DevOps

-- Chidanand

Program Agenda

- Please Refer to Word document that was circulated earlier

Introductions

What is DevOps?

What is DevOps?

"It's a movement of people who think its time for change in the IT industry – time to stop wasting money, time to start delivering great software and building systems that scale and last" **– Patrick Debois**

The DevOps movement, unlike Agile, lacks a manifesto!!

DevOps Definition

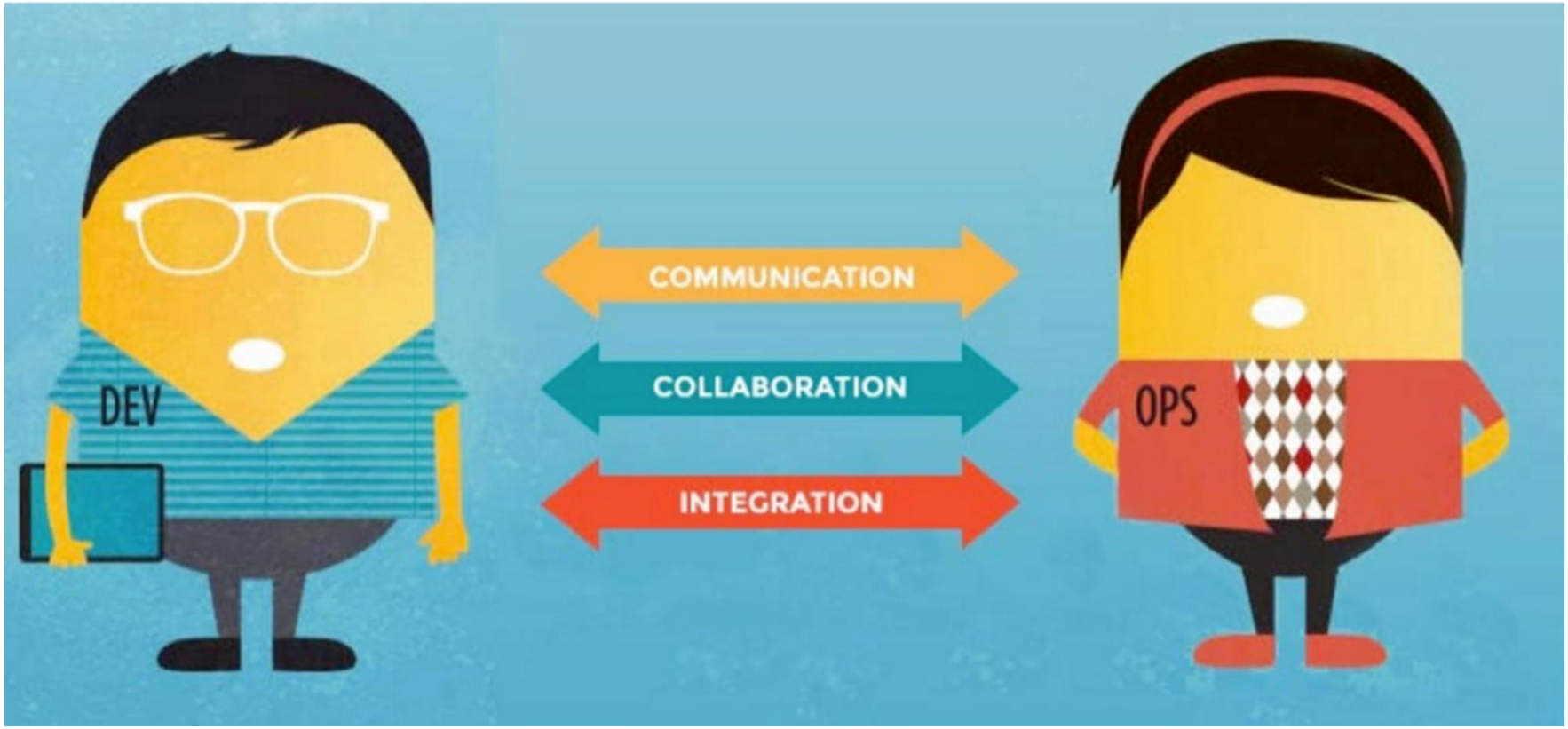
As defined by **Bass, Weber, and Zhu**

"DevOps is a set of practices intended to reduce the time between committing a change to a system and the change being placed into normal production, while ensuring high quality"

DevOps – Wiki Defintion

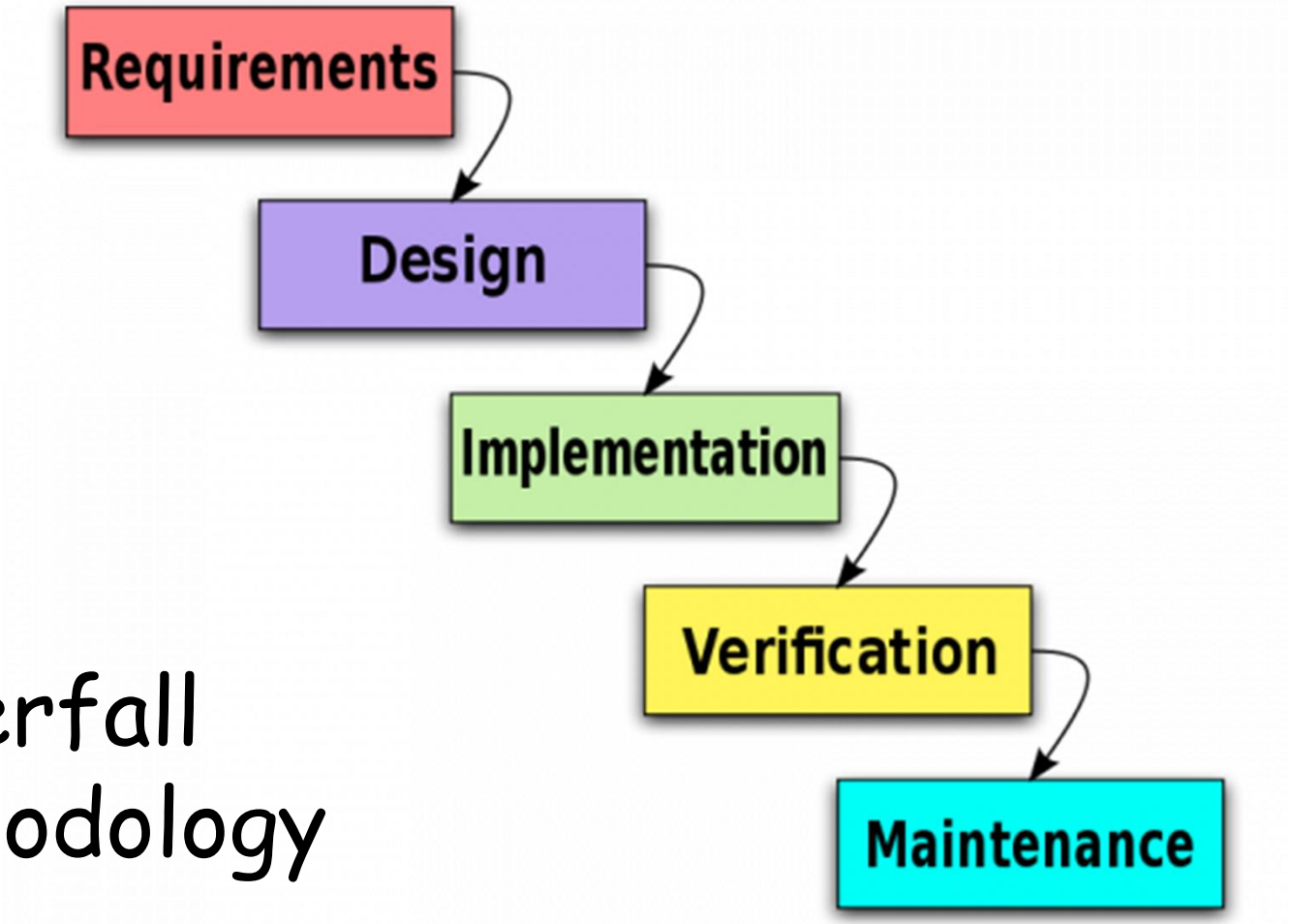
- Software Engineering culture and practice that aims at unifying software development (Dev) and software operation (Ops)
- Main characteristic is to use Automation & Monitoring at every step of software construction
- Shorter Dev Cycles, increased deployment frequency and more dependable releases

DevOps Philosophy



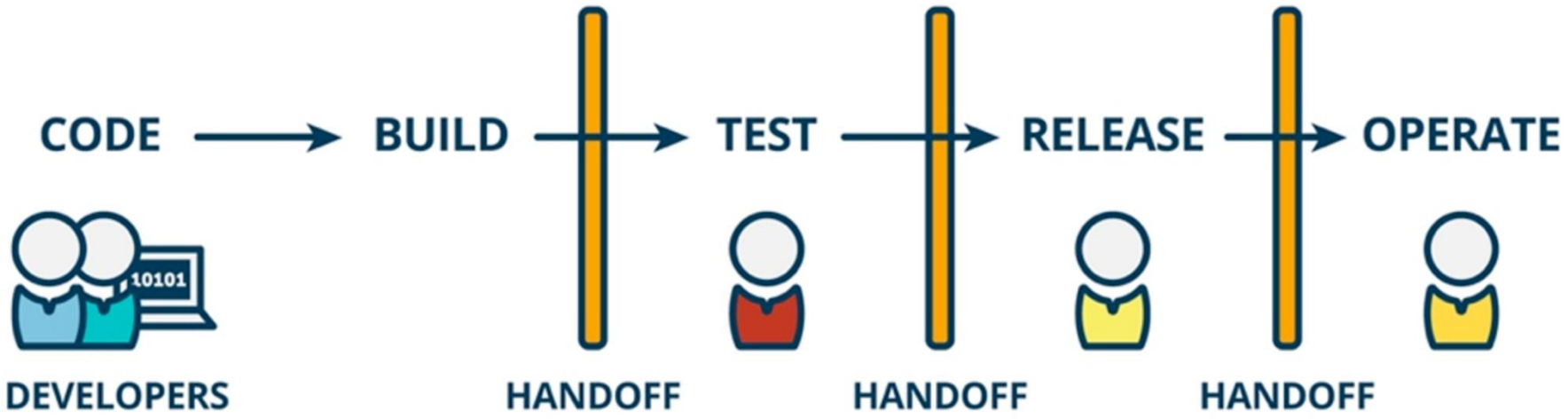
DevOps is the way in which a technology organization embeds itself in a business to the benefit of that business.

Once Upon a Time



Waterfall
Methodology

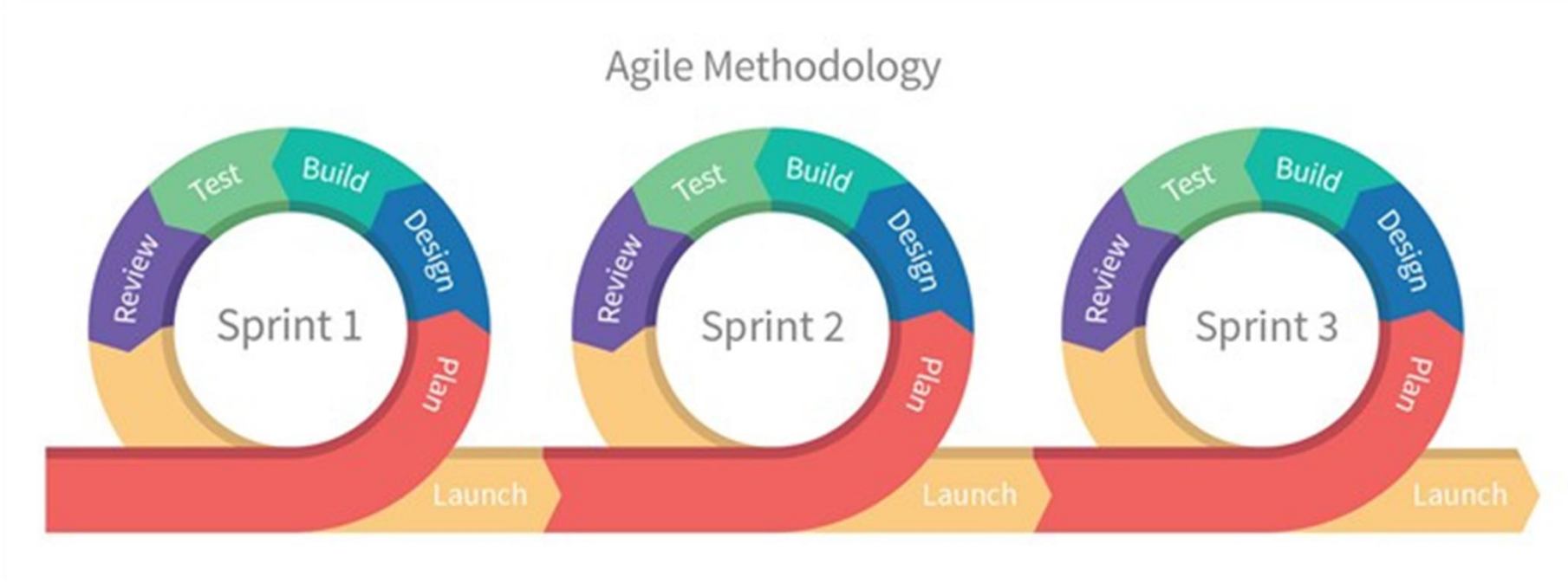
Once Upon a Time



Waterfall SDLC

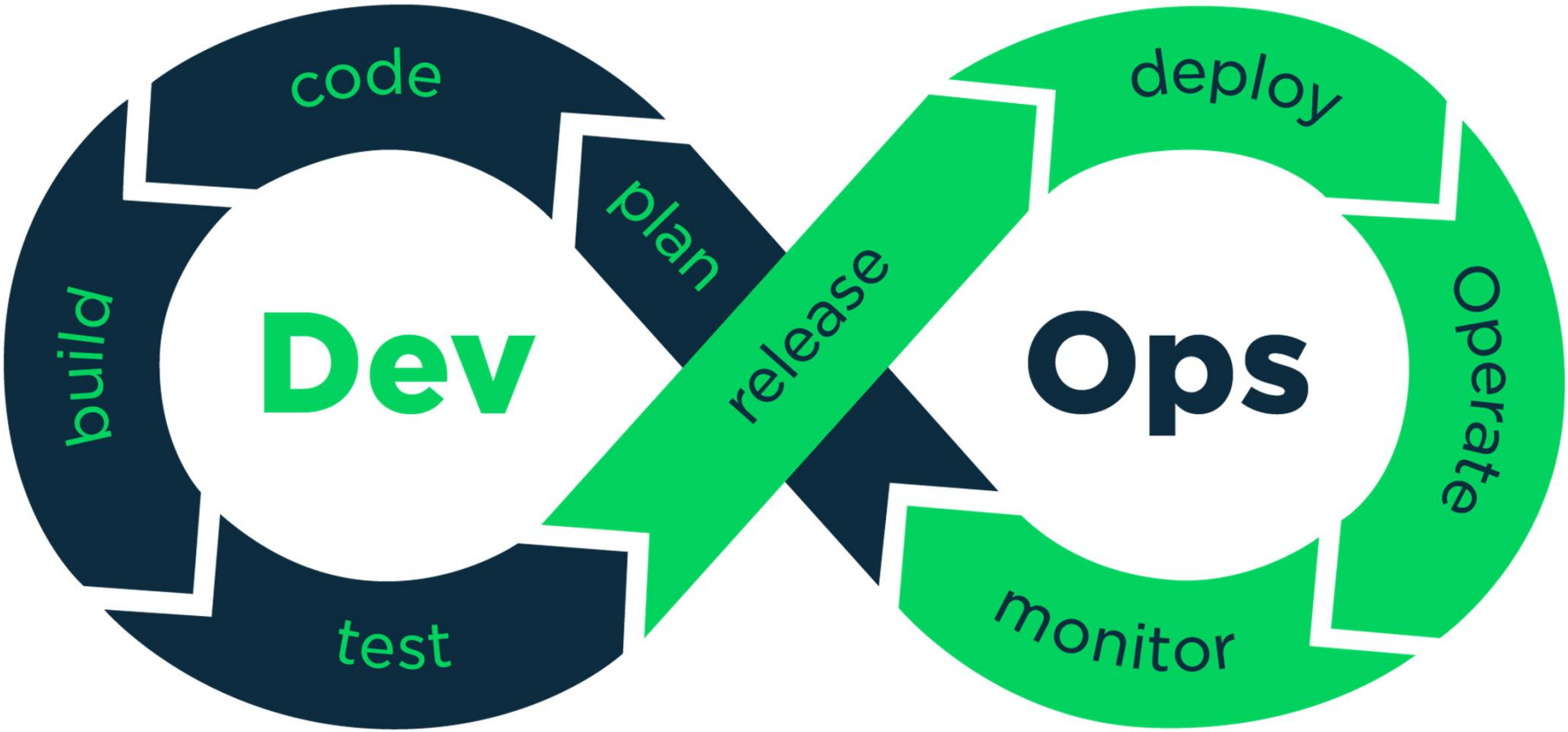
http://www.mindtheproduct.com/2016/02/what-the-hell-are-ci-cd-and-devops-a-cheatsheet-for-the-rest-of-us/

Not long ago….



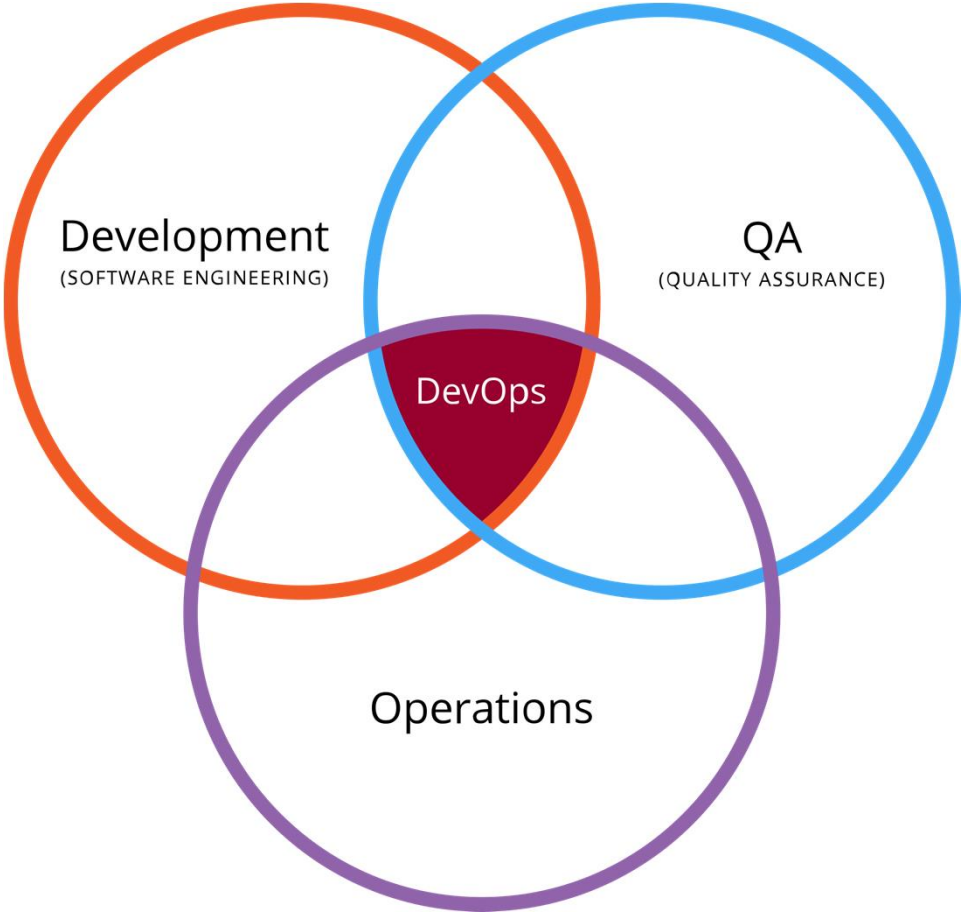
Agile & Incremental prototyping

Now (2018).....



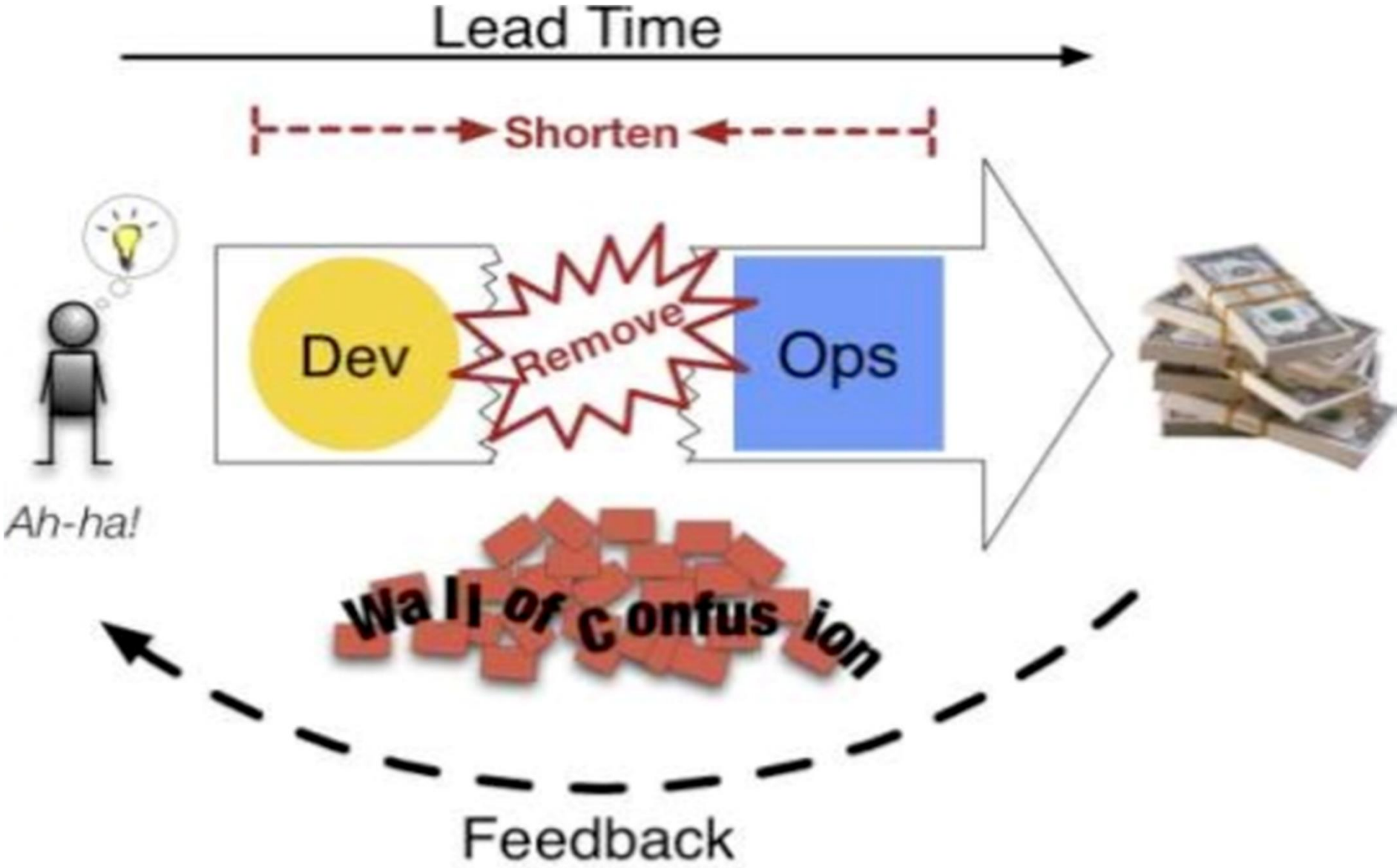
DevOps – Dev + Operations

Now (2018).....



DevOps – Dev + QA+ Operations

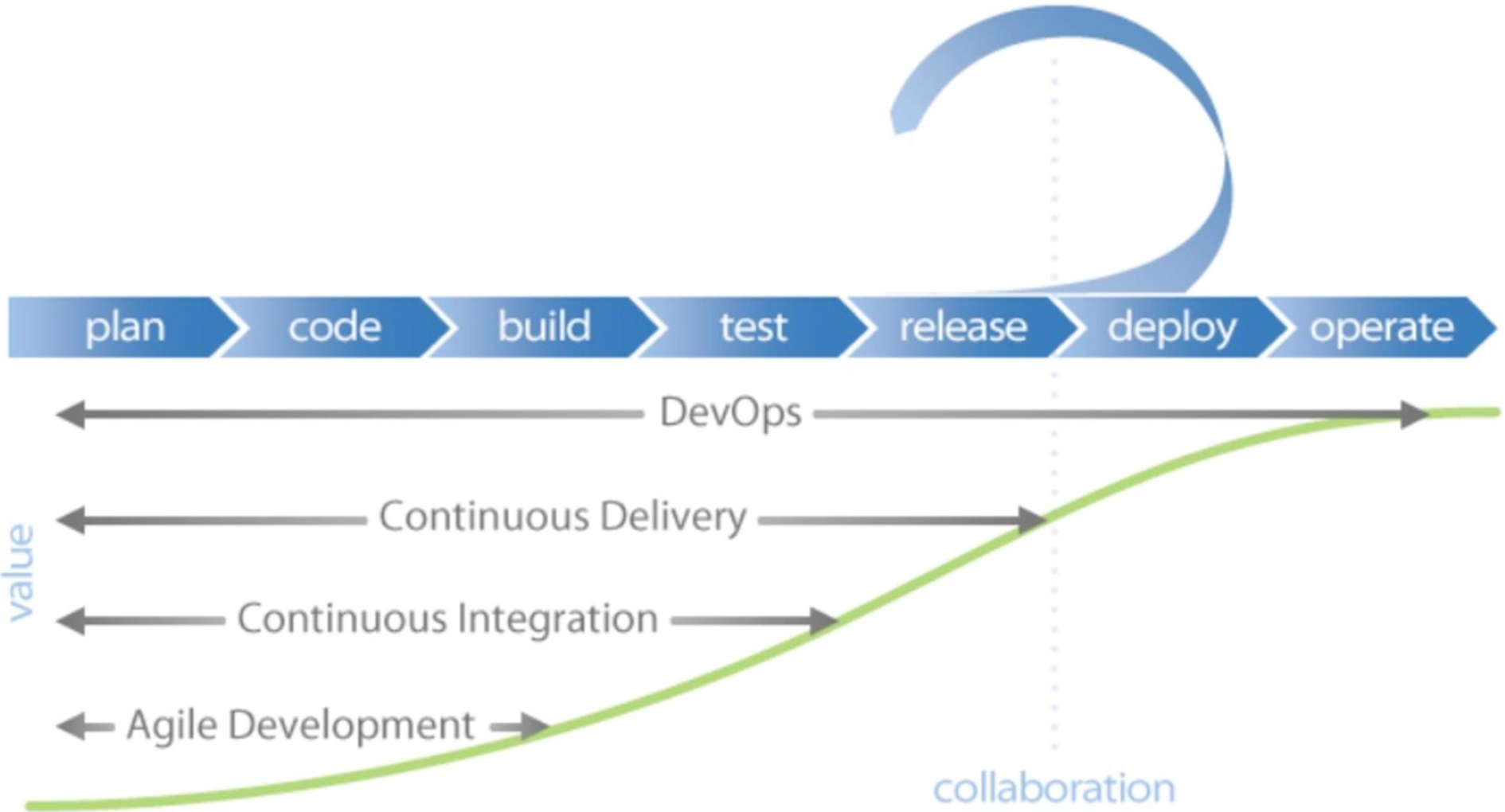
Why DevOps?

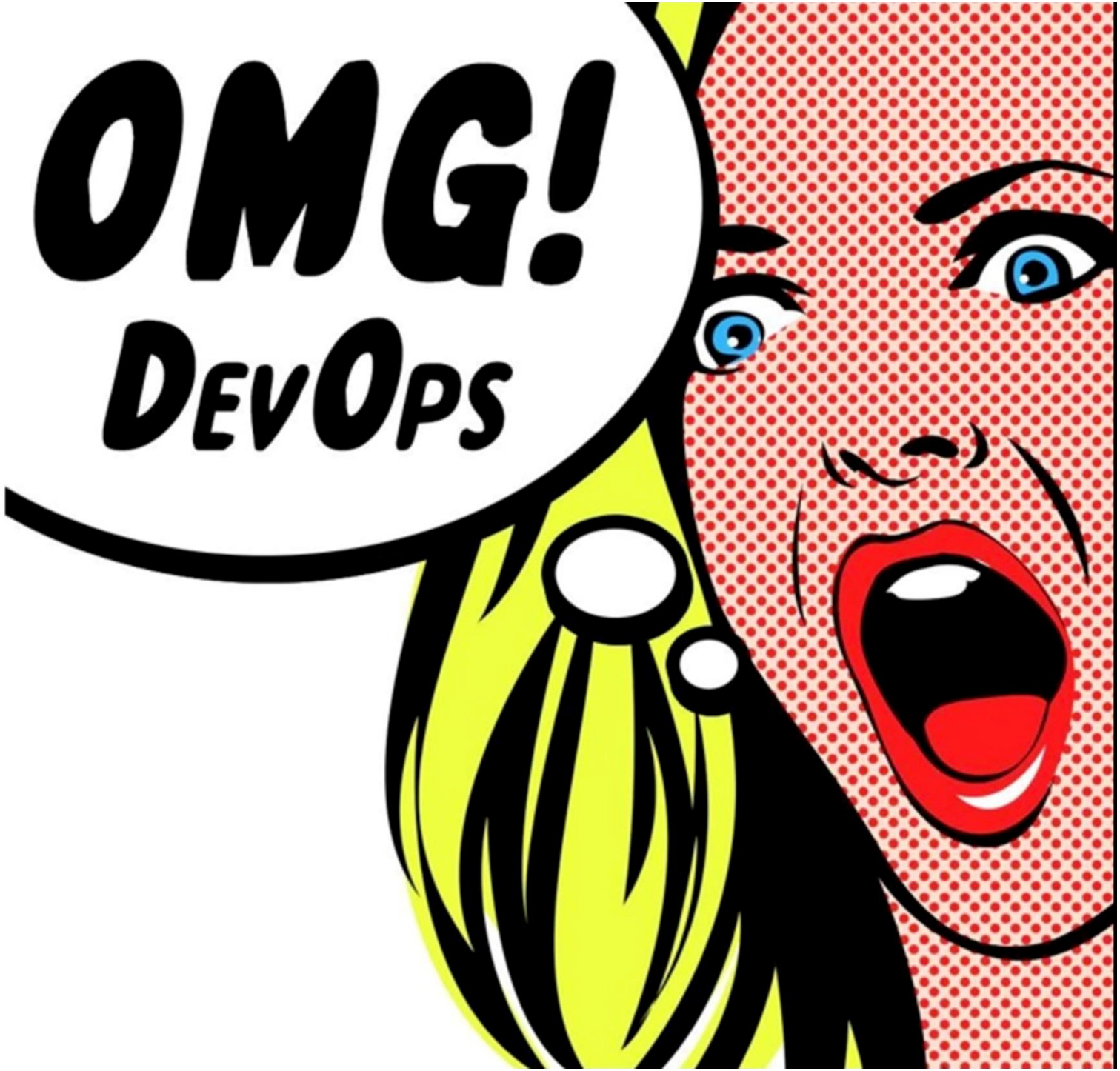

Lead Time
Shorten
Dev
Remove
Ops
Ah-ha!
Wall of Confusion
Feedback


WORKED FINE IN DEV...
...OPS PROBLEM NOW
memegenerator.net


WORKS
ON MY
MACHINE

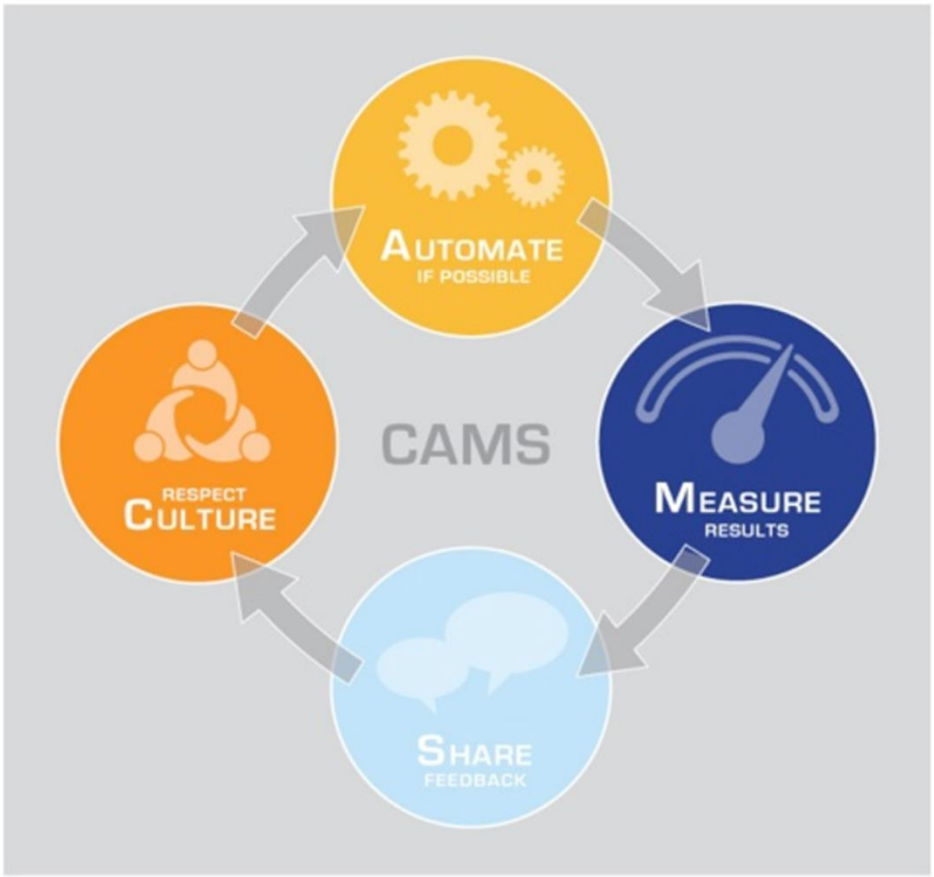
Continuous…..


plan
code
build
test
release
deploy
operate
DevOps
Continuous Delivery
Continuous Integration
Agile Development
value
collaboration


OMG!
DevOps

How to start DevOps?

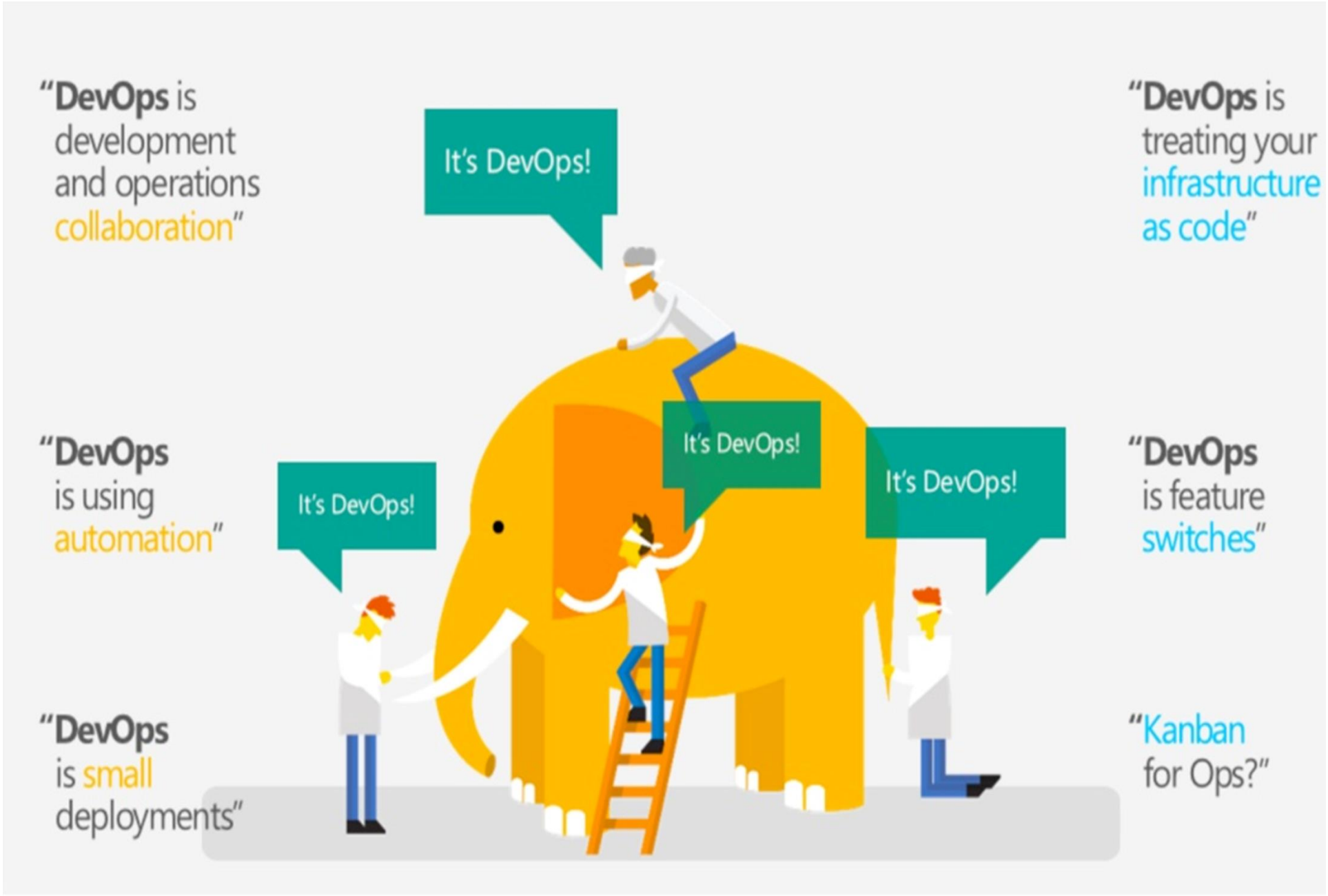
DevOps Principles



http://www.telehouse.com/2016/03/devops-how-a-culture-of-empathy-creates-massive-productivity/

DevOps Principles

Culture => People, Process, Tools
Automation => Infrastructure as Code
Measurement => Measure everything
Sharing => Collaboration/Feedback



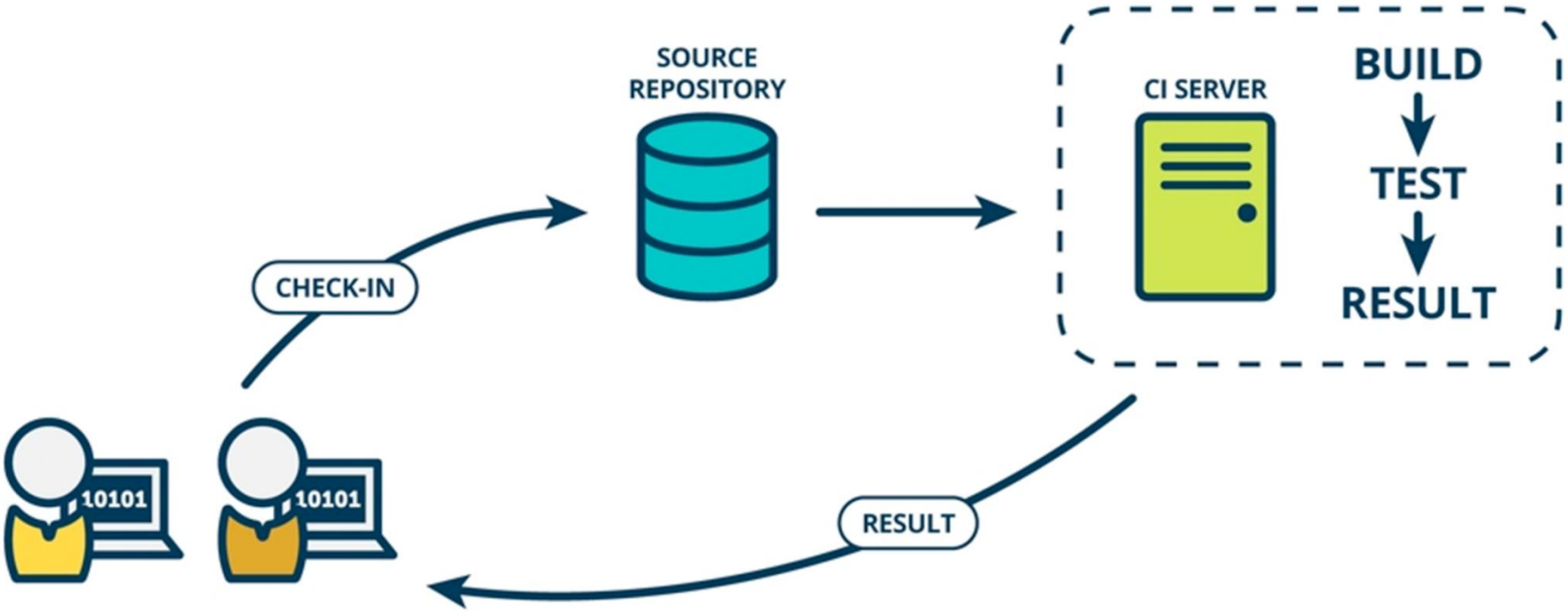
https://channel9.msdn.com/Series/DevOps-Fundamentals/Introduction-to-DevOps

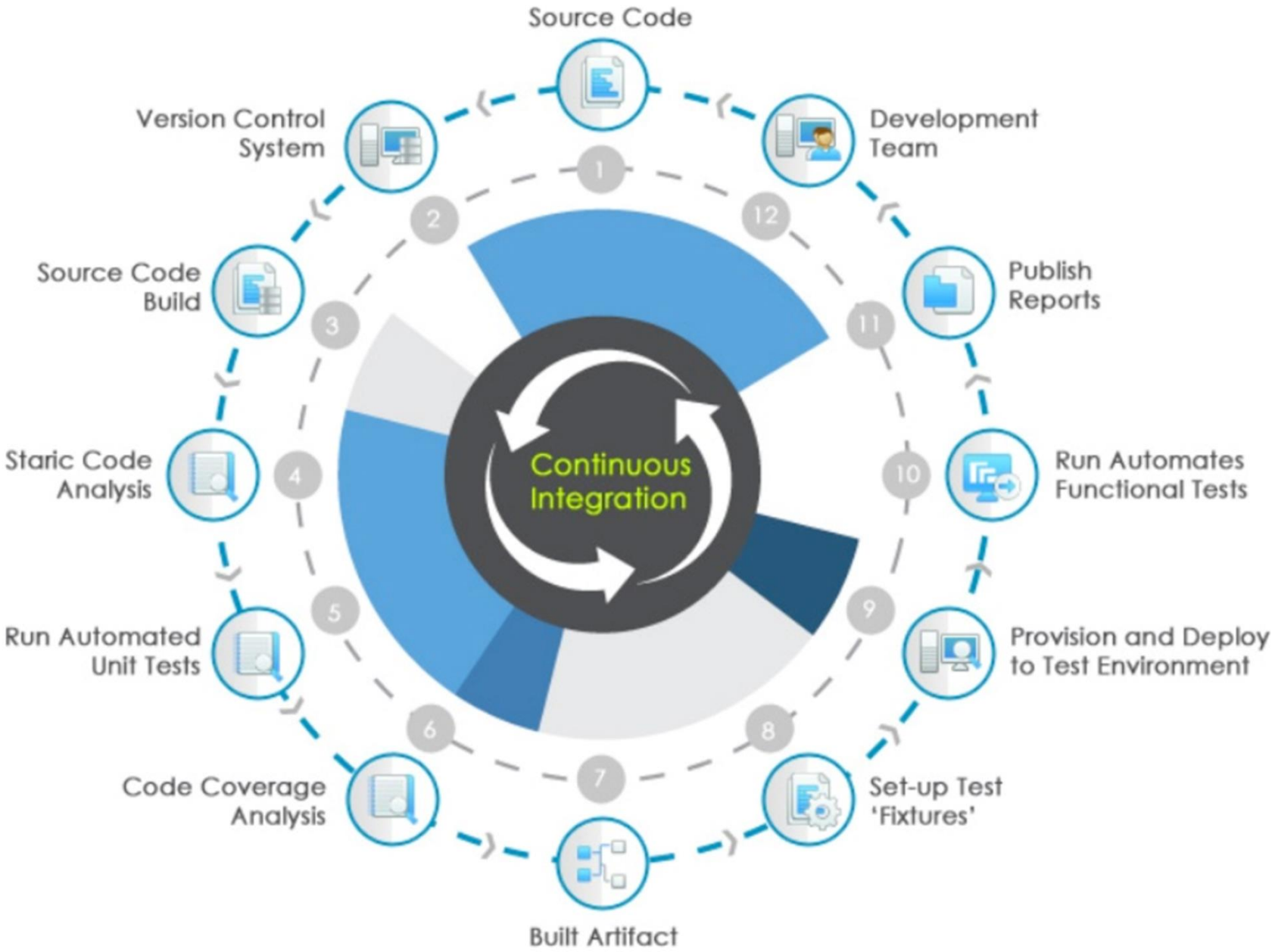
Continuous Integration

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly. –

Martin Fowler

Continuous Integration




Source Code
Development Team
Publish Reports
Run Automates Functional Tests
Provision and Deploy to Test Environment
Set-up Test 'Fixtures'
Built Artifact
Code Coverage Analysis
Run Automated Unit Tests
Staric Code Analysis
Source Code Build
Version Control System
Continuous Integration
1
2
3
4
5
6
7
8
9
10
11
12

Jenkins

Bamboo

TeamCity

go™

Visual Studio
Team Foundation Server

Hudson

Travis

wercker

circleci

CI is about what people do

not about what tools they use

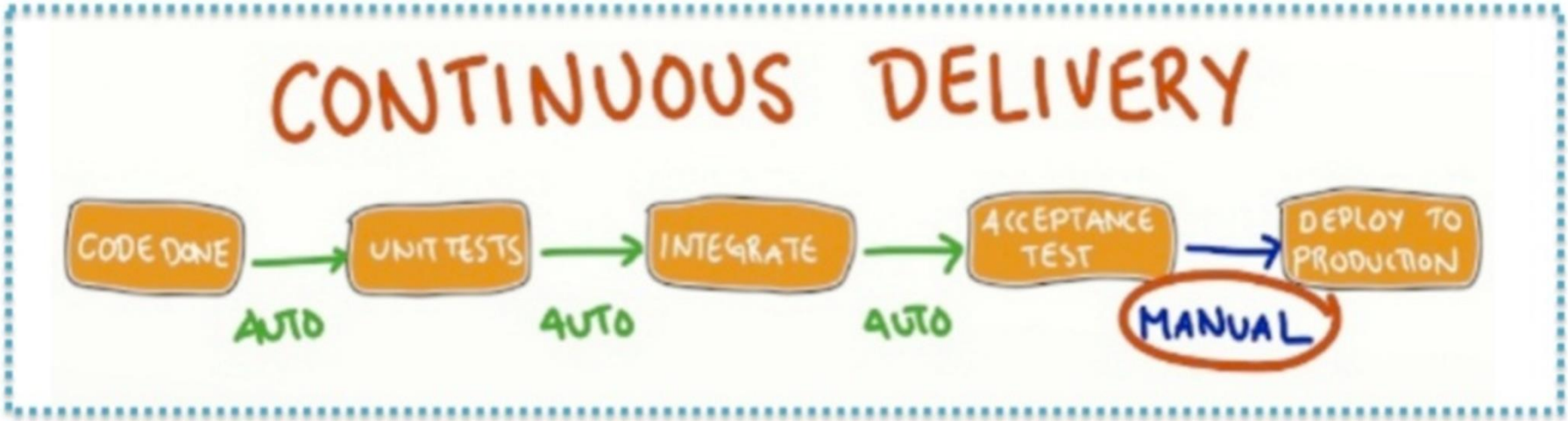
Continuous Delivery

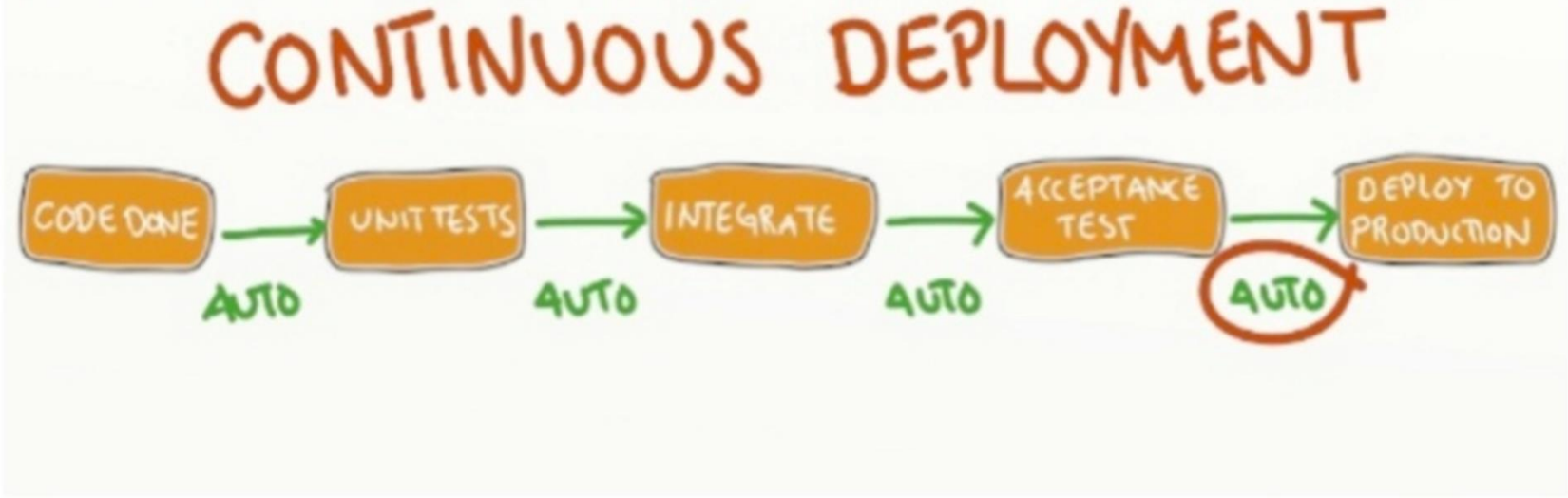
Continuous delivery (**CD**) is a Software Engineering approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time

Building, testing & Releasing software more faster & more frequently

Continuous Deployment

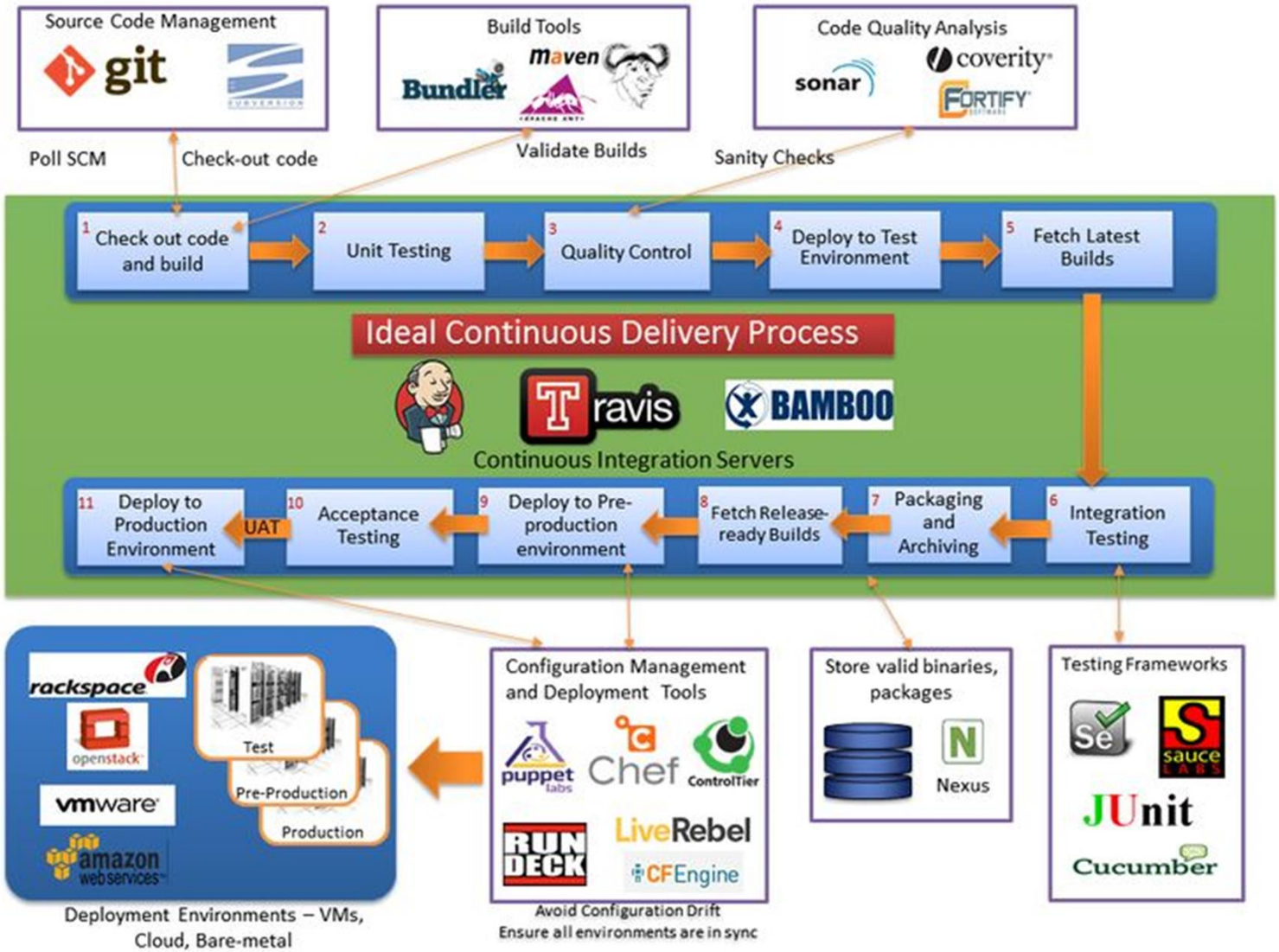
Strategy for software releases wherein any code commit that passes the **automated** testing phase is automatically released into the production environment, making changes that are visible to the software's users.





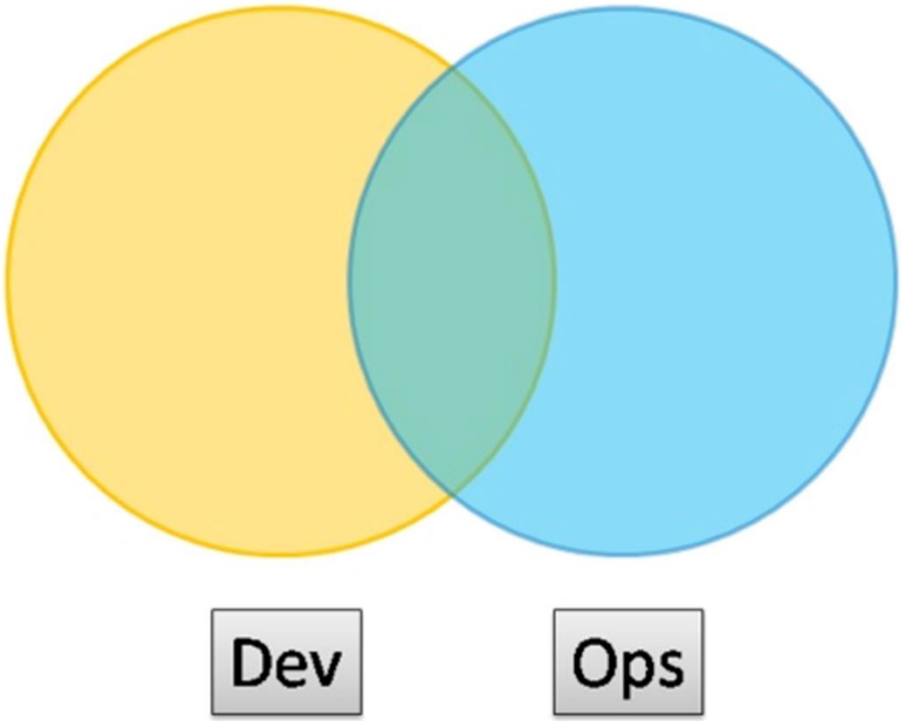
http://blog.crisp.se/2013/02/05/yassalsundman/continuous-delivery-vs-continuous-deployment

Delivery Pipeline



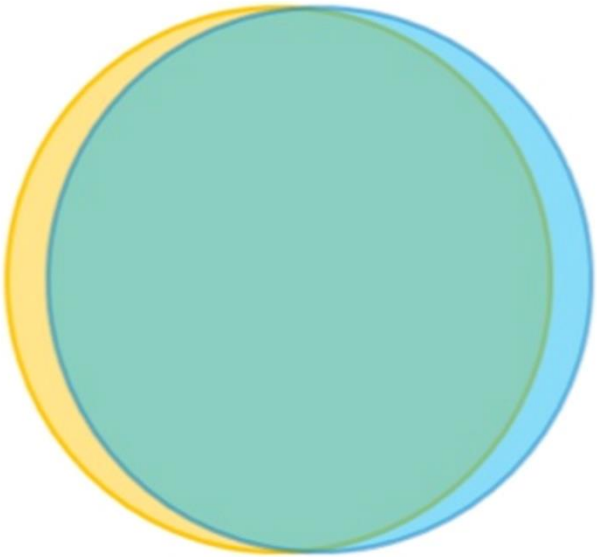
DevOps Team Topologies

Type 1 – Smooth Collaboration



https://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

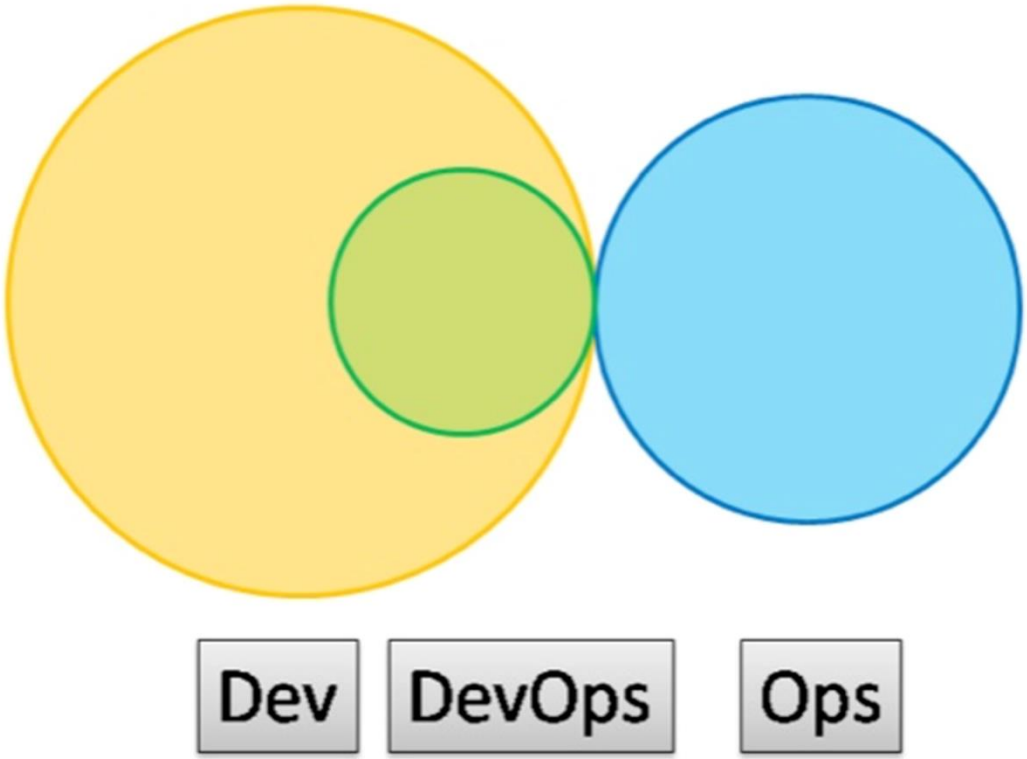
Type 2 – Fully Embedded



Dev | Ops

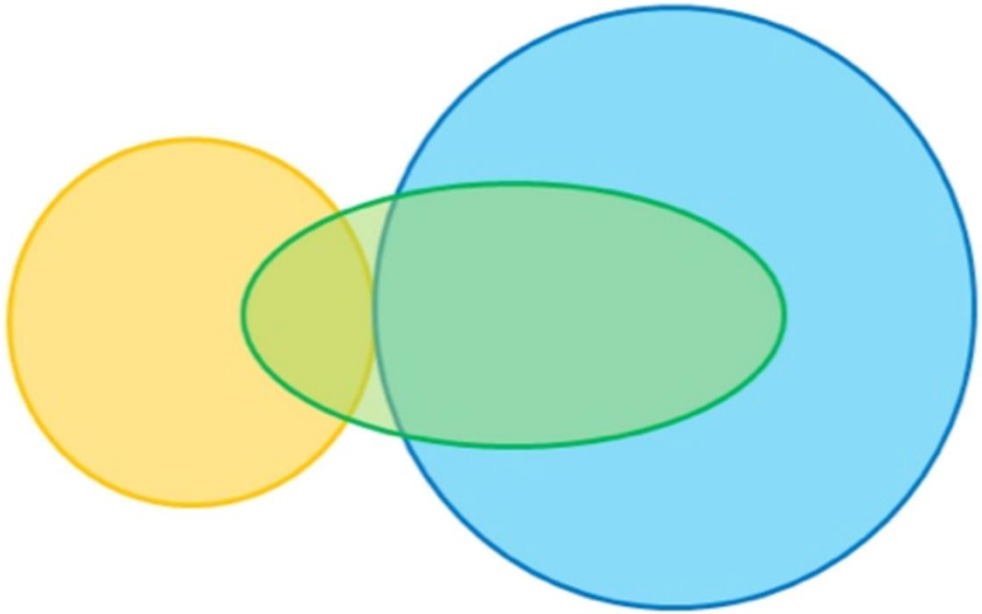
https://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

Type 3 – Infrastructure-as-a-Service



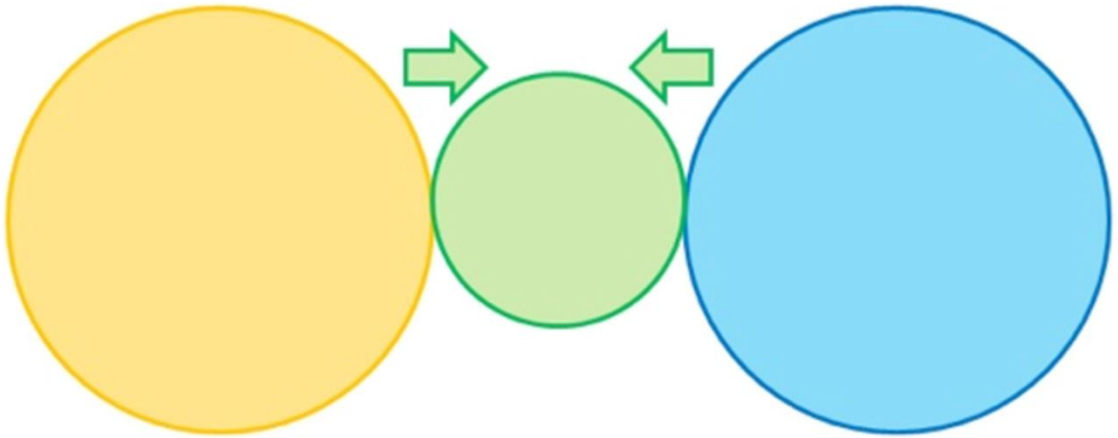
https://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

Type 4 – DevOps-as-a-Service



https://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

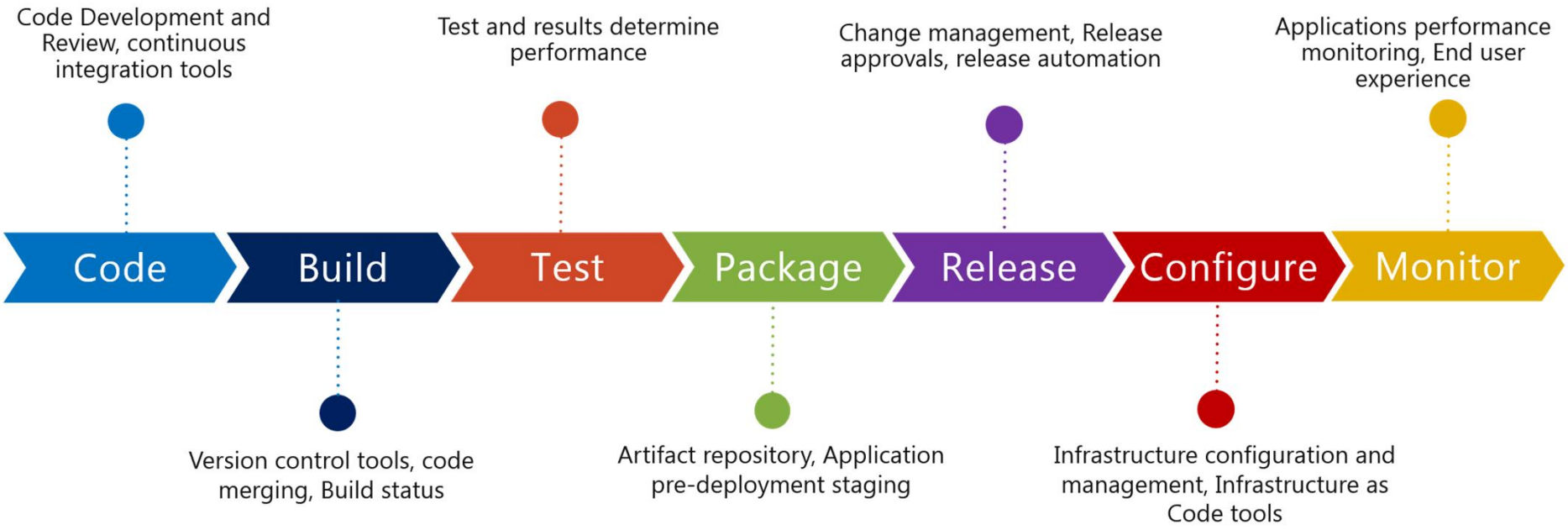
Type 5 – Temporary DevOps Team

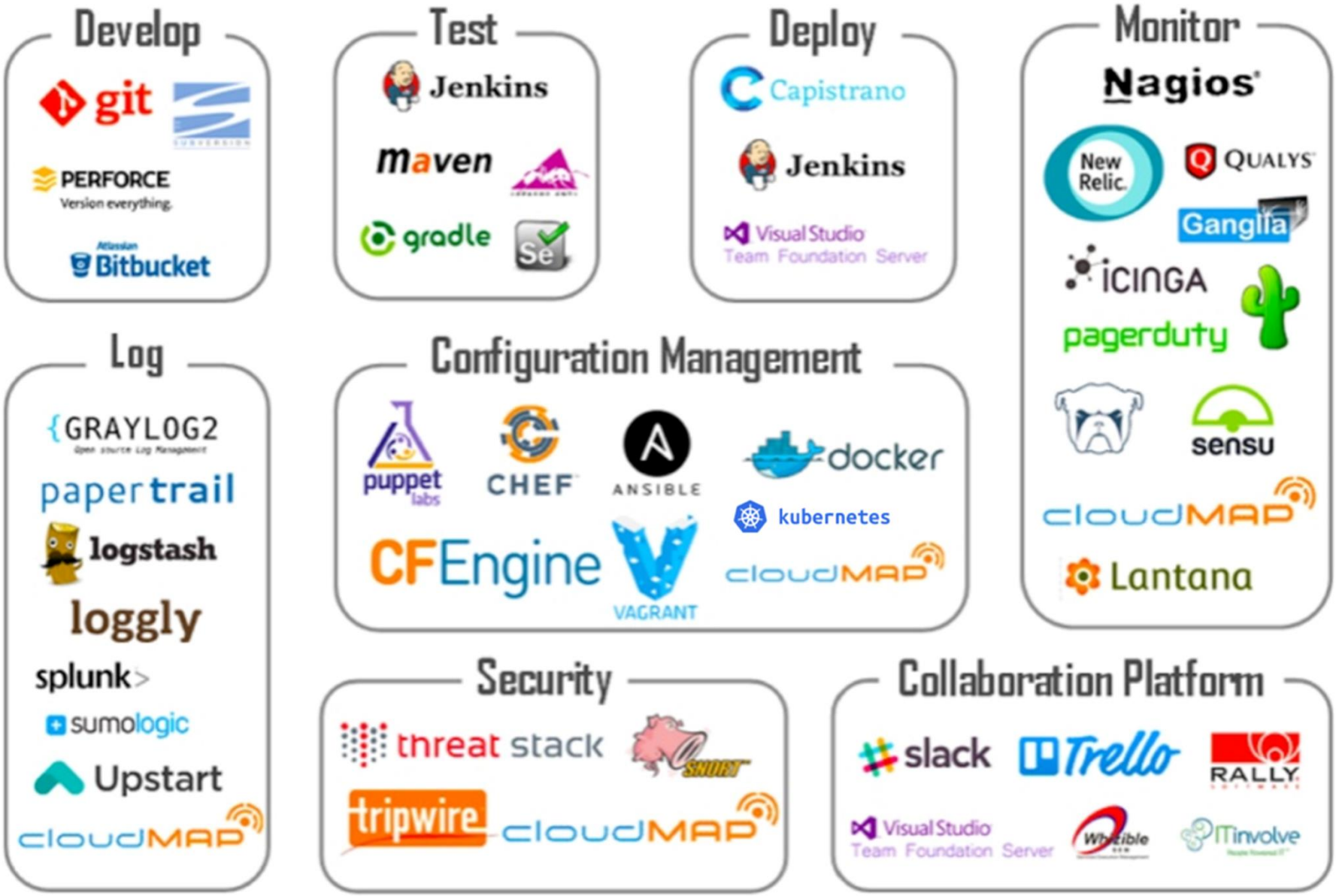


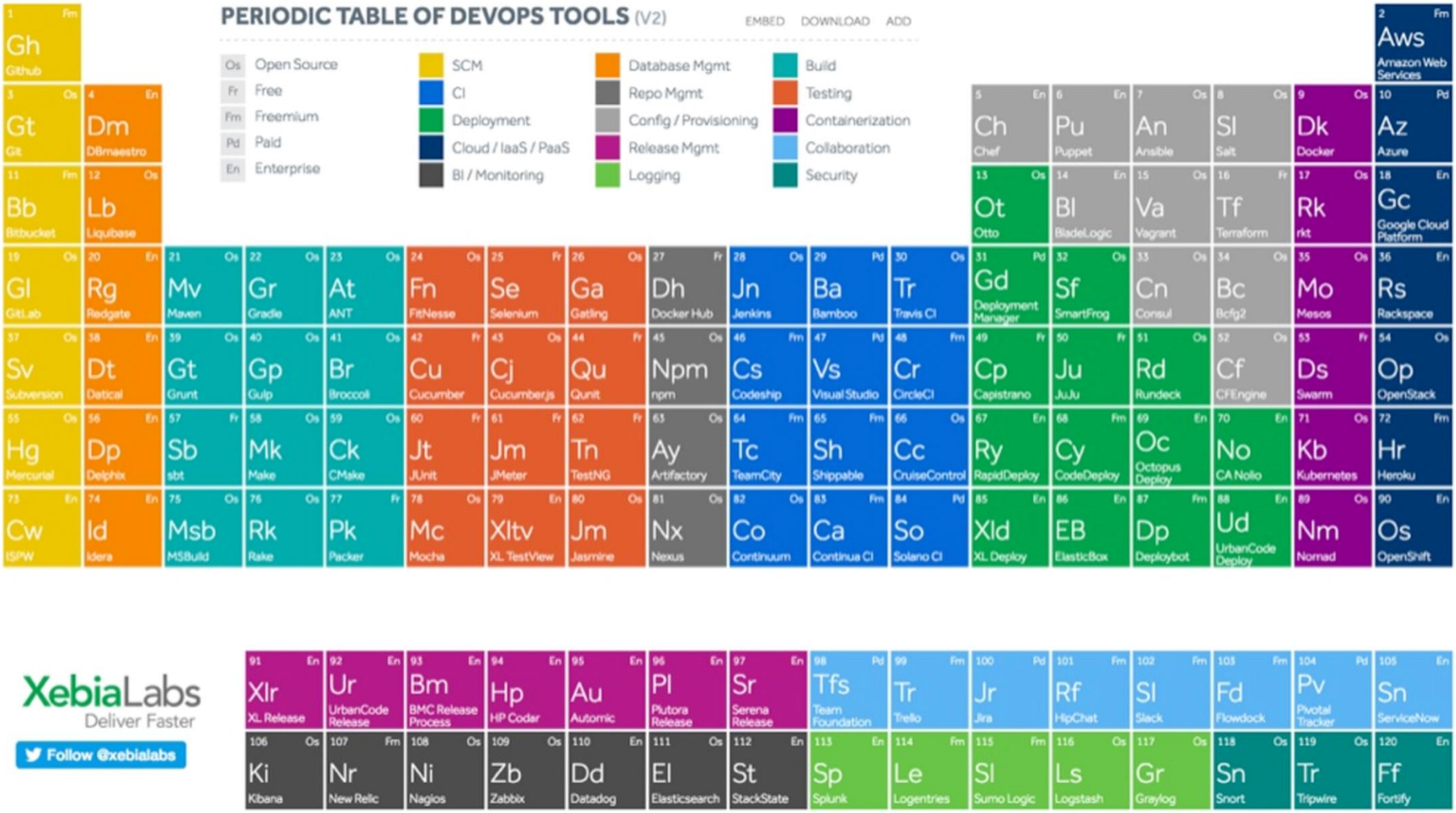
https://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

DevOps Tools ?

DevOps Toolchain




Develop
git
PERFORCE
Version everything.
Atlassian
Bitbucket
Test
Jenkins
maven
gradle
Deploy
Capistrano
Jenkins
Visual Studio
Team Foundation Server
Monitor
Nagios
New Relic.
QUALYS
Ganglia
ICINGA
pagerduty
sensu
cloudMAP
Lantana
Log
{GRAYLOG2
Open source Log Management
papertrail
logstash
loggly
splunk>
sumologic
Upstart
cloudMAP
Configuration Management
puppet
labs
CHEF
ANSIBLE
docker
kubernetes
CFEngine
VAGRANT
cloudMAP
Security
threat stack
SNORT
tripwire
cloudMAP
Collaboration Platform
slack
Trello
RALLY
Visual Studio
Team Foundation Server
Whitible
ITinvolve



XebiaLabs
Deliver Faster

Follow @xebialabs

https://xebialabs.com/periodic-table-of-devops-tools/

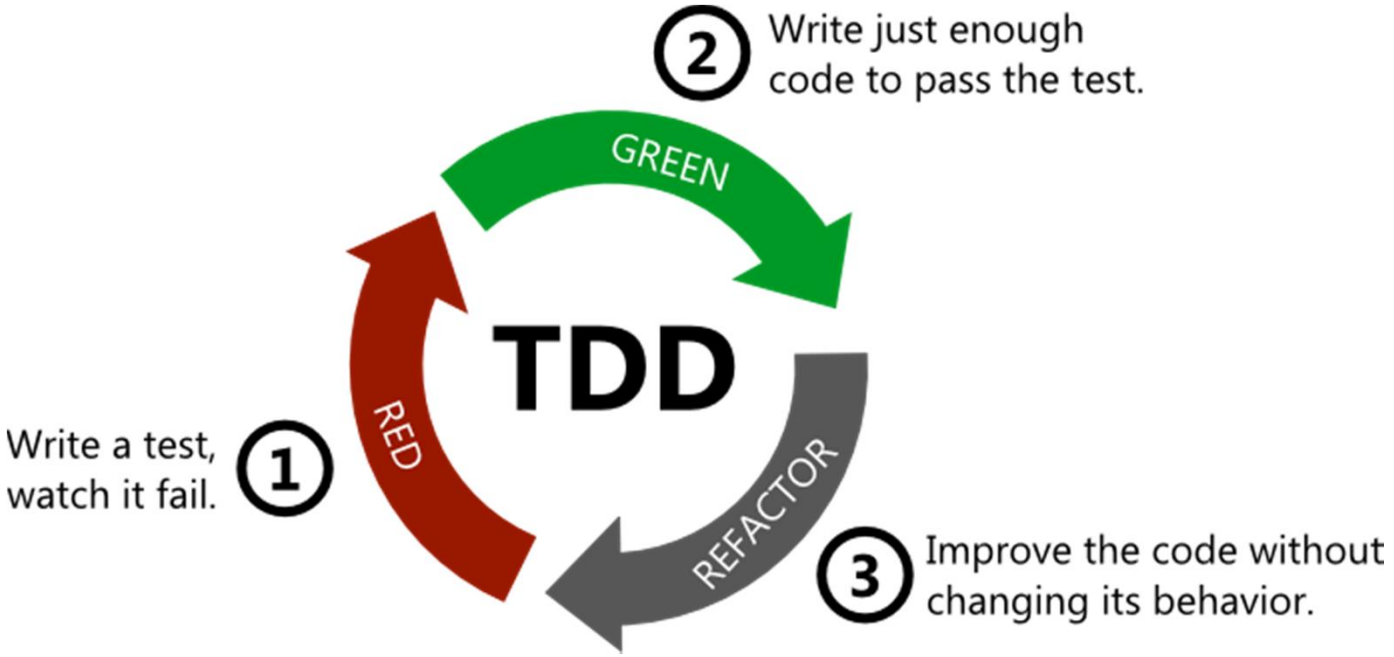
QA – Forgotten Hero

- Importance of Testing & Automation
- % of Automation
- Selenium, TestNG, JUnit
- Test Driven Development (TDD)

TDD Origins

- Originates from Extreme Programming
- Goals are to deliver clean well tested code
- Unit testing is paramount
- Try to produce minimum amount of code to deliver results
- Refactoring is strongly encouraged
- Work is performed in tight development cycles or Sprints

TDD Philosophy

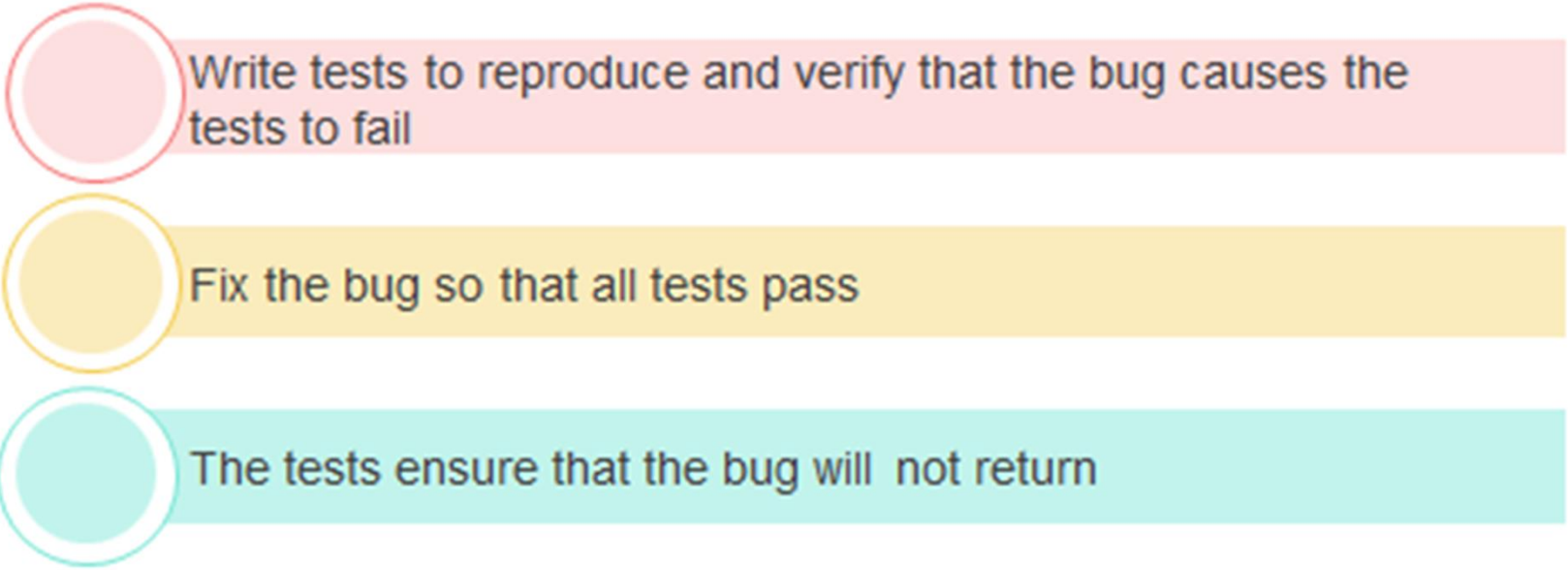


Why TDD

- Produces thoroughly tested code
- High Test Coverage and low regression
- With high test coverage, refactoring is safe
- Code tends to be simplified and focused
- Tests are maintained along with code

When to use TDD

- New Projects
- Legacy Applications
- When Requirements are not clear
- TDD is good for bug fixing



DevOps KPIs

Key Performance Indicators

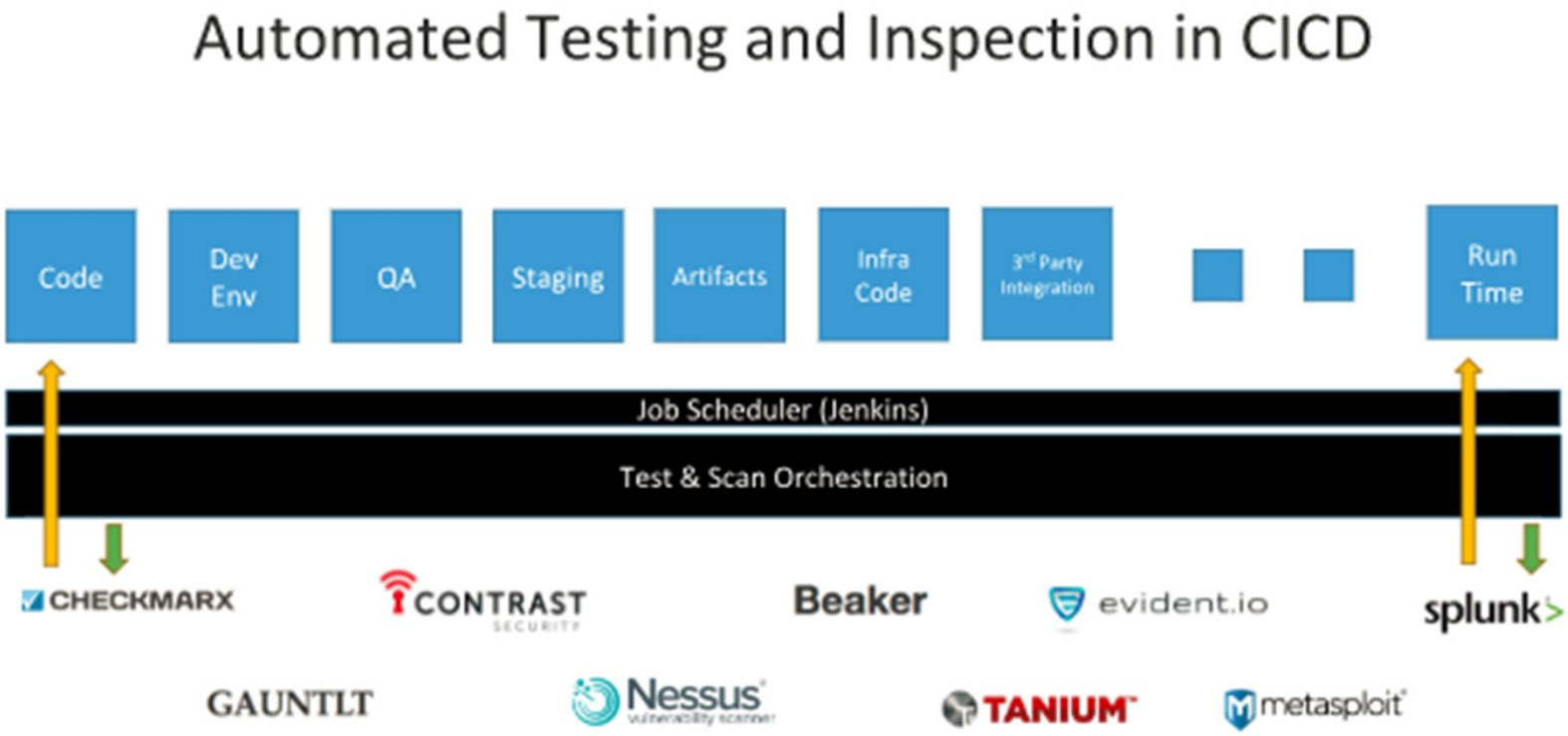
- Deployment Frequency
- Deployment Speed
- Failure Rate
- Time to Recovery

DevSecOps

- DevOps transformation with baked in Security at each & every level

DevSecOps

- Security as Code



DevSecOps Implementation

- Shift Security Left – Use CI/CD Pipeline to embed security
- Self Service – Give Developers & Ops visibility into Security activities
- Security Champions – Pick & Encourage few folks to pick Security tasks

DevSecOps Implementation

- Everything as Code (EAC) – Compliance as Code & hardening via configuration management system
- Use Secure by default frameworks

Questions?