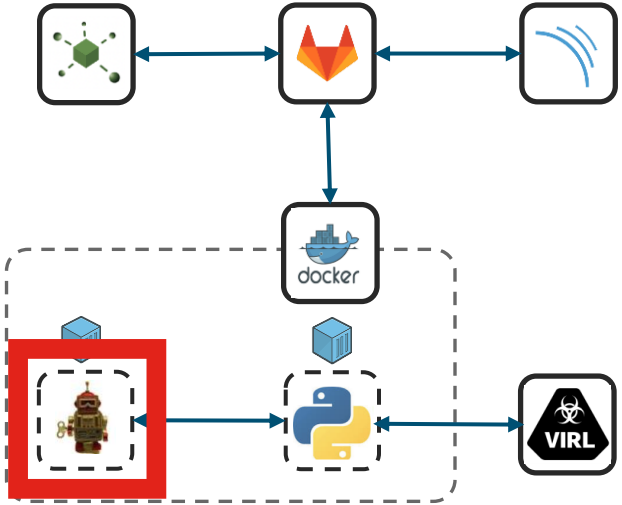
- Important metric on the code relative to its unit tests
- Percentage of the code which have been ”covered” by the tests.
- A part of the code not “covered” has not been tested.
- Strive for a high percentage of coverage (Do not waste time to attain 100%)

Functional Testing - CXTA
Cisco



- CXTA (CX Test Automation)
- Testing Framework
- Based on open-source Robot Framework
  - natural language
  - schedules executing
  - reporting
  - test cases to be written/controlled by non-programmers
- Integrates Python test libraries created within Cisco (PyATS) including projects (Singtel/NTT, NGENA, BU tests) so we can re-use/leverage existing work

Functional Testing - CXTA
Cisco



Test Engineer writes test cases using the Robot framework

Robot Script

```
*** Test Cases ***
Connect to device
    use testbed testbed.yaml
    connect to device google_D9
    command show_interface ${kwdargs}
```

Keyword Library

```
@keyword(u'execute command "${command}" on device "${device}"')
def execute_command_on_device(self, command=None, device=None):
    devices = self.testbed.devices
    rtr = devices[device]
    output = rtr.execute(command)
    self.command_output[device] = {}
    self.command_output[device][command] = output
    return output
```

Developer writes Python for new keywords

Code quality

Code quality analysis does not replace code review

- Save time for what can be automated:
  - spell check
  - syntax
- Help the team focus on what is important:
  - Design Review
  - Problem Solving
  - Test plan
- Code reviews bring additional benefits to the team

Successful tests do not mean high quality software

Need to ensure Code Quality

Use Existing tools

Follow Standards

Iterative process

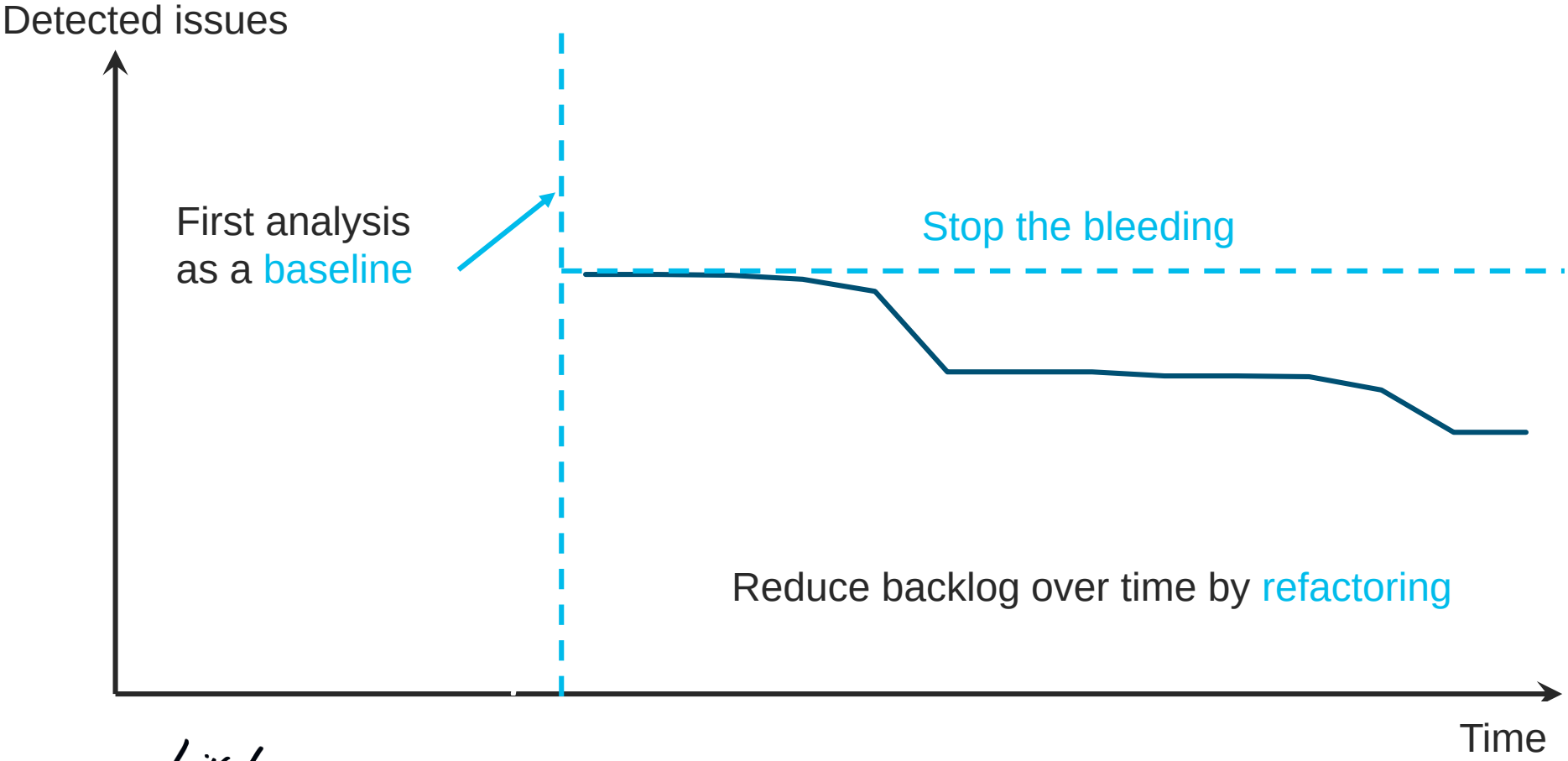
Static analysis should be done from the start

Prevent accumulation of Technical Debt

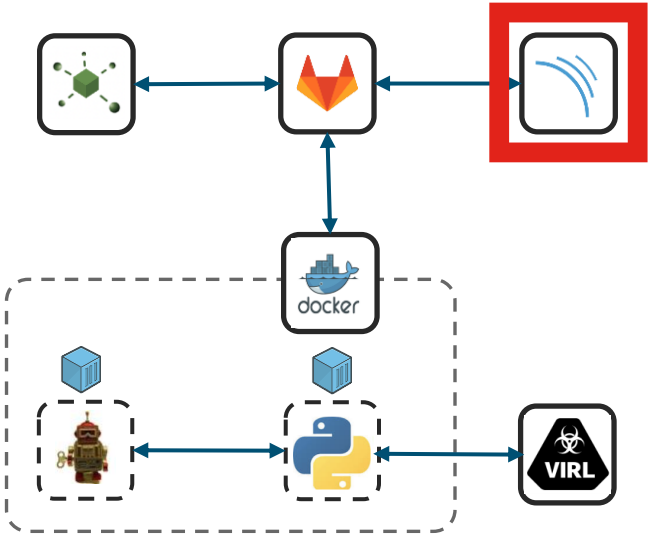
Enforce Consistency in the codebase

Educate developers on best coding practices

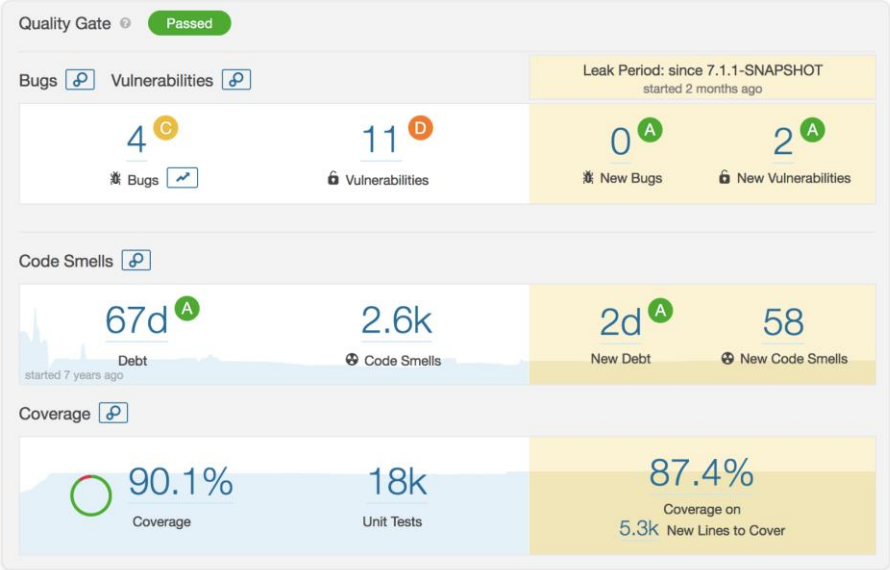
It is still worth doing it on existing project



SonarQube
SonarSource



- Static analysis
  - Bugs, Code smells, Vulnerabilities
  - Unit Tests
  - Code Coverage
  - Break the build



Deliver / Deploy

CD – Continuous Delivery

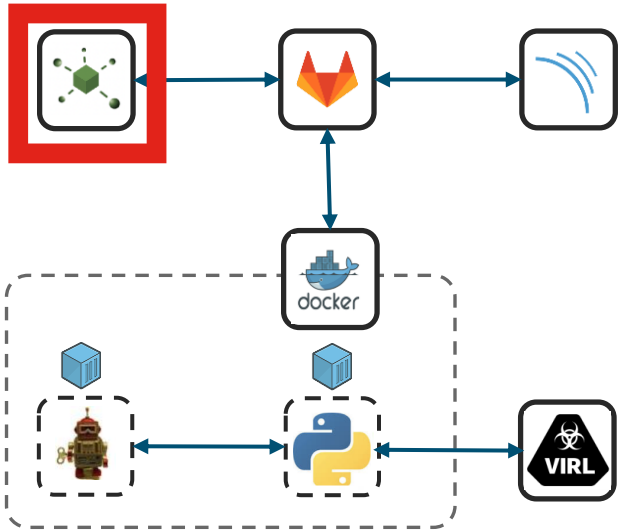
- All verifications (build, unit tests, code quality, functional tests,...) passed.
- The build can be promoted to deployable version
- Use a consistent versioning scheme
- Tag the artifact with the new version and make it available in a central repository

CD – Continuous Deployment

- Triggered on Delivery
- Automatically deploys a new version to production environment
- Manual deployment to production for example after approval

Nexus repository

Sonatype



- A storage for the artifacts
  - Docker images here
- Possibility to deploy them to production environment

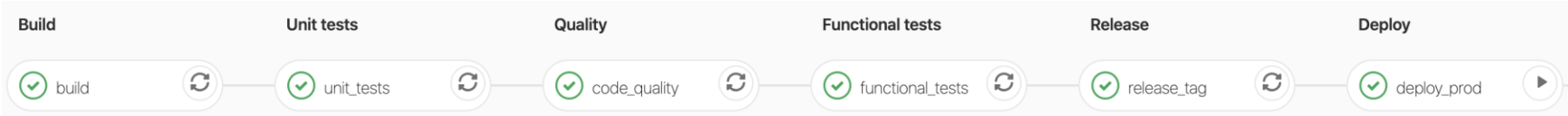
Pipeline

The pipeline verifies the changes are valid

Production like environment

Fail fast strategy

- Verify Build
- Fast Tests passing
- Check new code quality
- Slow Tests passing
- Create deployable artifacts for certain branches



Pipeline Concept

Pull changes

Changes in the code repository trigger the pipeline

Unit Test

Run the Unit Tests

Functional Test

Run the functional tests.

Deploy (Optional)

Deploy to Production.

Pull Changes | Build | Unit Test | Code Quality | Functional Test | Deliver | Deploy

Build

Build the piece of software. The build will be used in all the pipeline

Code Quality

Verify Code Quality

Deliver

Publish an artifact once validated to some artefact server

Process

Tools are not enough, processes are needed

- Release management
- Versioning
- Branching strategy
- Testing strategy
- Split work (Sprint)
- Follow methodology (Agile)
- Track features (backlog)
- Ticketing system for bugs

Track features in Agile manner

Feature must have:

- owner
- due date
- description
- be trackable

Should be:

- estimated
- assigned to sprints
- divided into smaller stories or tasks

CiscoLive › barna2020 › Issues › #2

Open Opened just now by Administrator

Close issue New issue

my_feature

Create a new set of functionality

0 0

Show all activity Create branch

Administrator @root changed due date to January 15, 2020 31 seconds ago

Write Preview

Write a comment or drag your files here…

Markdown and quick actions are supported

Comment Close issue

Create merge request and branch

Create branch

Branch name

2-my_feature

Branch name is available

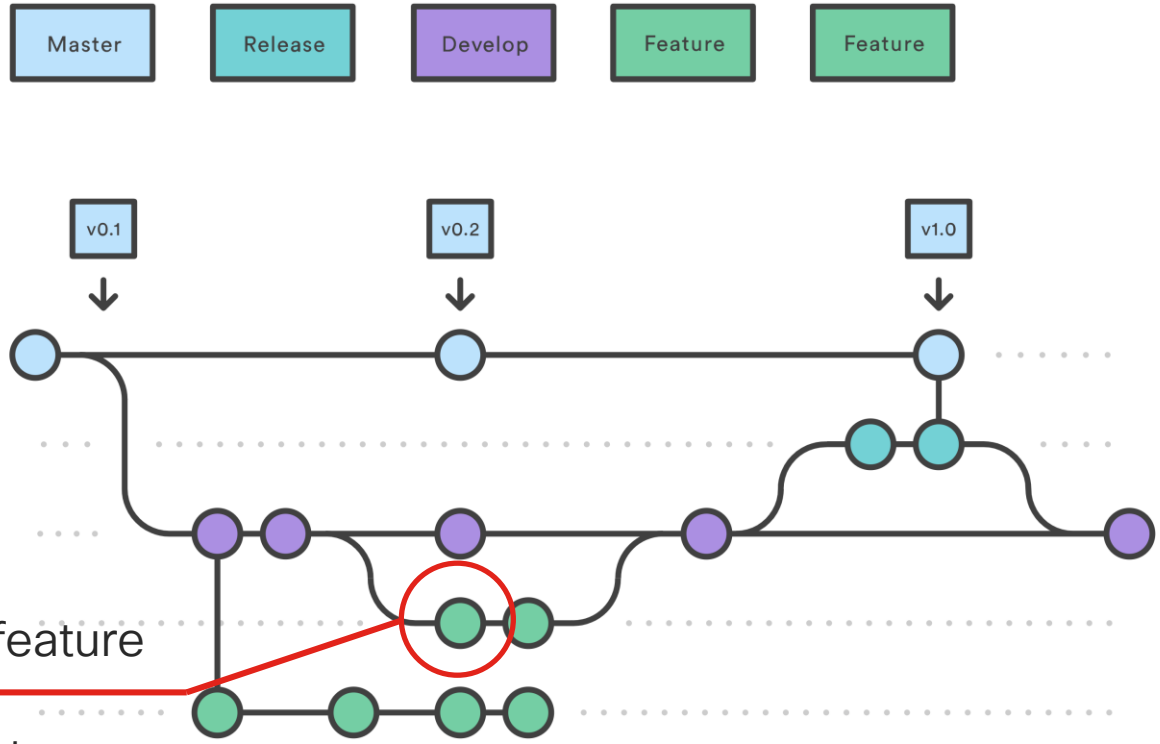
Source (branch or tag)

develop

Source is available

Create branch

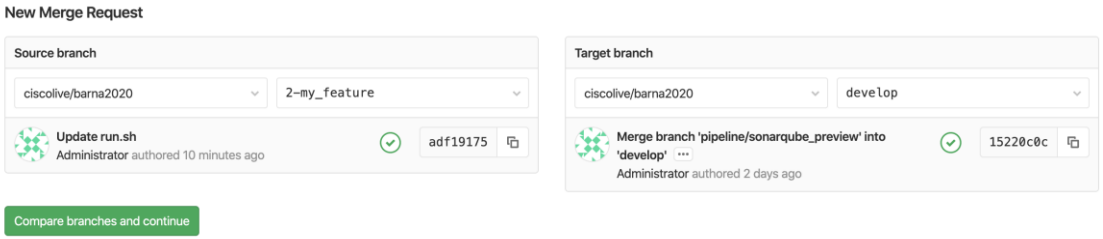
Feature branches

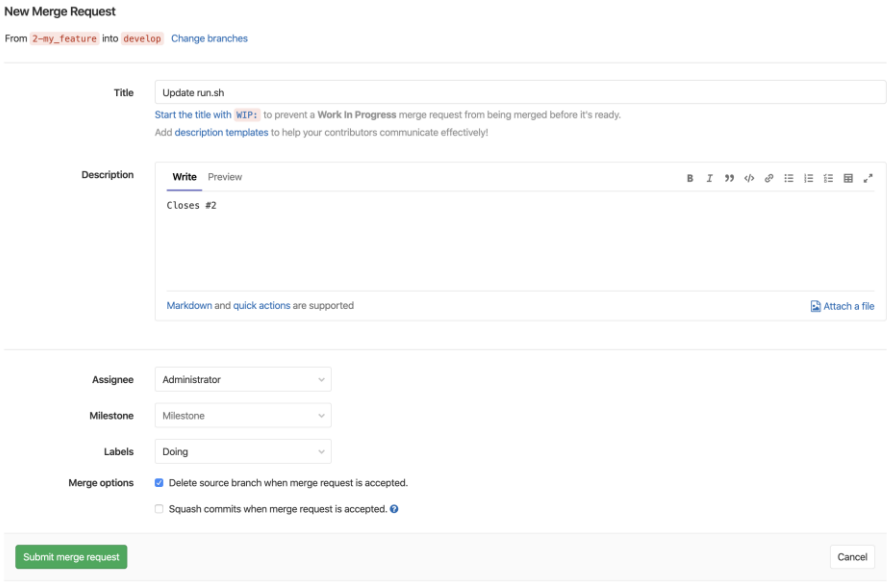


work on new feature

automated test cases
fired on every commit

Track features in Agile manner

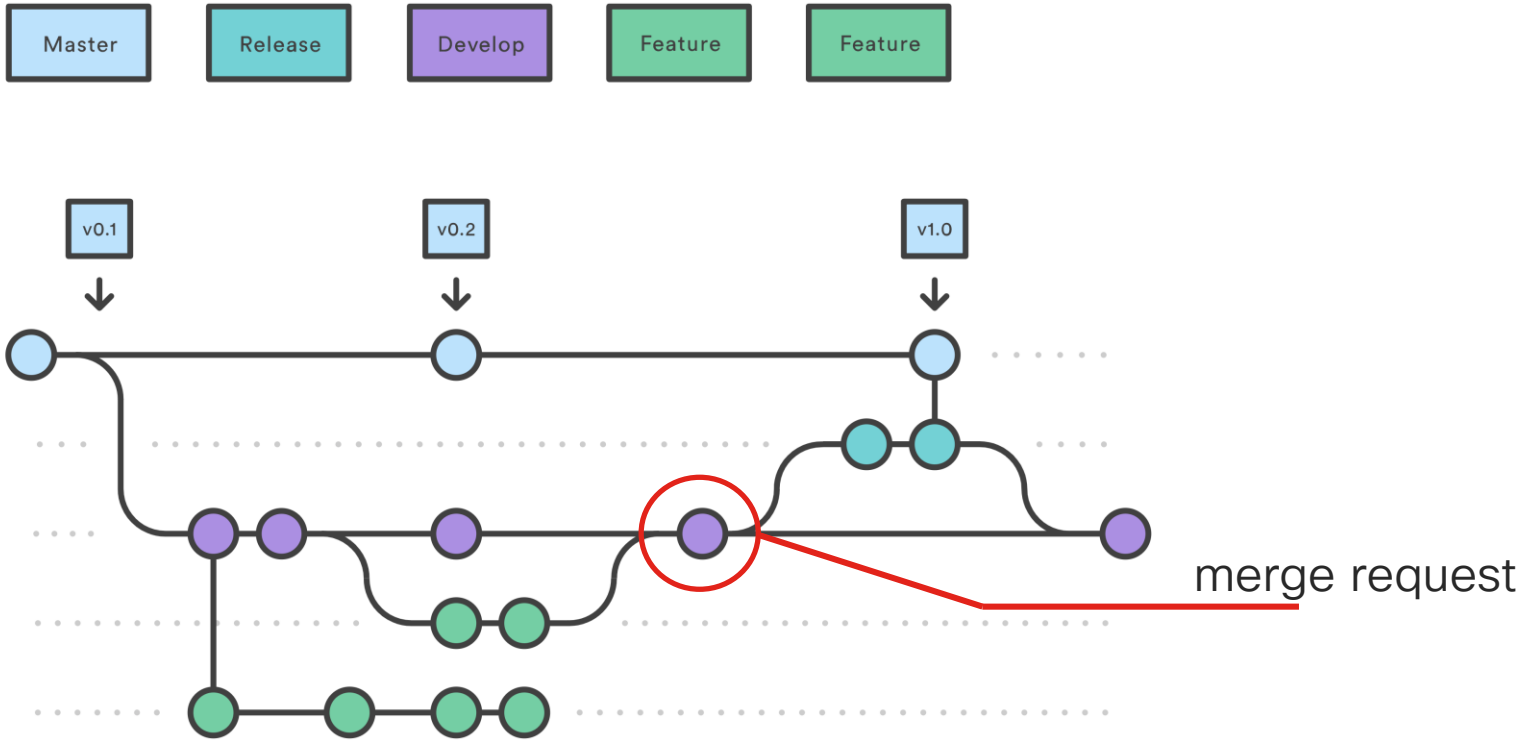




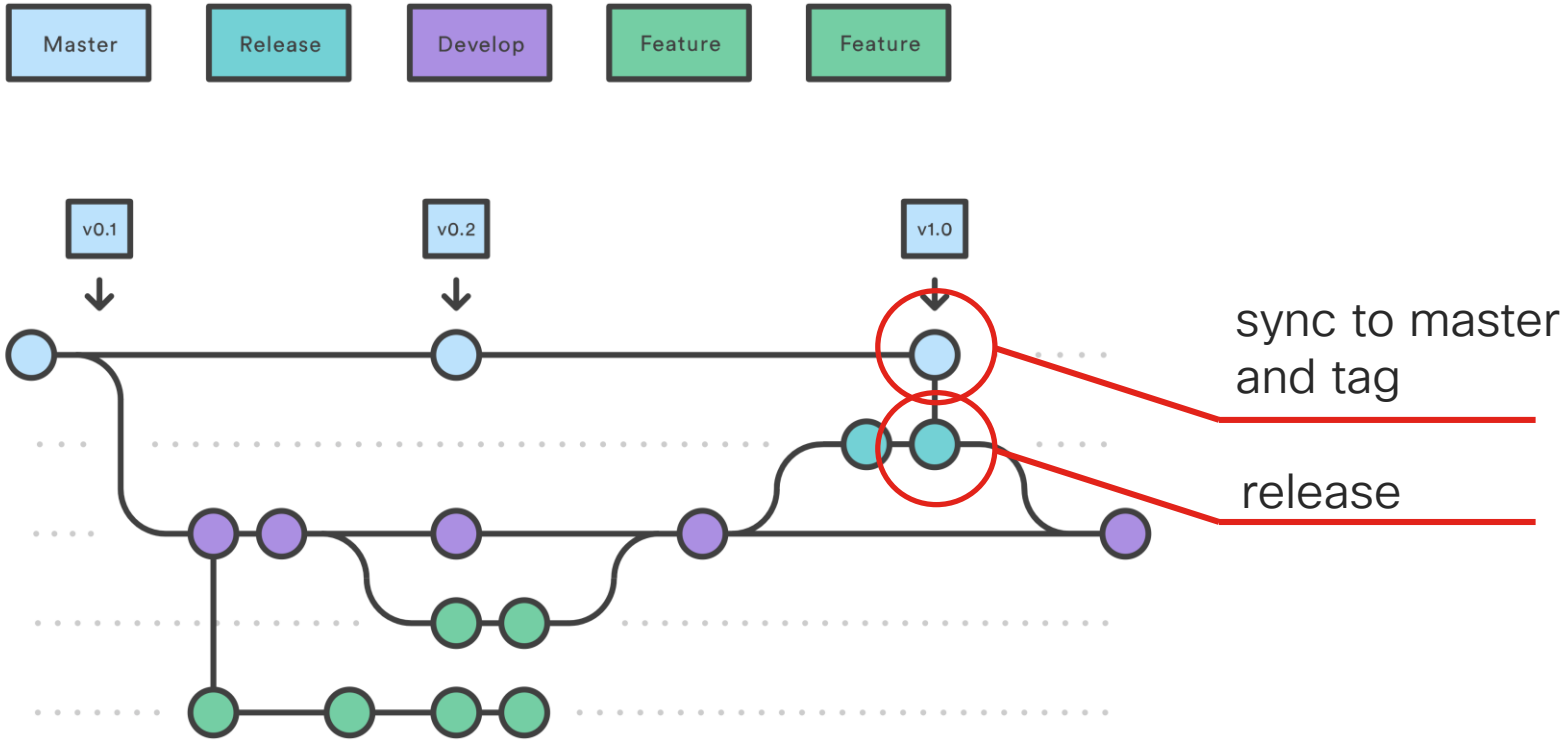
Prior to merge both pipelines should pass

On merge target branch pipeline should pass

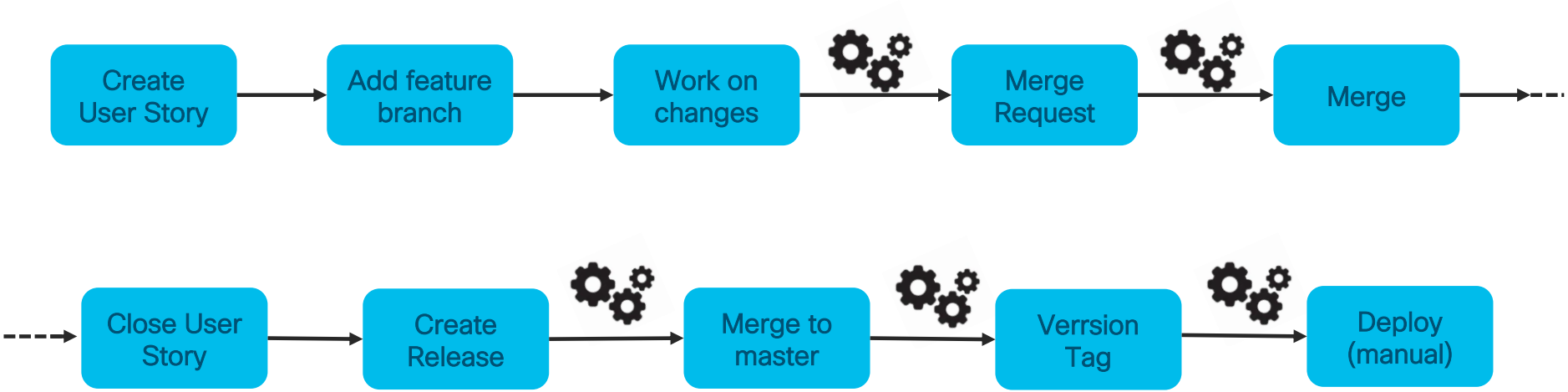
Merge



Release



Process

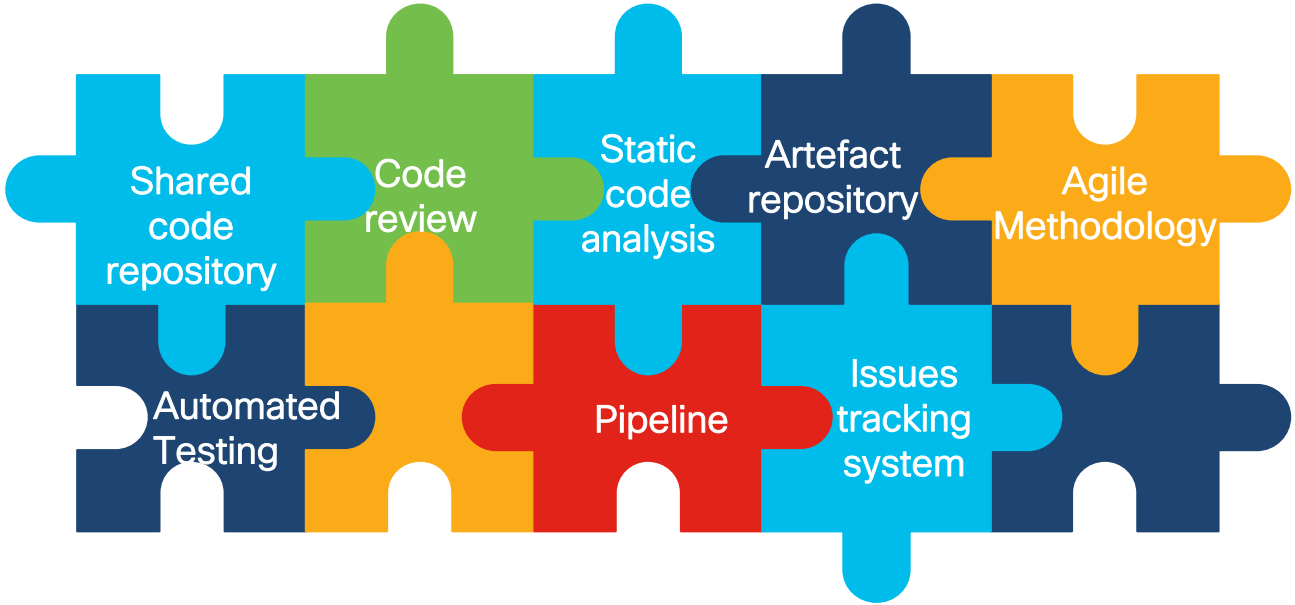


Conclusions

Objectives

- ☑ Create a Continuous Deployment pipeline
- ☑ Triggered on git changes
- ☑ Perform Automated Unit Tests
- ☑ Run Static Analysis
- ☑ Perform Automated Functional & Integration Tests
- ☑ Notify Interested parties
- ☑ Publish Release Candidate
- ☑ Tear down Test environment

Bringing it all together



Questions?

What next ?

At Cisco Live

- Webex teams (cs.co/ciscolivebot#BRKOPS-1871)
- Meet the Engineer
- See other sessions on CI/CD and DevOps

After Cisco Live

- Starting a new software development project?
  - Take the time to introduce CI/CD from the start!
- Already have one ongoing project?
  - What are the one or two points from this presentation missing in the project? Introduce them!

Cisco Live sessions on the topic:

Tuesday, January 28

- DevNet Workshop: SD-WAN DevOps Step 3: Continuous Integration/Continuous Deployment - DEVWKS-2030 (01:00 PM - 01:45 PM)
- DevNet Workshop: Utilizing Cisco CXTA service framework to validate network elements - DEVWKS-1407 (05:00 PM - 05:45 PM)

Wednesday, January 29

- CICD Pipelines for Cisco's IoT Edge compute platforms - DEVNET-1559 (04:00 PM - 04:45 PM)
- DevNet Workshop: SD-WAN DevOps Step 1: Automating Test Environments - DEVWKS-2226 (10:00 AM - 10:45 AM)
- DevOps with CloudCenter and Kubernetes in a multicloud environment - BRKCLD-2826 (11:00 AM - 12:00 PM)
- DevNet Workshop: Deploy and Manage Microservice-based Applications across Multicloud - DEVWKS-2986 (01:00 PM - 01:45 PM)
- DevNet Workshop: SD-WAN DevOps Step 2: Automating Day 1 & 2 Configuration/Operations - DEVWKS-2028 (02:00 PM - 02:45 PM)
- Continuous Integration and Testing for SD-WAN with Ansible - BRKDEV-3326 (04:45 PM - 06:15 PM)
- GitHub and Continuous Integration (CI) for Python Network Developers - DEVNET-2315 (05:00 PM - 05:45 PM)

Thursday, January 30

- DevNet Workshop: Utilizing Cisco CXTA service framework to validate network elements - DEVWKS-1407 (11:00 AM - 11:45 AM)
- DevNet Workshop: SD-WAN DevOps Step 3: Continuous Integration/Continuous Deployment - DEVWKS-2030 (10:00 AM - 10:45 AM)
- DevNet Workshop: SD-WAN DevOps Step 1: Automating Test Environments - DEVWKS-2226 (02:00 PM - 02:45 PM)
- CICD Pipelines for Cisco's IoT Edge compute platforms - - DEVNET-1559 (04:00 PM - 04:45 PM)

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on Thursday) to receive your Cisco Live t-shirt.
- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

Take the time to automate

Thank you

CISCO

You make **possible**