Modern culture to sustainably improve software quality

exploiting automated build, test and delivery schema

Kelvin Kühn, Claudius Stern

FOM University of Applied Sciences
for Economics and Management

Prof. Dr. Claudius Stern

Short resume

Station (1999 – 2007):

Studies of Computer Science at the University of Paderborn

Station (2007 – 2014):

Doctorate at the University of Paderborn
in the field of embedded systems

Research projects:
Approaches to location-based mobile services
Approaches to the digitization of primary care

Station (2007 – today):

Own company for the development of embedded systems

Station (2013 – 2018):

Head of embedded systems development , biozoom services GmbH, Kassel, Germany

Station (2018 – today):

Professor at FOM University of Applied Sciences, Kassel, Germany

Research Area: Embedded Systems

About the authors

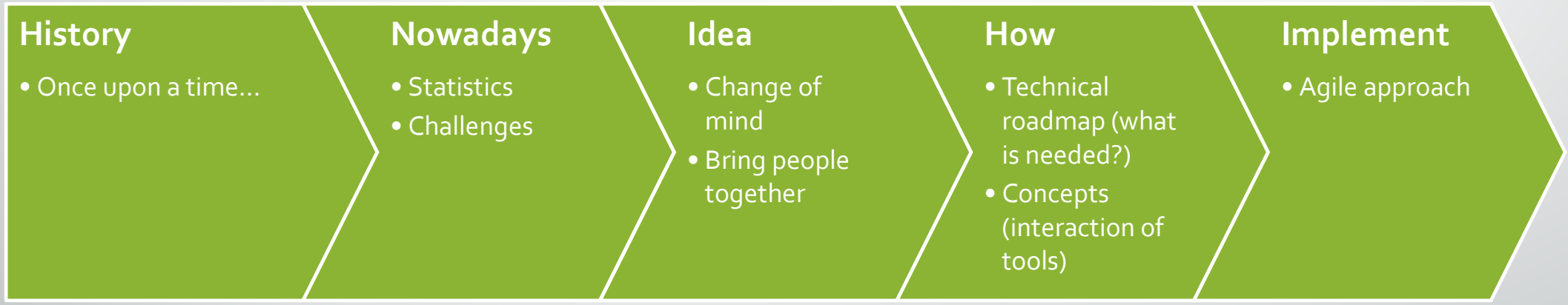
Kelvin Kühn

- B.Sc. Business Informatics
- IT specialist
- Bachelor degree at FOM University of Applied Sciences, Kassel, Germany
- Business Informatics, IT department, bdks, Baunatal, Germany

Prof. Dr. Claudius Stern

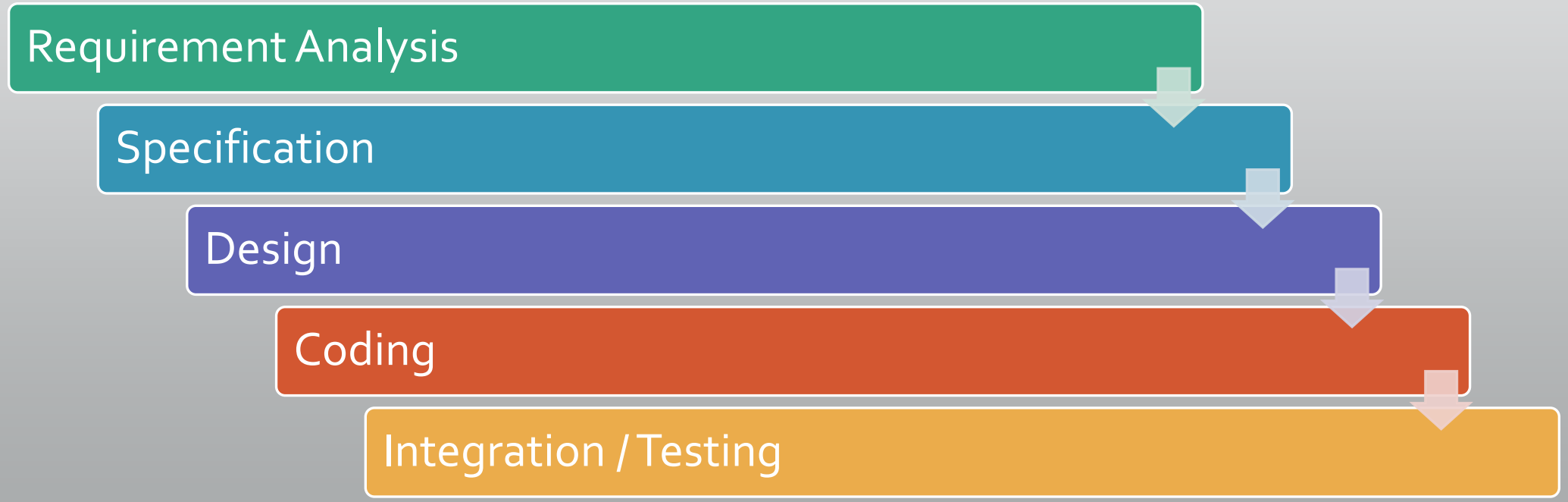
- Professor at FOM University of Applied Sciences, Kassel, Germany
- Head of embedded systems development , biozoom services GmbH, Kassel, Germany
- PhD in computer science, specialized on embedded systems

Agenda


History
• Once upon a time…
Nowadays
• Statistics
• Challenges
Idea
• Change of mind
• Bring people together
How
• Technical roadmap (what is needed?)
• Concepts (interaction of tools)
Implement
• Agile approach

Once upon a time…

Traditional Development



Velocity

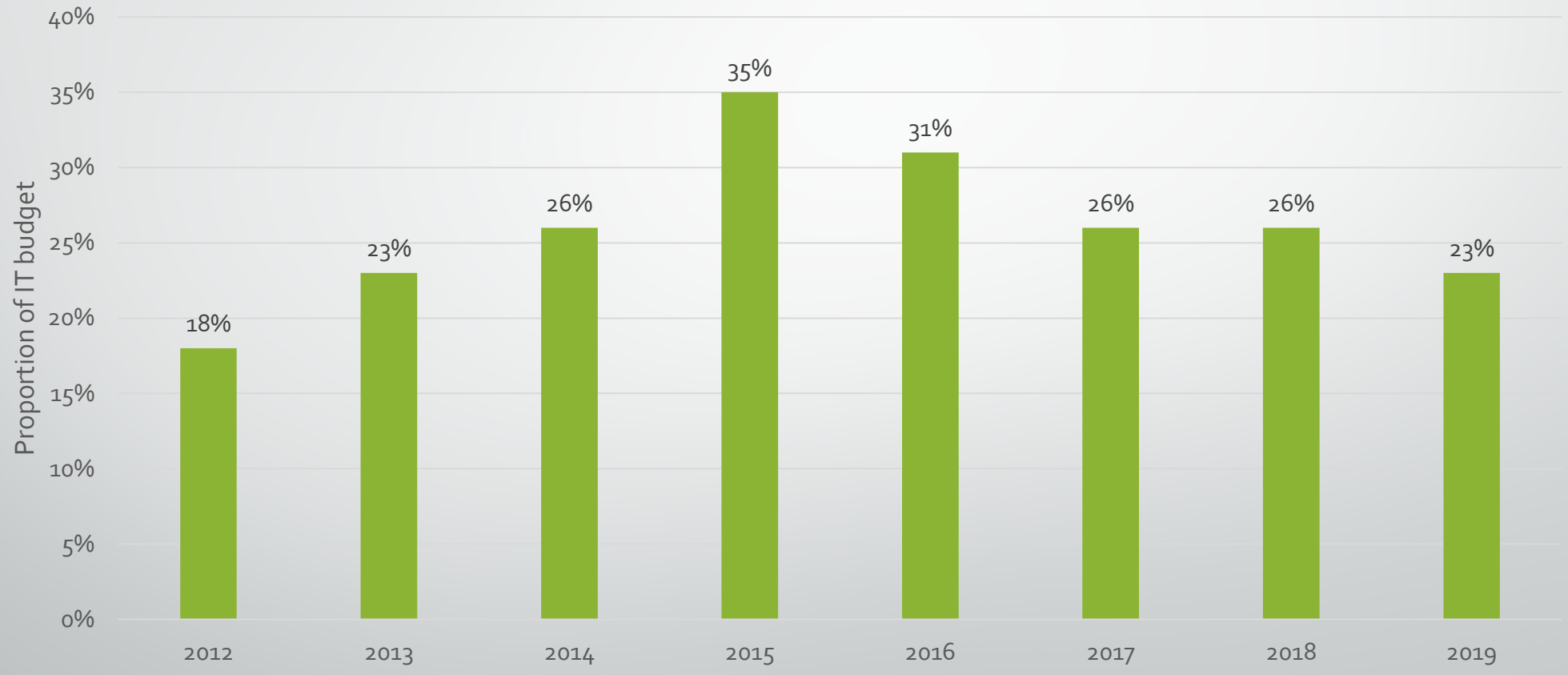
	1970s-1980s	1990s	2000s - Present
Era	Mainframes	Client/Server	Commoditization and Cloud
Typical Technology	COBOL, DB2 on MVS, etc.	C++, Oracle, Solaris, etc.	Java, MySQL, Red Hat, Ruby on Rails, PHP, etc.
Cycle time	1-5 years	3-12 months	2-12 weeks
Cost	$1M - $100M	$100k - $10M	$10k - $1M
At risk	Whole company	A product line or division	A product feature
Cost of failure	Bankruptcy, sell the company, massive layoffs	Revenue loss, CIO‘s job	Negligible

Source: Adrian Cockroft, „Velocity and Volume (or Speed Wins)“, presentation at FlowCon, San Francisco, CA, November 2013

Nowadays…

Proportion of budget allocated to quality assurance and testing as a percentage of IT spend from 2012 to 2019

Quality assurance and testing budget allocation as a share of IT spend 2012-2019

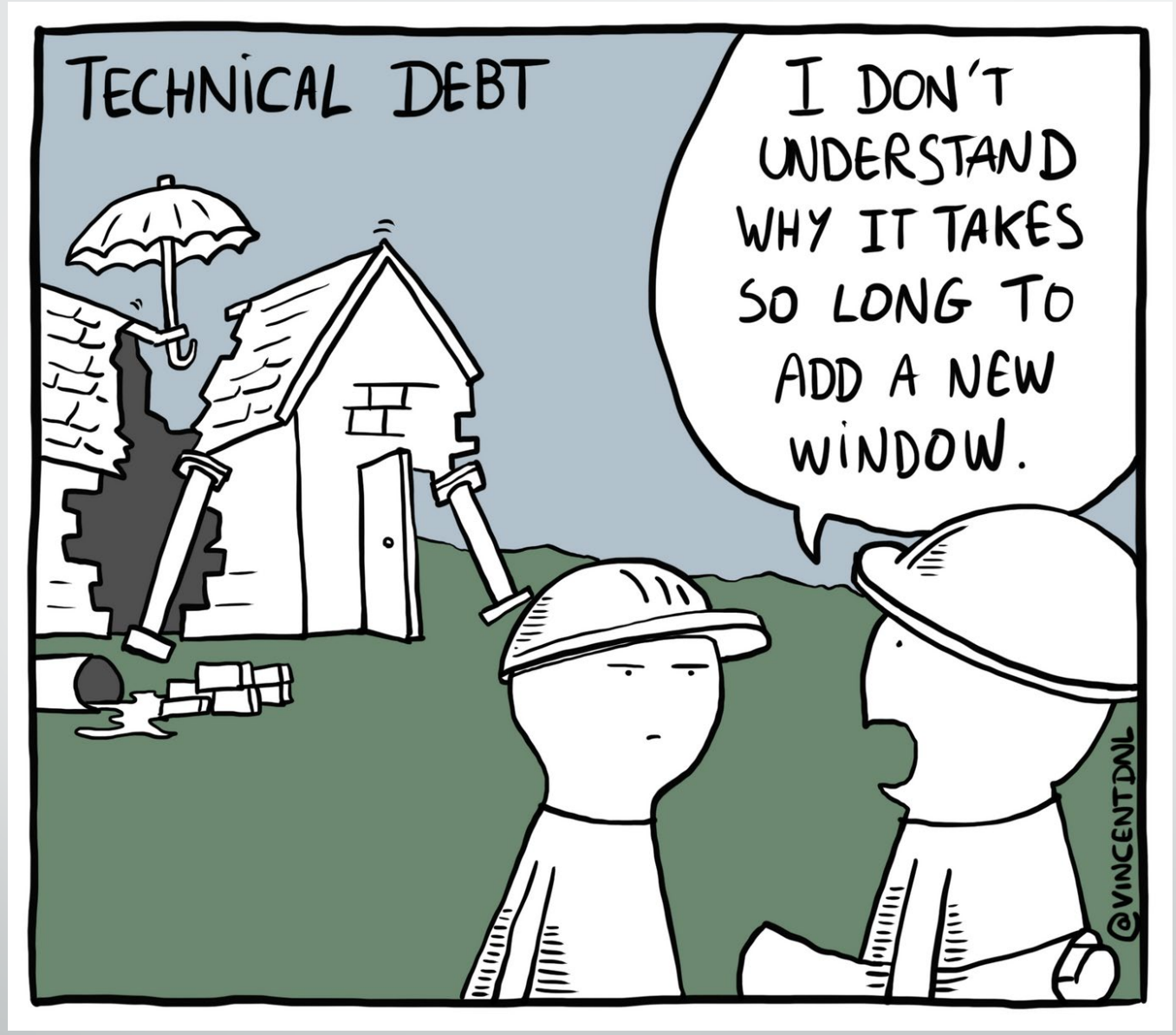


Note(s): Worldwide; 2012 to 2019; 1725 respondents; CIOs and other senior technology professionals
Further information regarding this statistic can be found on page 8.
Source(s): Capgemini; Sogeti; HPE; Micro Focus; ID 500641

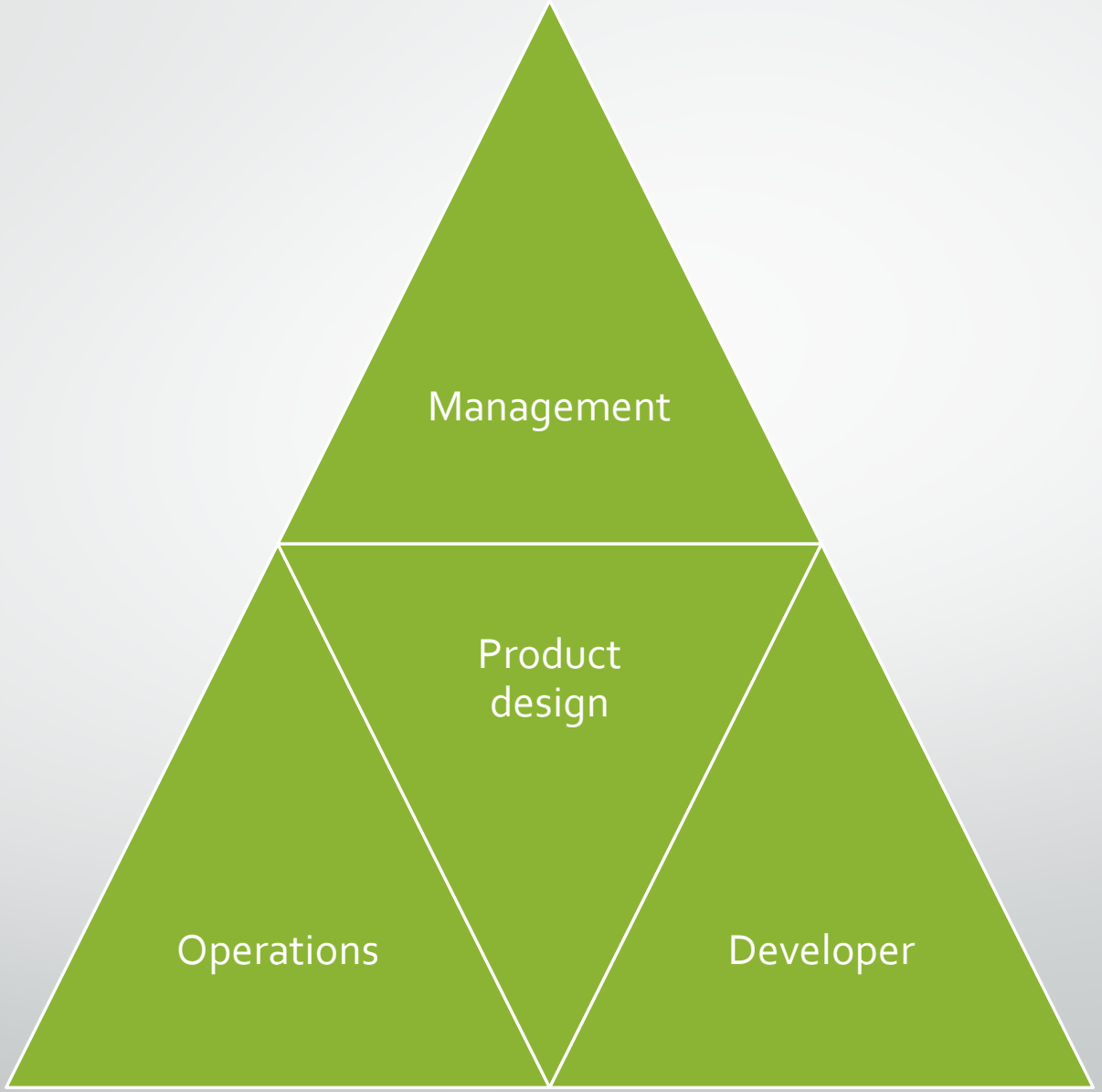
LESS money spent for QUALITY???

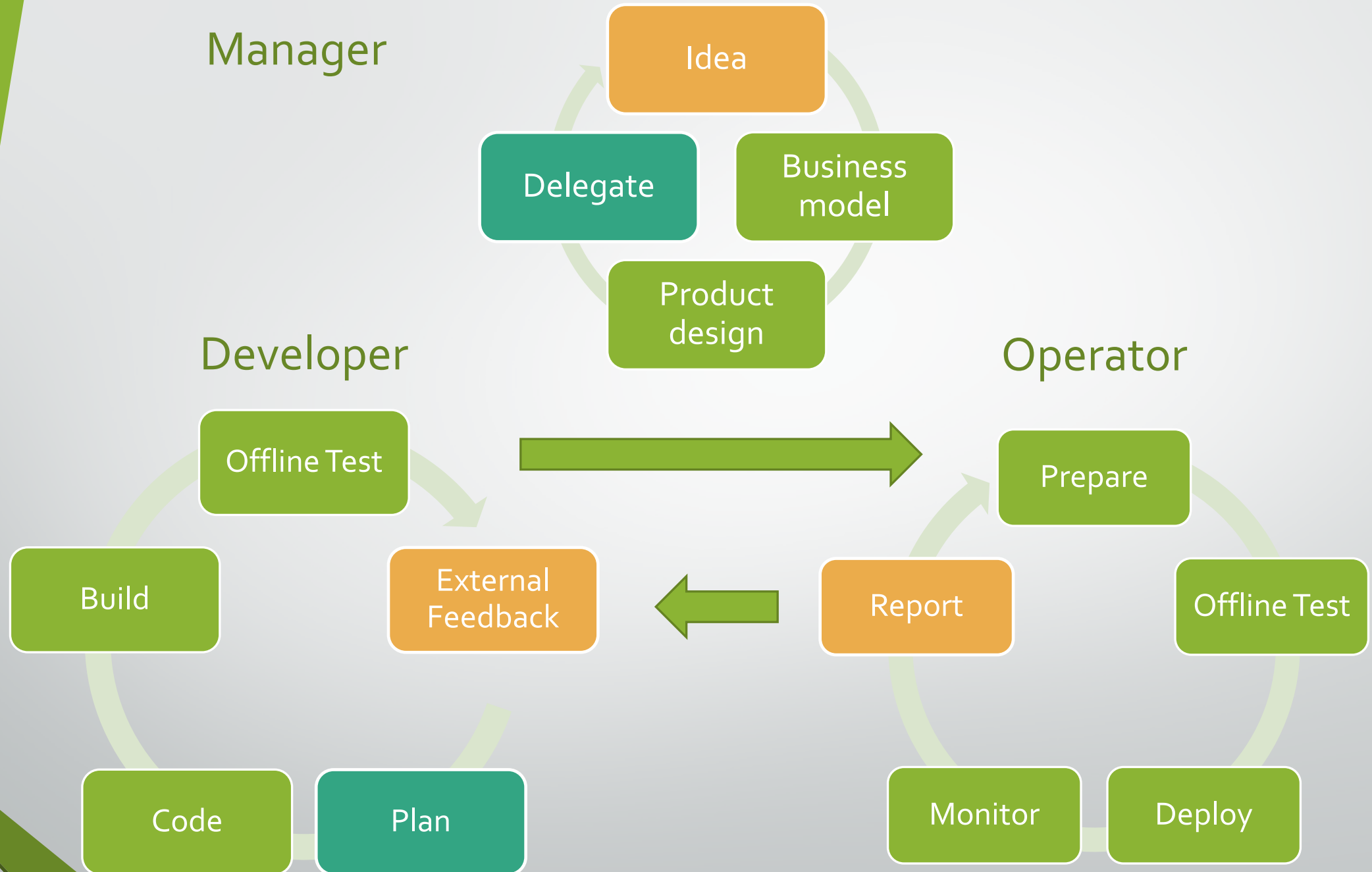
Challenges

- Splitted Teams
  - Communication problems
  - Manual and double testing
  - The teams expertise is not shared across department borders
- Every Team works on their own
  - No sense of unity



Source: https://vincentdnl.com/drawings/


Management
Product design
Operations
Developer


Manager
Idea
Business model
Product design
Delegate
Developer
Offline Test
Build
External Feedback
Code
Plan
Operator
Prepare
Offline Test
Deploy
Monitor
Report

Idea…

What to improve?

- Communication
- Company knowledge base
- Product quality
- Cycle time
- Human-factor errors
- Repetitive tasks

Approaches

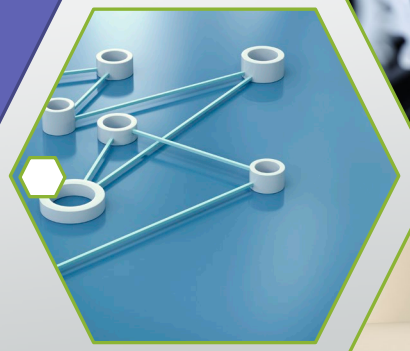
- Communication
- Company knowledge base
- Product quality
- Cycle time
- Human-factor errors
- Repetitive tasks

Wiki

Versioning

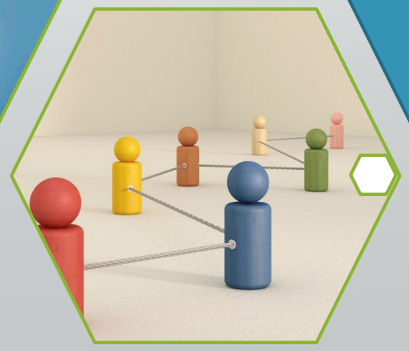
Project Management

Automation



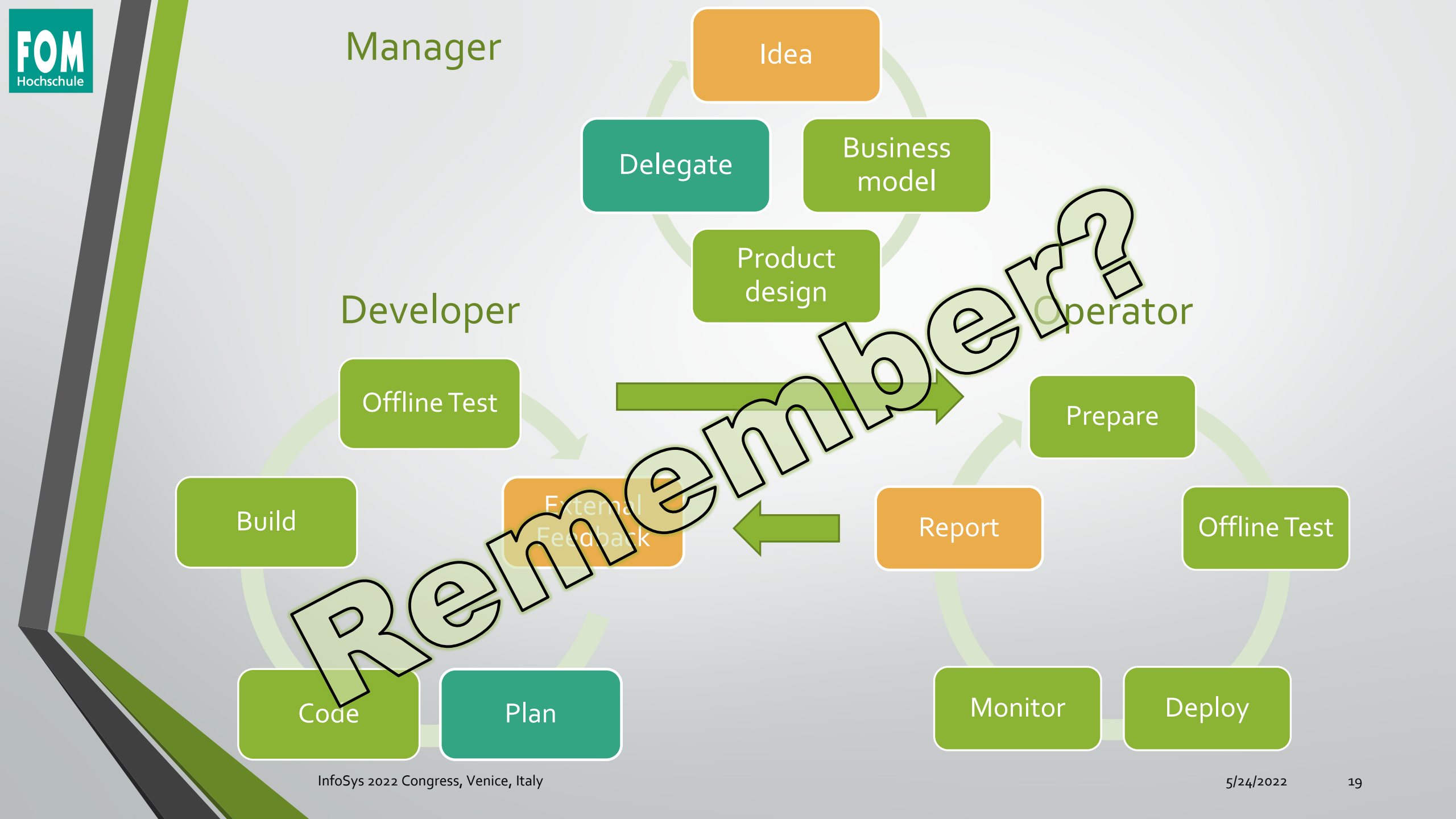
Blameless meetings

Feedback

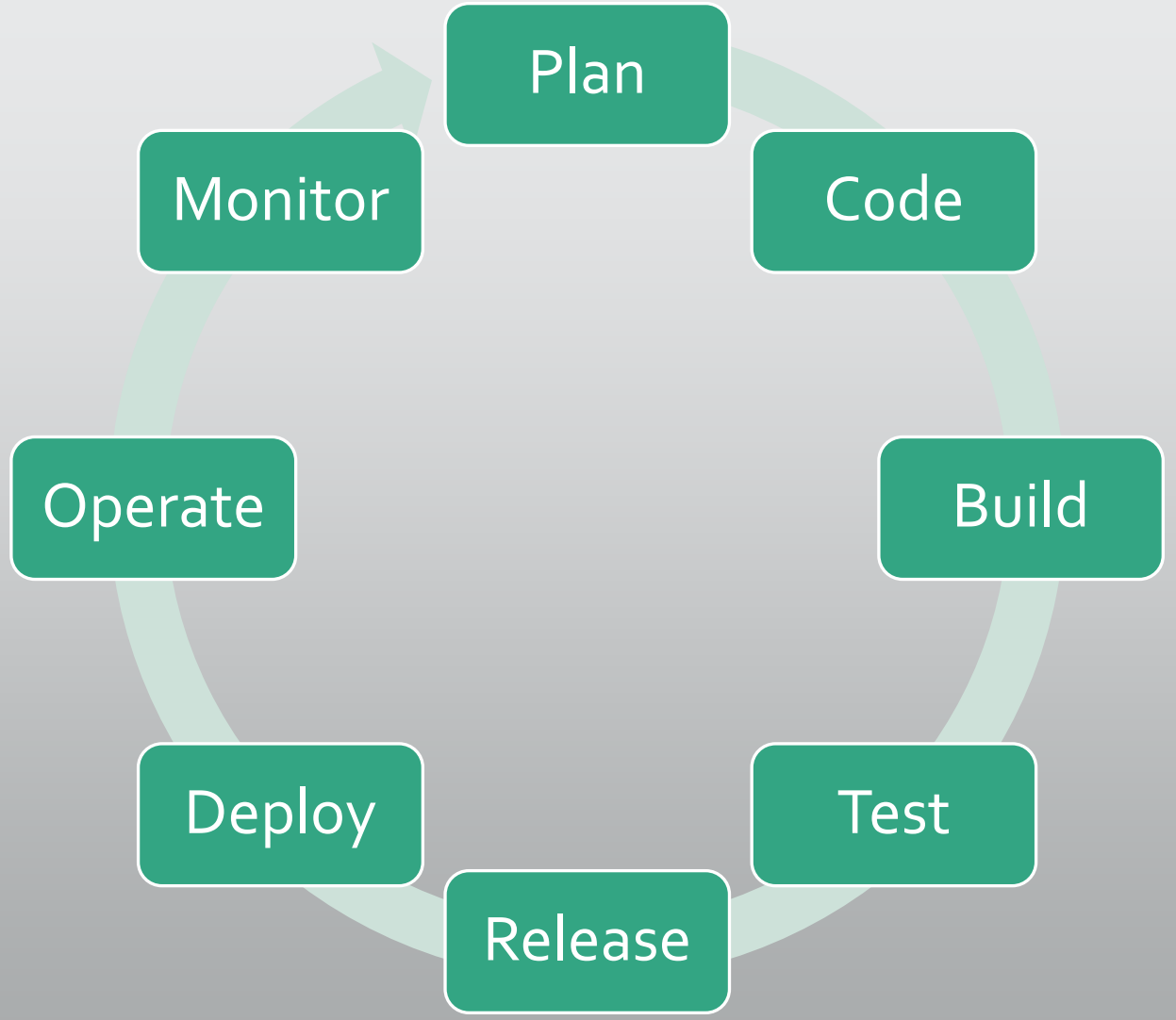


FOM Hochschule

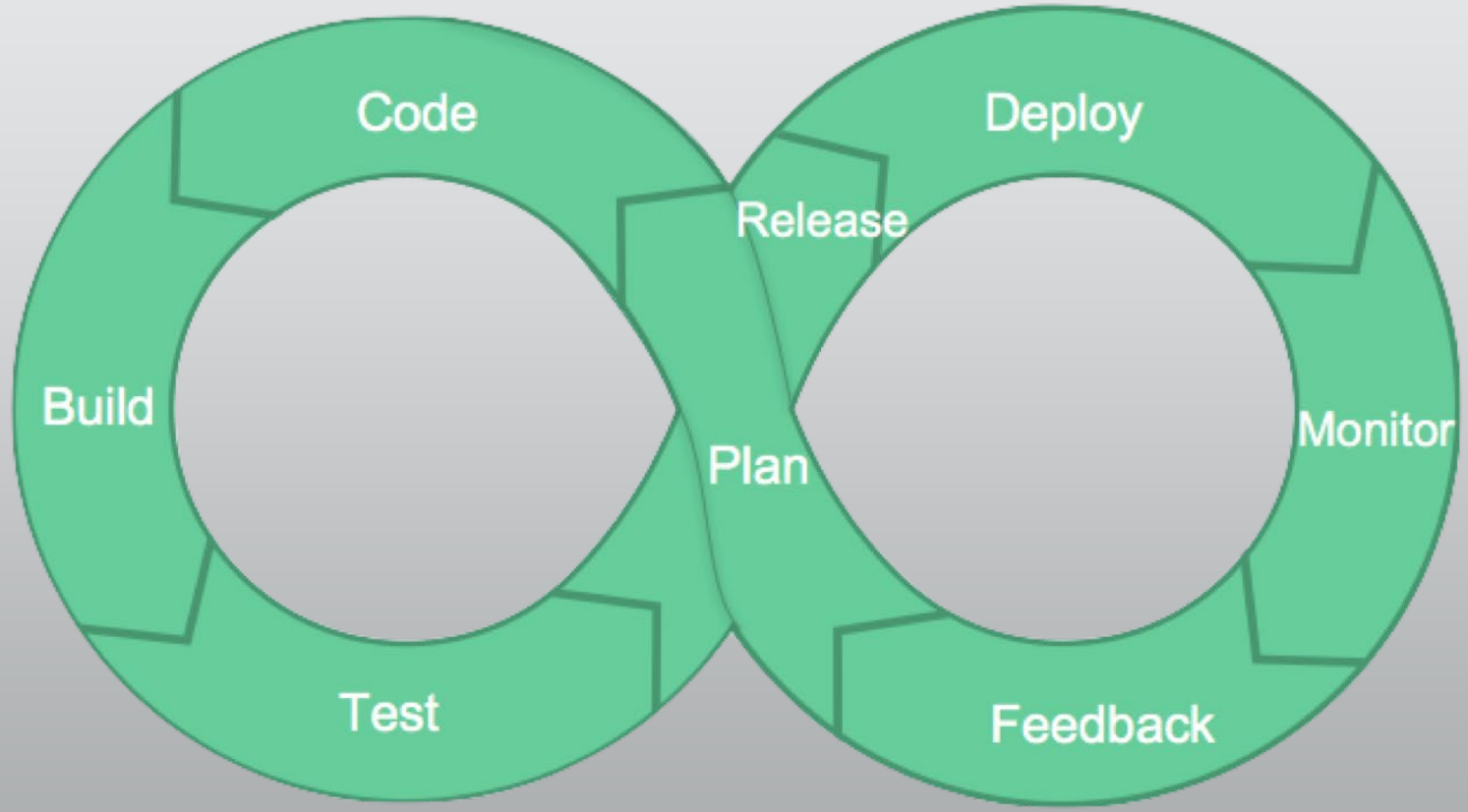
Combine Development & Operations = **DevOps**


Manager
Idea
Business model
Product design
Delegate
Developer
Offline Test
Build
Code
Plan
Operator
Prepare
Offline Test
Deploy
Monitor
Report
Remember?

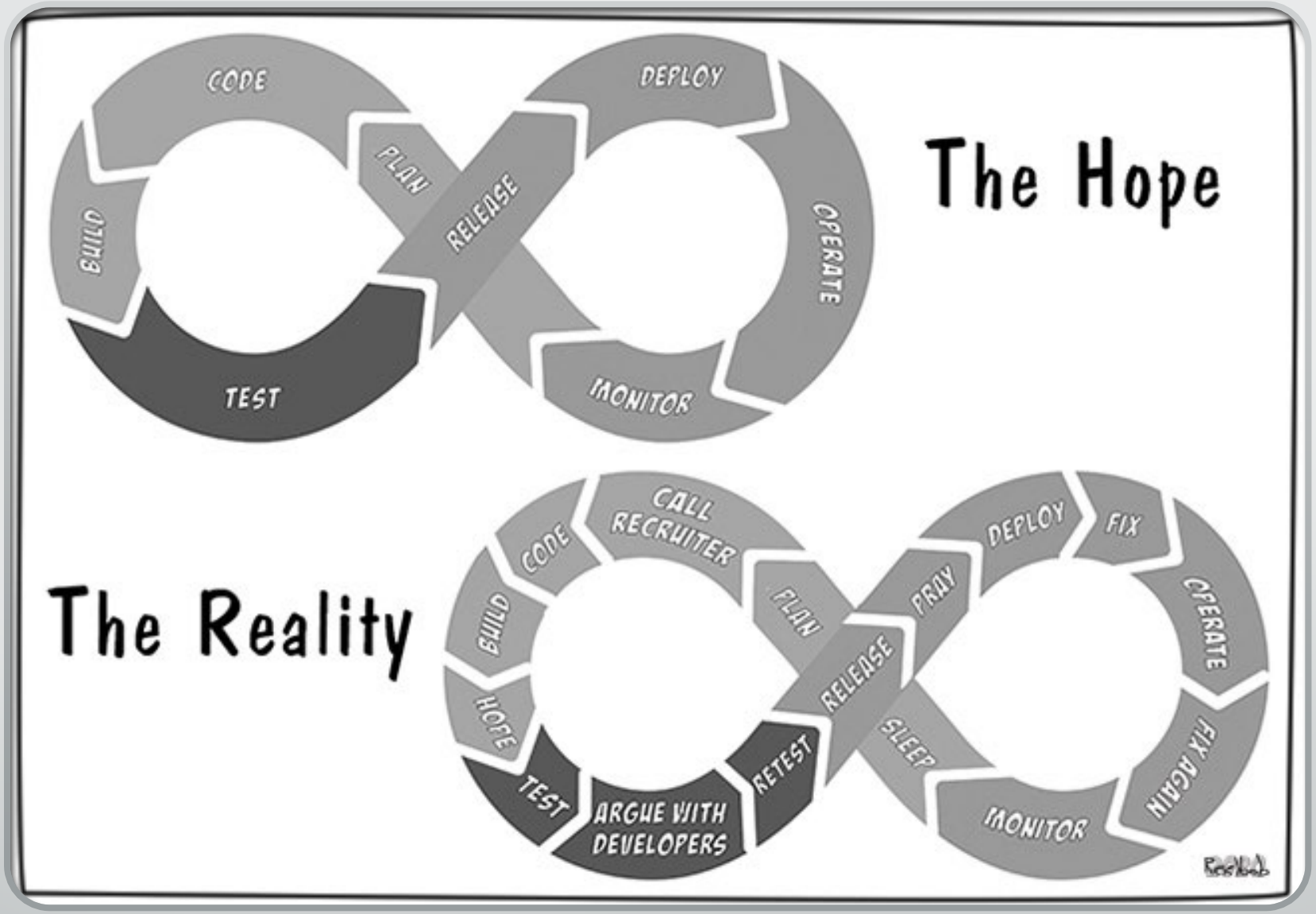
DevOps Lifecycle



DevOps Lifecycle



Source: https://ovh.github.io/tat/imgs/devops-lifecycle.png



Source: https://devops.com/the-devops-life-cycle/

DevOps Culture

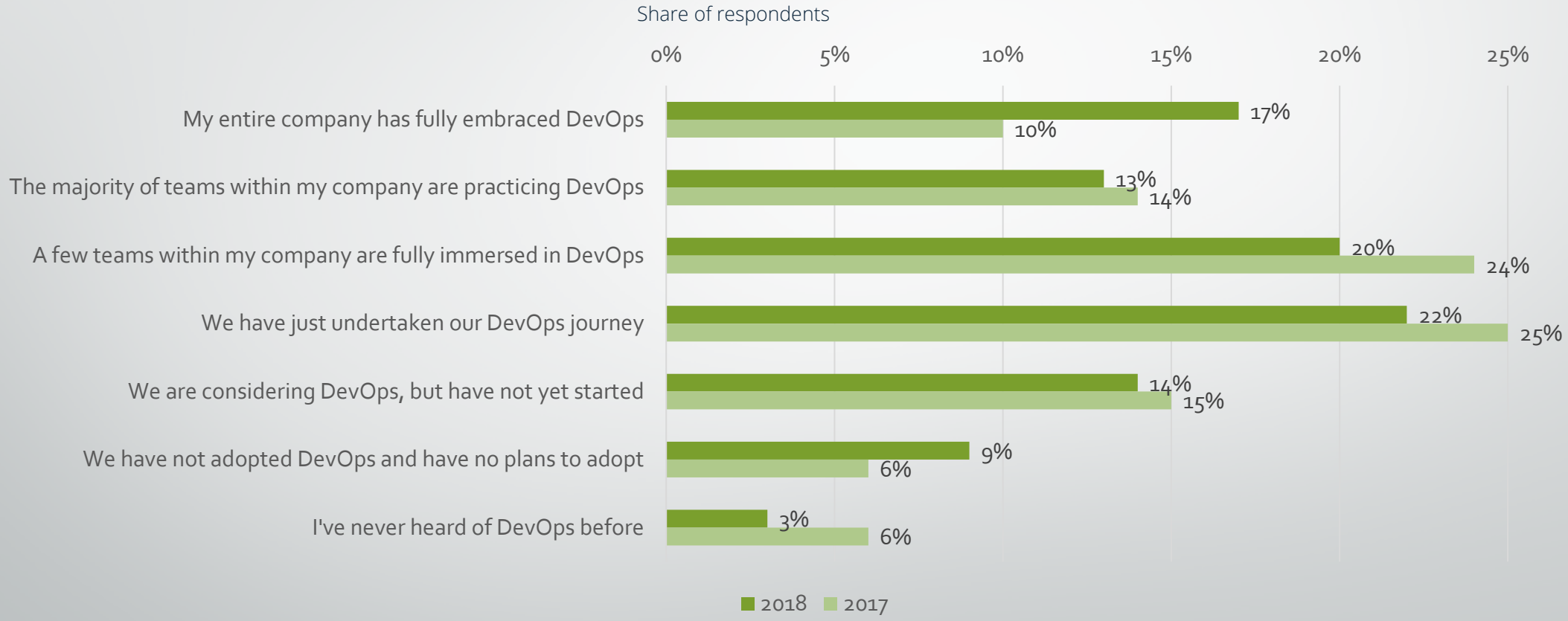
- No naming
- No blaming
- No shaming

- Toyota Production System
  - Andon-Cord (https://www.shmula.com/about-peter-abilla/what-is-andon-in-the-toyota-production-system/)


Source: https://avani.vn/en/andon-day-keo-huyen-thoai-cua-toyota/

Extent of DevOps adoption by software developers worldwide in 2017 and 2018

DevOps adoption among software developers globally 2017-2018



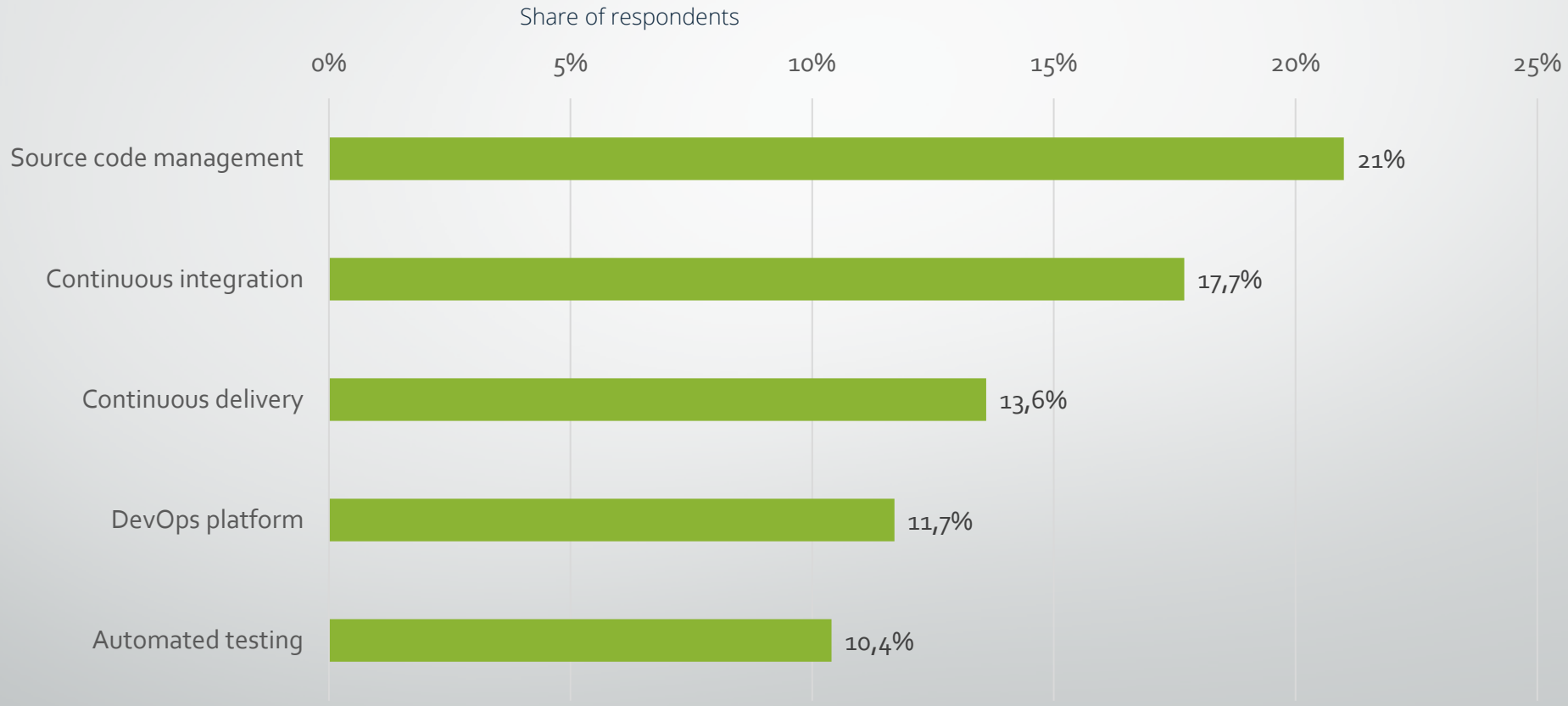
Note(s): Worldwide; 2016 to 2018; 1091 respondents; technology professionals responsible for development and quality of web and mobile applications

Further information regarding this statistic can be found on page 8.

Source(s): Sauce Labs; Dimensional Research; ID 673505

Changes made to software development process in DevOps teams in organizations worldwide in 2021

DevOps teams software development process changes in organizations worldwide 2021



Note(s): Worldwide; February to March, 2021; 4,294 respondents; developers, operations, and security professionals*
Further information regarding this statistic can be found on page 8.
Source(s): GitLab; ID 1234098

Benefits

- Improved software quality
- Use of modern technologies
- Less cycle time
- Company wide learning
  - Publish in knowledge management system

How to…

How to start implementing an DevOps environment?

- There are thousand ways and tools to build a DevOps infrastructure
  - A blueprint doesn't exist
- Where to start?
  - Understand the concept and phases of DevOps (e.g. lifecycle and practices)
  - Define your companys requirements (e.g. objectives, technologies and budget)

Technologies

- Most used technologies to implement DevOps
  - Source Code Management
  - Infrastructure-as-Code (IaC)
  - Software Container / Container Registry
  - CI/CD platform
  - Container Management

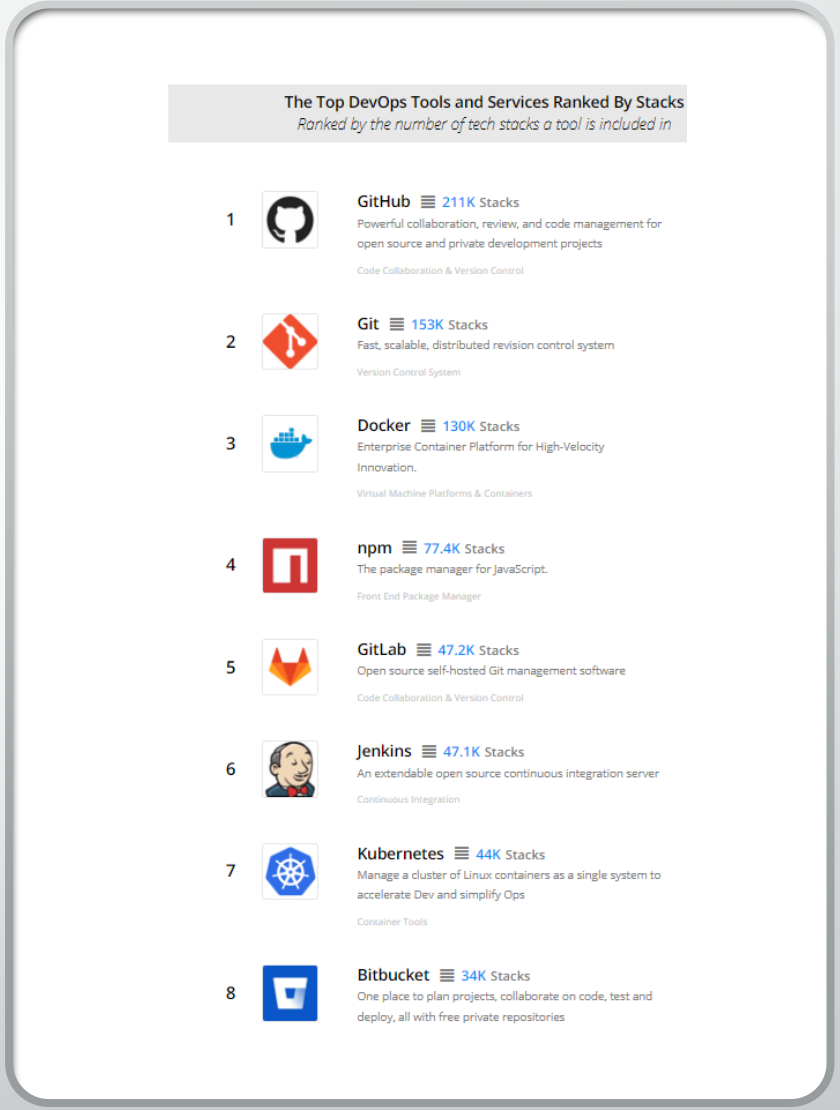
DevOps tools

- Functionality match to purpose
  - Limited to one technology (single part of the tool set)
  - Combining multiple technologies
- Licensing
  - Open Source Software
  - Commercial

Which tools are available?

- Stackshare offers a good overview
  - e.g. DevOps Index
  - Shows how often Software is used in a stack
    - Indicates realistic popularity

Source: https://stackshare.io/index/devops - retrieved on 2022-05-06



The Top DevOps Tools and Services Ranked By Stacks
Ranked by the number of tech stacks a tool is included in

1 **GitHub** 211K Stacks
Powerful collaboration, review, and code management for open source and private development projects
Code Collaboration & Version Control

2 **Git** 153K Stacks
Fast, scalable, distributed revision control system
Version Control System

3 **Docker** 130K Stacks
Enterprise Container Platform for High-Velocity Innovation.
Virtual Machine Platforms & Containers

4 **npm** 77.4K Stacks
The package manager for JavaScript.
Front End Package Manager

5 **GitLab** 47.2K Stacks
Open source self-hosted Git management software
Code Collaboration & Version Control

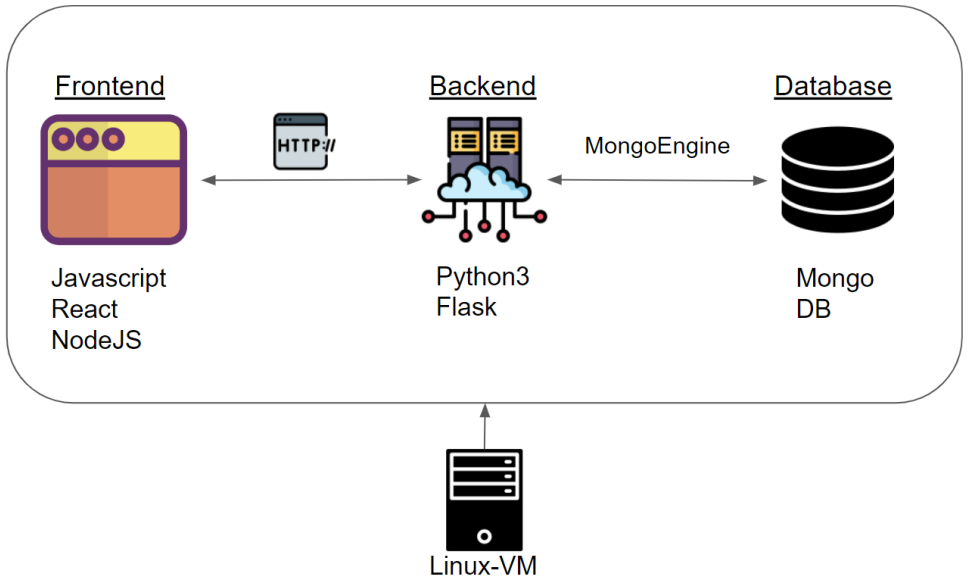
6 **Jenkins** 47.1K Stacks
An extendable open source continuous integration server
Continuous Integration

7 **Kubernetes** 44K Stacks
Manage a cluster of Linux containers as a single system to accelerate Dev and simplify Ops
Container Tools

8 **Bitbucket** 34K Stacks
One place to plan projects, collaborate on code, test and deploy, all with free private repositories

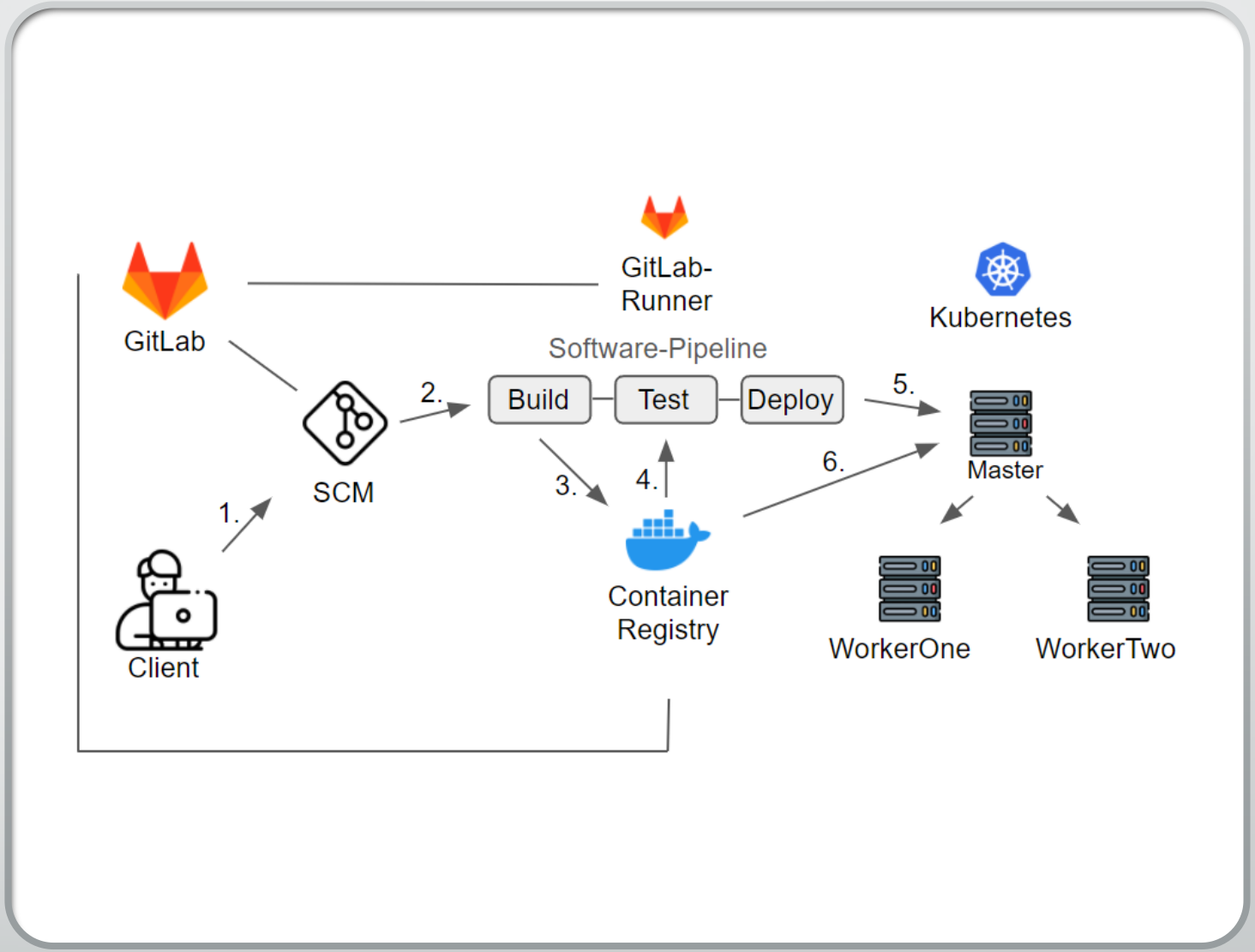
Case Studies

- can help defining what you need



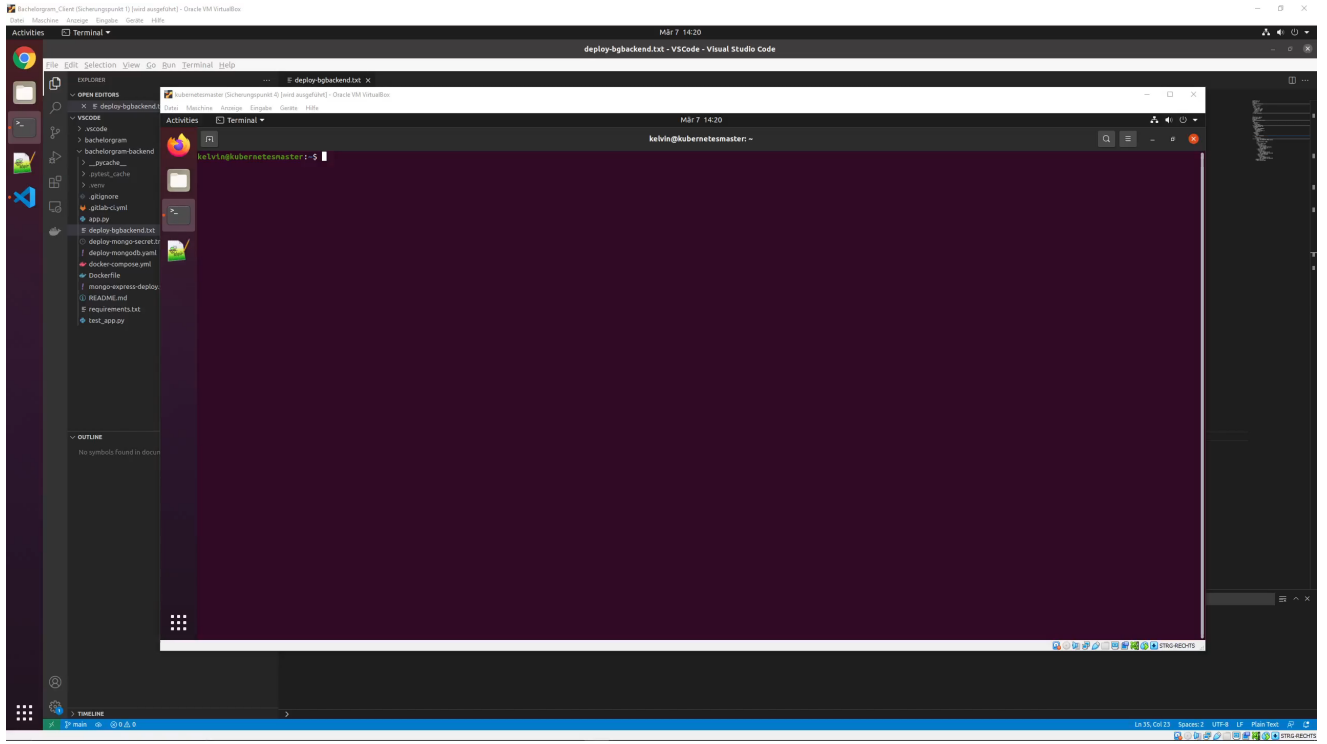
Example DevOps environment

- Full Open Source
  - No license costs
- Scalable
- External monitoring (e.g., Prometheus or Nagios) can be added

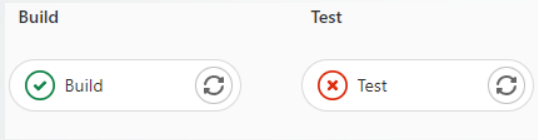


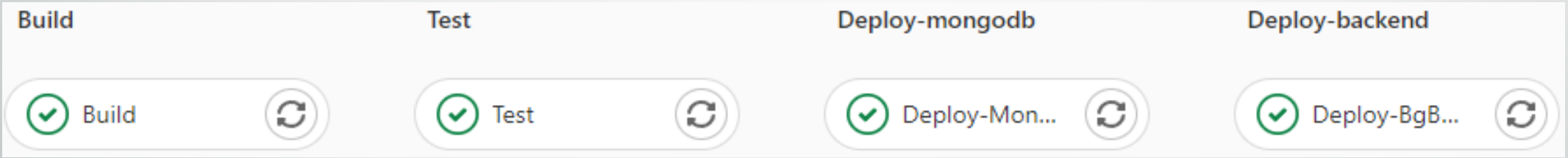
Example DevOps environment

Source: Kelvin Kühn - created on 2022-03-07



- Full Open Source
  - No license costs
- Scalable
- External monitoring (e.g., Prometheus or Nagios) can be added


Build
Test
Build
Test

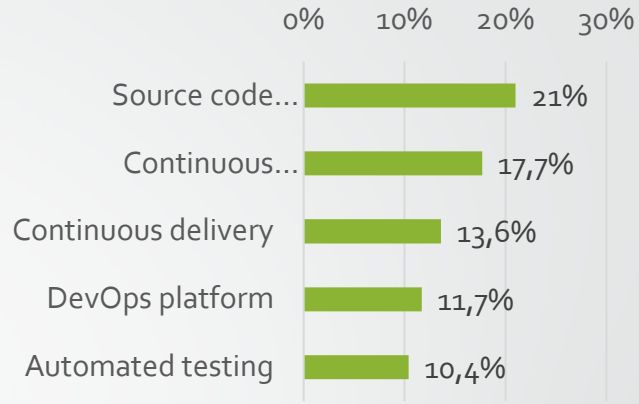

Build
Test
Deploy-mongodb
Deploy-backend
Build
Test
Deploy-Mon...
Deploy-BgB...

Implement…

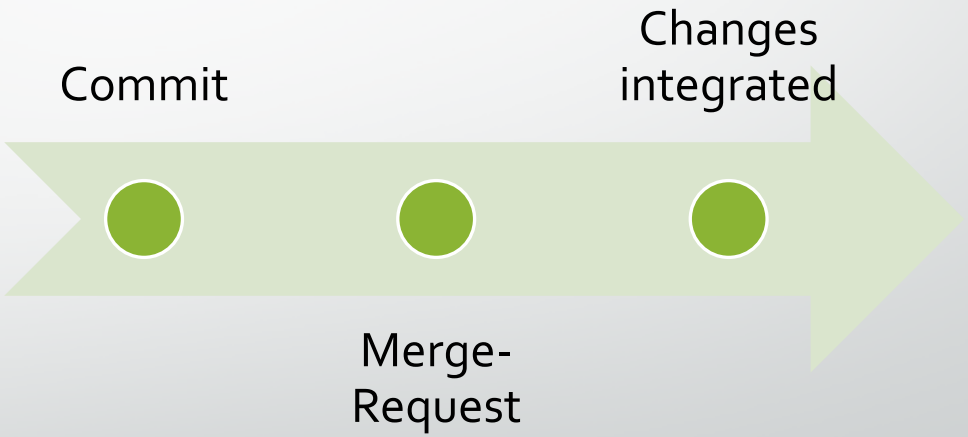
Step-by-Step

- Every technology works on its own and grants immediate benefits
  - Implement one-by-one
  - Finally combine them to automize your end-to-end value chain
- Most modern companys are already using some of the technologies
  - Create a big picture for your company's own DevOps environment

Versioning



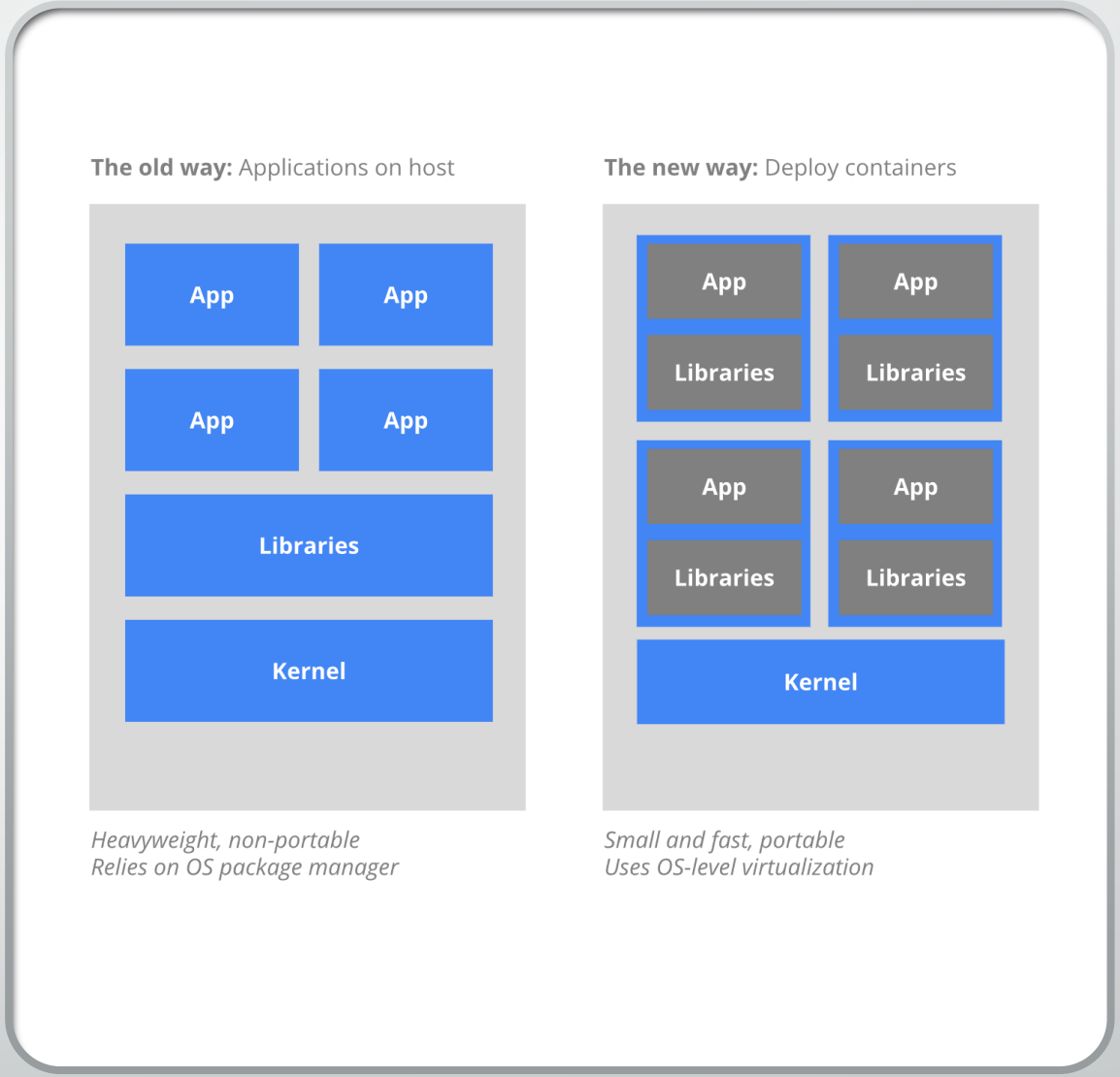
- Good option to start
  - Creates changelog from commit comments
  - Grants version control
    - Source Code
    - Software Container
  - Enables distributed and decentralized working



Software Container

- Portable
- Lightweight
- Fast to use

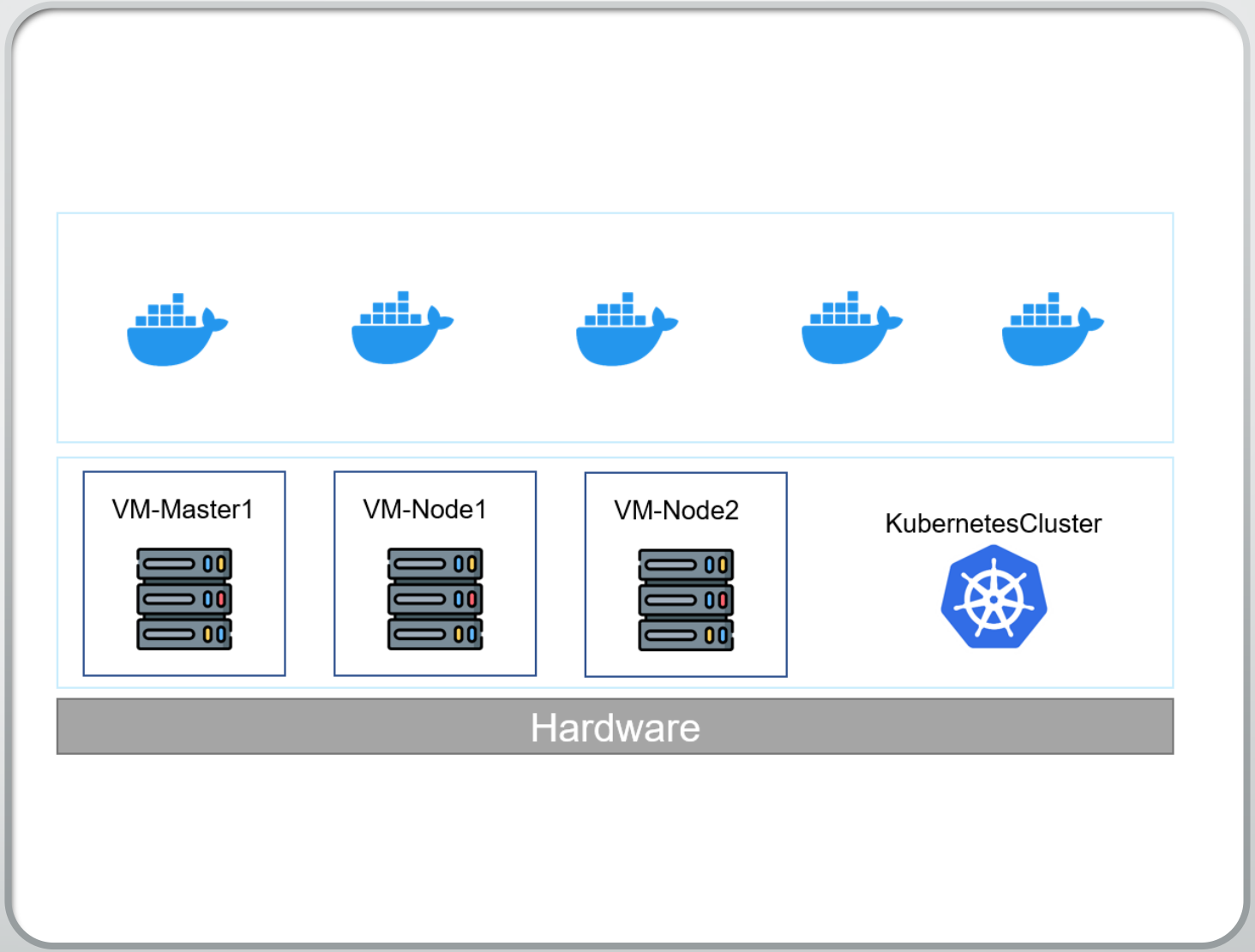
- Develop and build your software for software containers



Source: https://kubernetes.io/de/docs/concepts/overview/what-is-kubernetes/

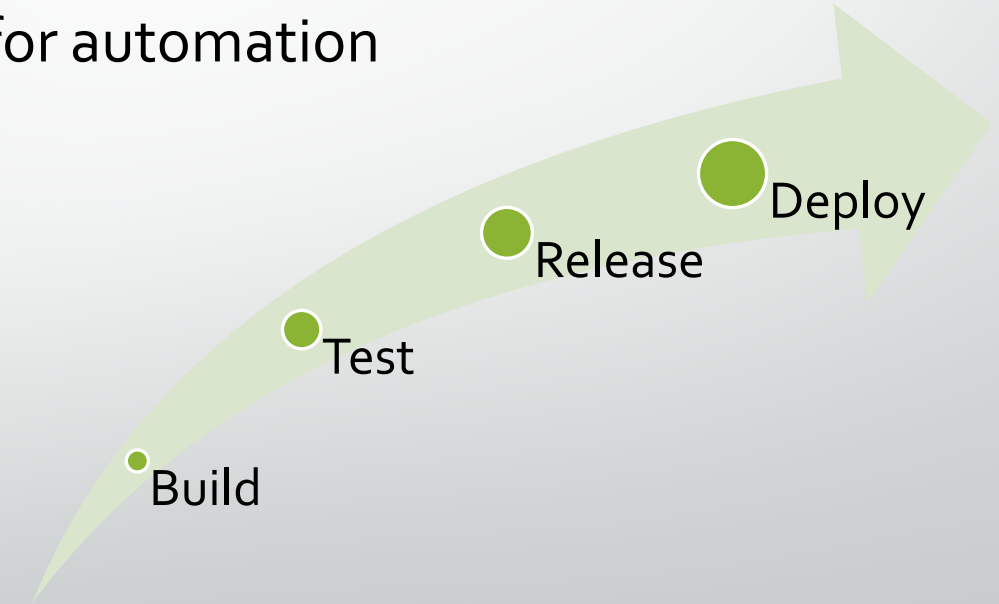
Container Management

- Horizontal scaling (scale-out)
- Infrastructure
- Networking
- Configurations
- Desired state approach to ensure availability



CI/CD platform

- Uses code/artifacts from versioning for automation
  - Building the application
  - Testing (e.g., unit or performance tests)
  - Release and Deploy on environment



Real-World Examples

Start simple

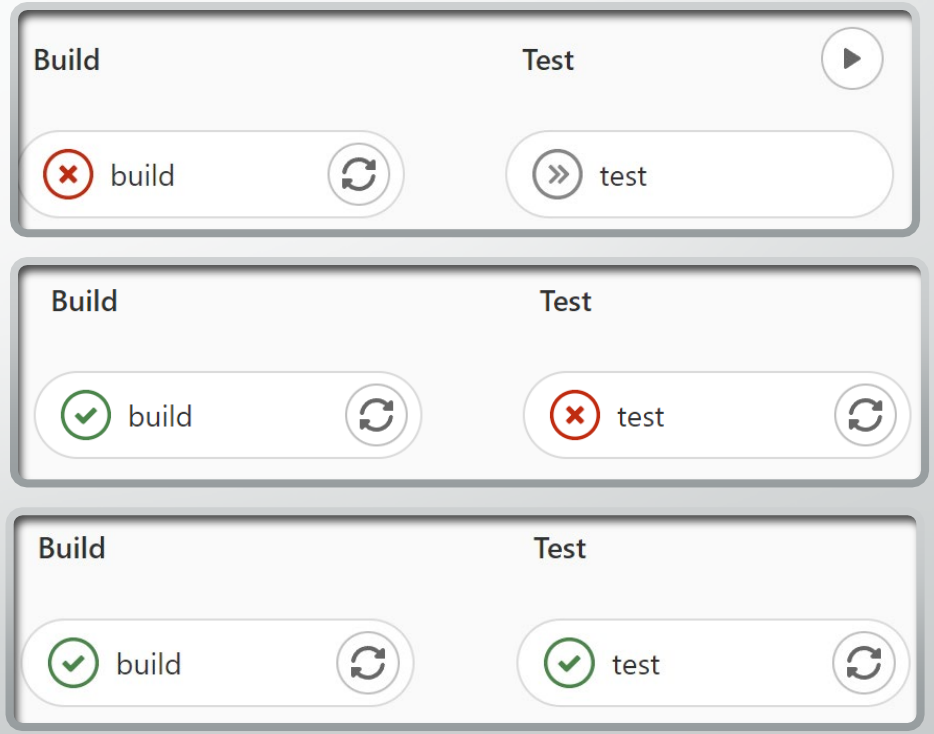

```yaml
image: python:3.7

stages:
    - test
cache:
    paths:
        - "~/.platformio"

before_script:
    - pip install -U configparser
    - pip install -U platformio
    - platformio update

job:
    stage: test
    script:
        - platformio run -e test
    tags:
        - python27
```

- Even for embedded systems development!
- Start simple, grow more complex over time
- Actually saved my day…
  - Caught error in a third-party library

WebApp Example

```
image: trion/ng-cli

cache:
  paths:
    - node_modules/

build:
  stage: build
  before_script:
    - |
    - NPM_PACKAGE_NAME=$(node -p "require('./package.json').name")
    - NPM_PACKAGE_VERSION=$(node -p "require('./package.json').version")
    - npm ci
    - npm i
    - npm audit fix

  script:
    - ng build --configuration production
  artifacts:
    expire_in: 1 day
    paths:
      - dist/
  tags:
    - docker

test:
  stage: test
  before_script:
    - |
    - NPM_PACKAGE_NAME=$(node -p "require('./package.json').name")
    - NPM_PACKAGE_VERSION=$(node -p "require('./package.json').version")
    - npm i
    - npm audit fix

  script:
    - ng lint
  tags:
    - docker
```

GitLab



Technical Things to Remember

- CI
  - „Continuous integration“
  - code, build, test
- CD
  - „Continuous delivery“
  - release (deliver) and deploy
- CF
  - „Continuous feedback“
  - monitor performance and availability
- DevOps
  - Blend of Development and Operations
- Pipeline
  - Connects technologies for automation
  - Only successfully run jobs reach next stage

Culture Things to Remember

- You are part of a community!
- Blameless working environment
- Test driven development
- Failures used to learn

Lessons Learned

- Head-in-the-sand is not a solution
- Rome wasn't built in a day either.
- Start tiny, step-by-step evolution
- Even small steps may lead to big improvement