UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Tecnologie Cloud & Mobile

Tecnologie Cloud e Mobile

Cloud & Mobile Project

mauro.pelucchi@gmail.com
Mauro Pelucchi

21059 - Tecnologie cloud e mobile

Cosa vedremo?

- DevOps
- Design as a Service
- Git
- Agile Methodology (basics)

DevOps

Quale è il problema più grosso delle applicazioni?

Problem statement

- Everything needs software and application nowadays
- Software has to run on a server to become a service
- Delivering a service is too slow
- Internal friction points (inside the IT organisation)
- Delay = loss = -money
- IT is frequently (uhm… always) the bottleneck in the transition of “concept to cash.”

Problem statment

Bug

Diagnosis too slow

Problems in some environment

Long delays while dev

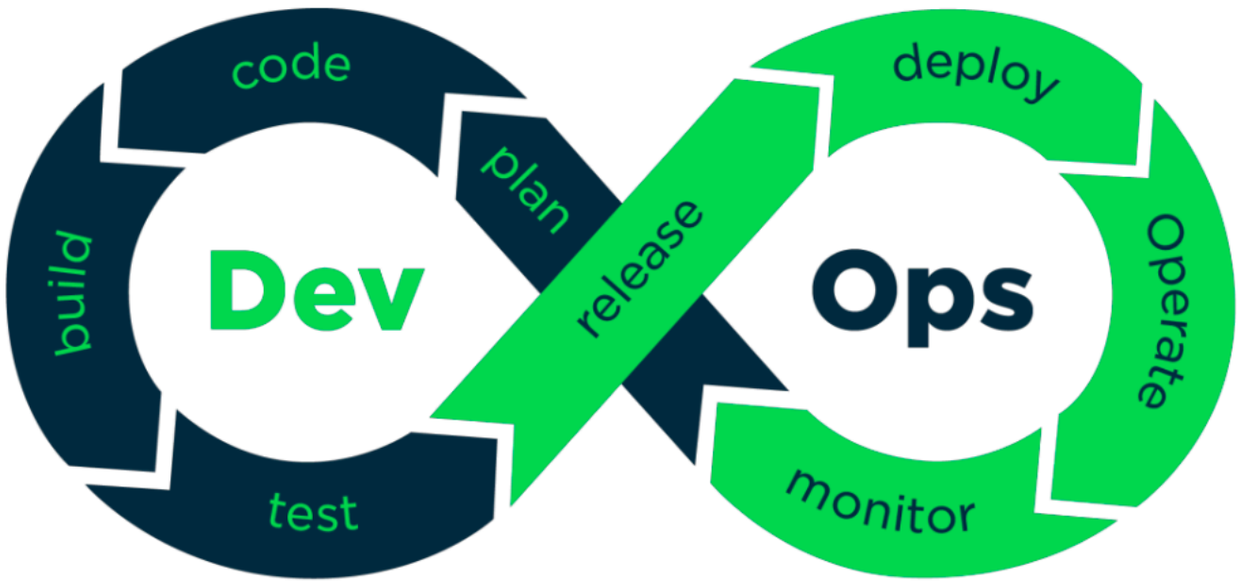
Manual errors

Quality of life issues in IT

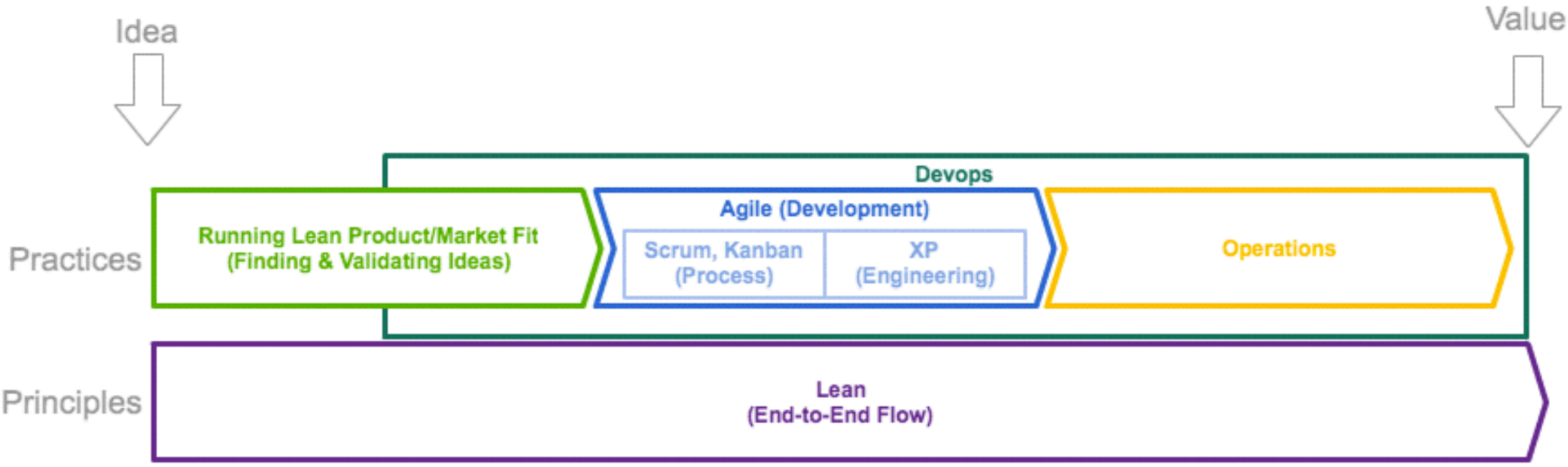
Releases fail

Recap

DevOps is the practice of operations and development engineers **participating together in the entire service lifecycle**, from design through the development process to production support



Developer and Operations = DevOps


Idea
Value
Devops
Practices
Running Lean Product/Market Fit
(Finding & Validating Ideas)
Agile (Development)
Scrum, Kanban
(Process)
XP
(Engineering)
Operations
Principles
Lean
(End-to-End Flow)

Core Values

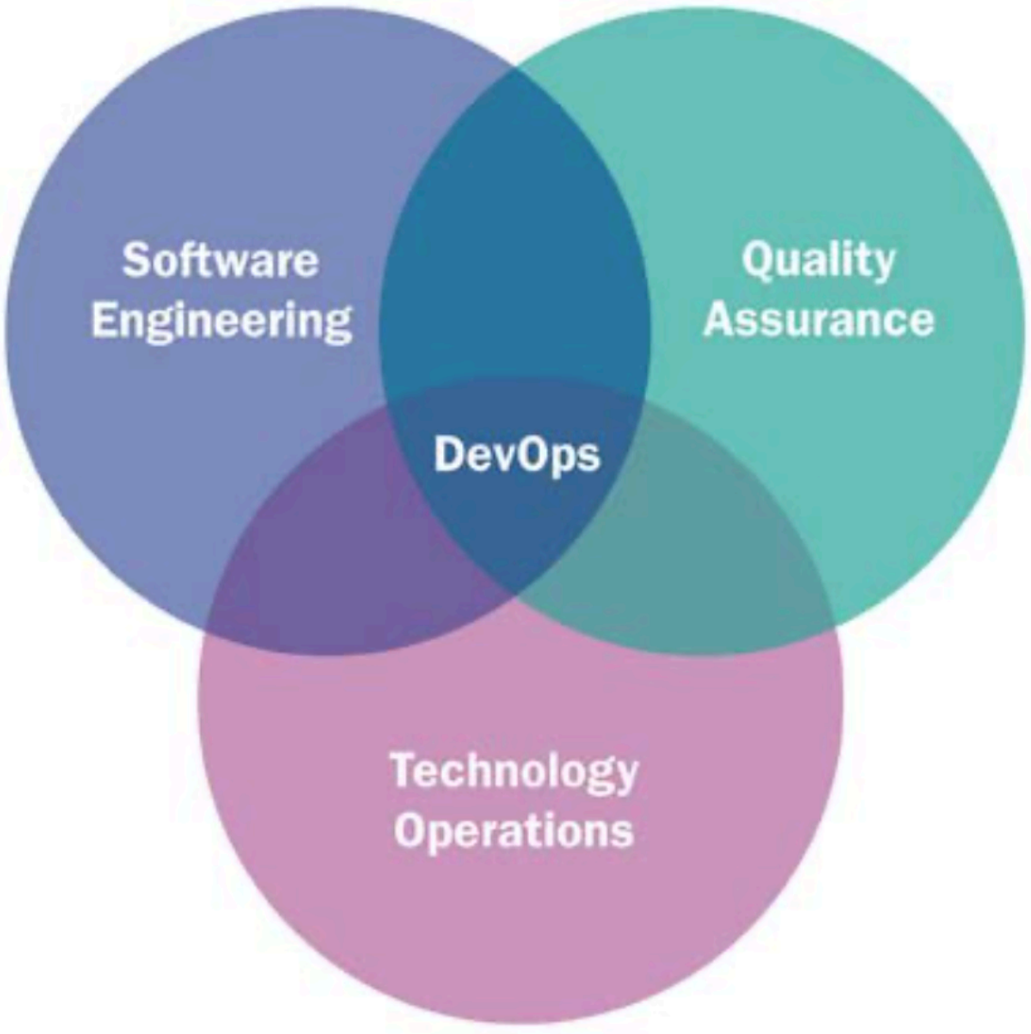
C — Culture: People -> Process -> Tools

A — Automation: Infrastructure as Code

M — Measurement: Measure Everything

S — Sharing: Collaboration & Feedback

Multiple teams integrations

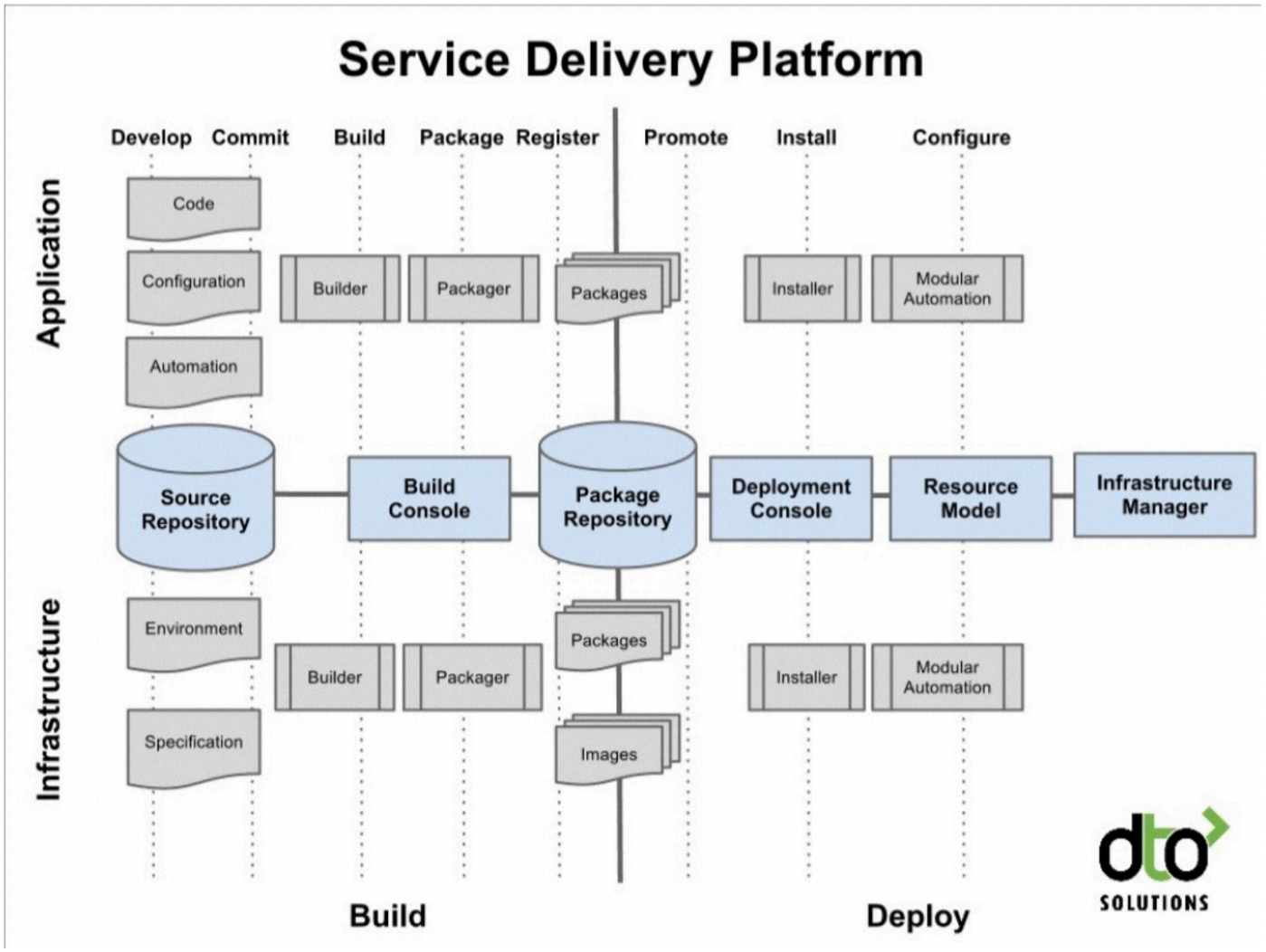


Pillars

1. Infrastructure Automation
2. Continuous Delivery
3. Reliability Engineering

Infrastructure as code

- Dev workflow
  - Write it in code
  - Validate the code
  - Unit test the code
  - Built it into an artifact
  - Deploy artifact to test
  - Integration test it
  - Deploy artifact to prod



AWS - Infrastructure as code

Infrastructure as code

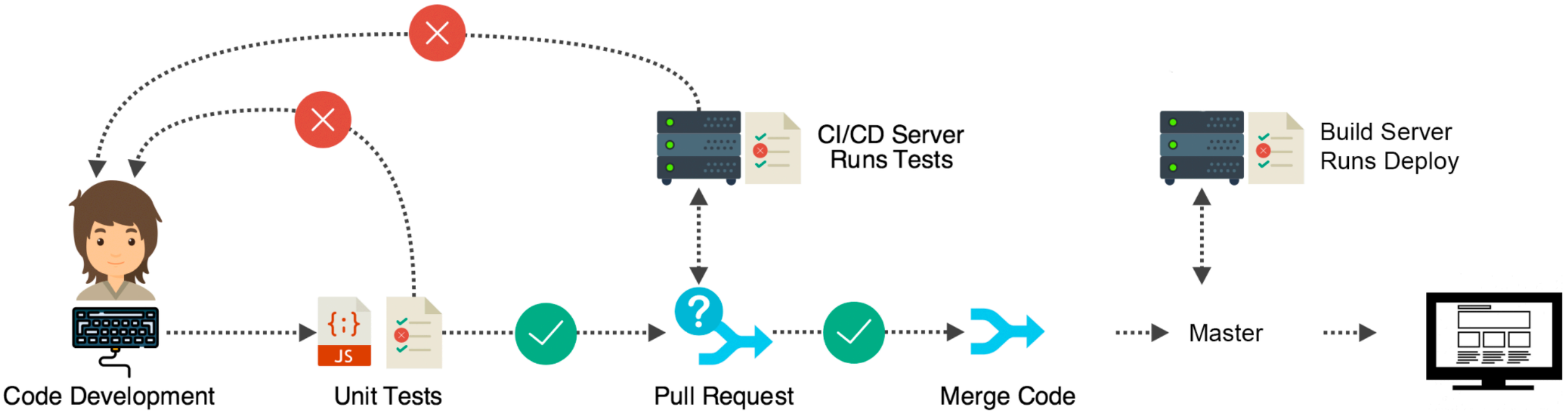
- Immutable deployment with docker
- Serverless with
  - AWS Lambda
  - Azure Functions
  - Google Cloud Functions
- Single model for infrastructure and apps is best (Kubernetes)

Continuos Delivery

- Integration (CI): Build and test
- Deployment (CD): Deploy and integration test
- Delivery: All the way to production

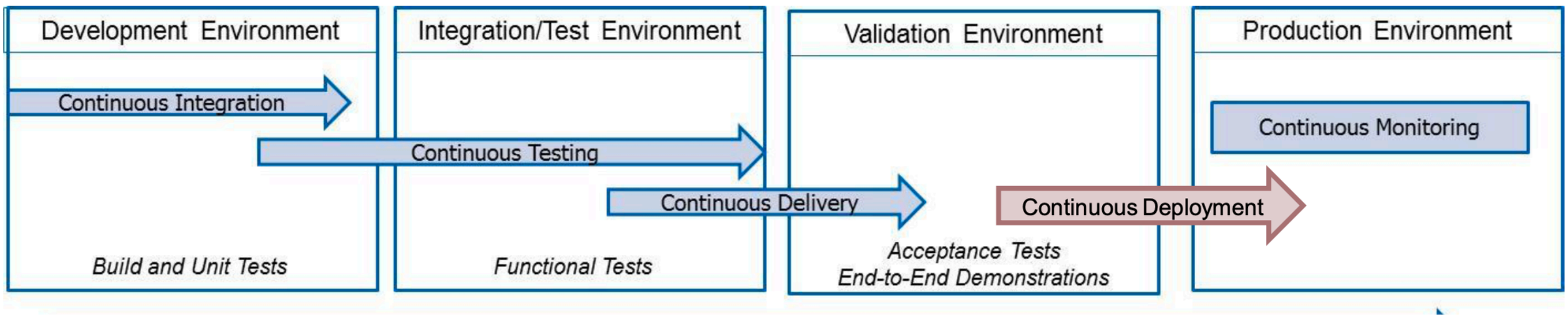
Best practice

- Builds should pass the coffee test (< 5 minutes)
- Commit really small bits
- Don’t leave the build broken
- Use a trunk/ master based development flow
- Don't allow flaky tests
- The build should return a status (a log)
- The build should return an artifact


CI/CD Server
Runs Tests
Build Server
Runs Deploy
JS
Code Development
Unit Tests
Pull Request
Merge Code
Master

Tools

- Version Control (Git… GitHub, CodeCommit, CodeLab, …)
- CI System (Jenkins, bamboo, AWS CI)
- Build (make/rake, maven, ….)
- Test (*unit, …)
- Artifact Repository (JFrog Artifactory, Dockerhub, S3)
- Deployment (AWS Code*)


Development Environment
Integration/Test Environment
Validation Environment
Production Environment
Continuous Monitoring
Continuous Integration
Continuous Testing
Continuous Delivery
Continuous Deployment
Build and Unit Tests
Functional Tests
Acceptance Tests
End-to-End Demonstrations

Reliability Engineering

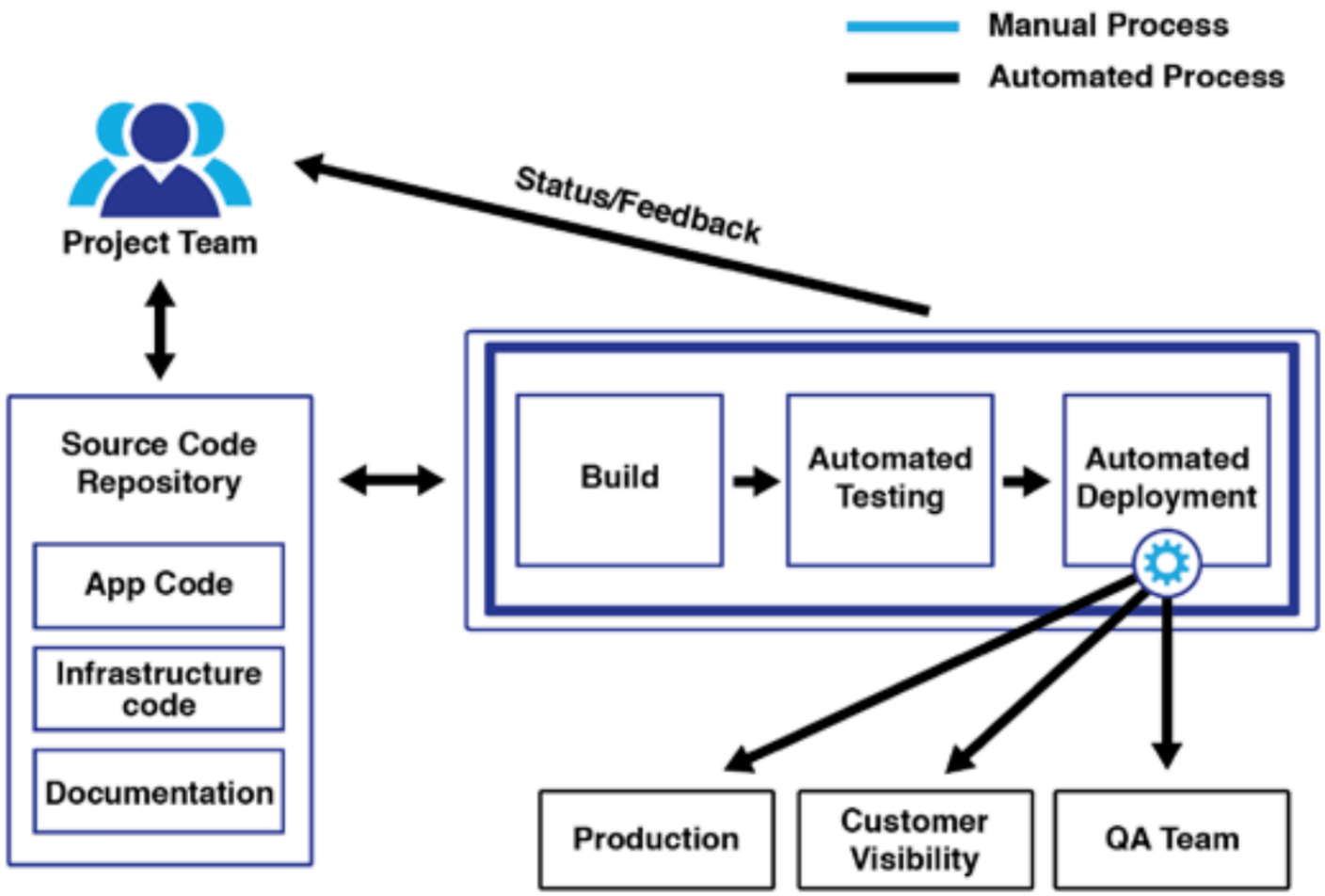
- Enhance **Service Design** With Operational Knowledge
  - Reliability
  - Performance
  - Security
  - Test These
  - Design Patterns

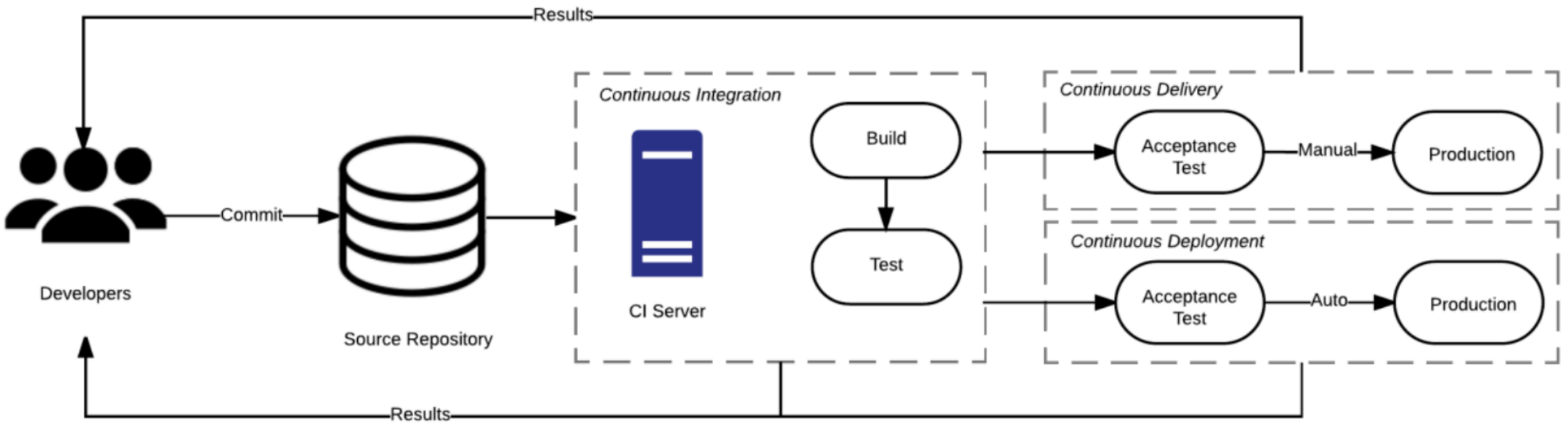
CI/CD

- Continuous Integration
  - The practice of merging development work with the main branch constantly.
- Continuous Delivery
  - Continual delivery of code to an environment once the code is ready to ship. This could be staging or production. The idea is the product is delivered to a user base, which can be QAs or customers for review and inspection.
- Continuous Deployment
  - The deployment or release of code to production as soon as it is ready.

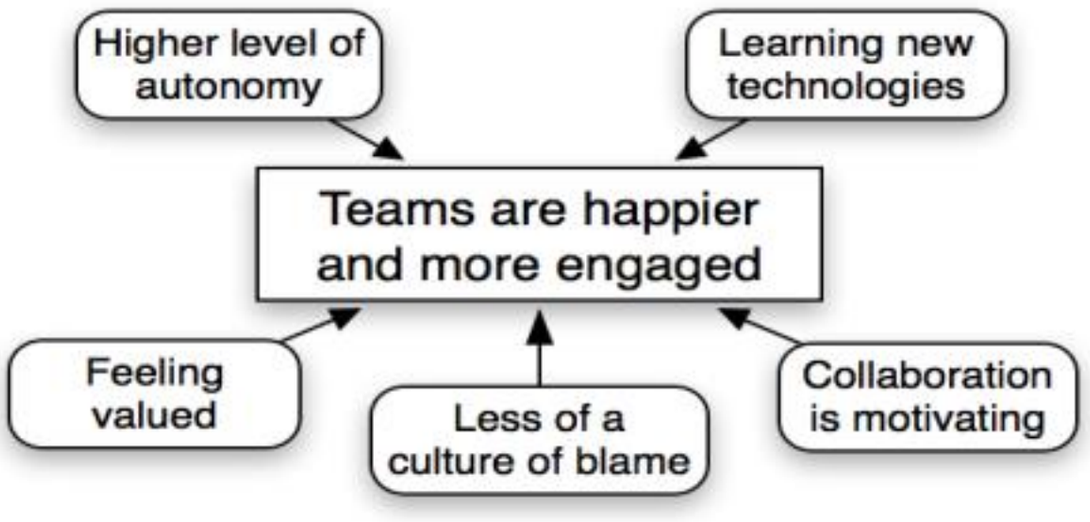
Whether your code passes test, gets deployed, and stays up for users is your responsibility – not someone else's

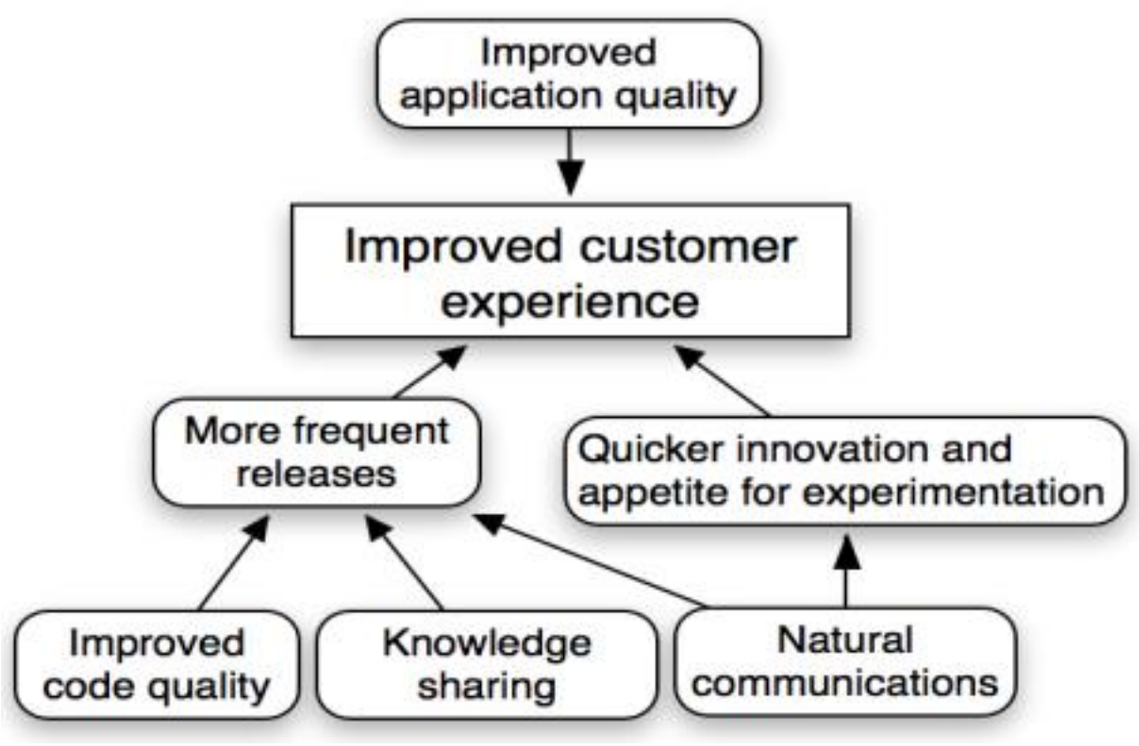
CI Model


Manual Process
Automated Process
Project Team
Status/Feedback
Source Code Repository
App Code
Infrastructure code
Documentation
Build
Automated Testing
Automated Deployment
Production
Customer Visibility
QA Team

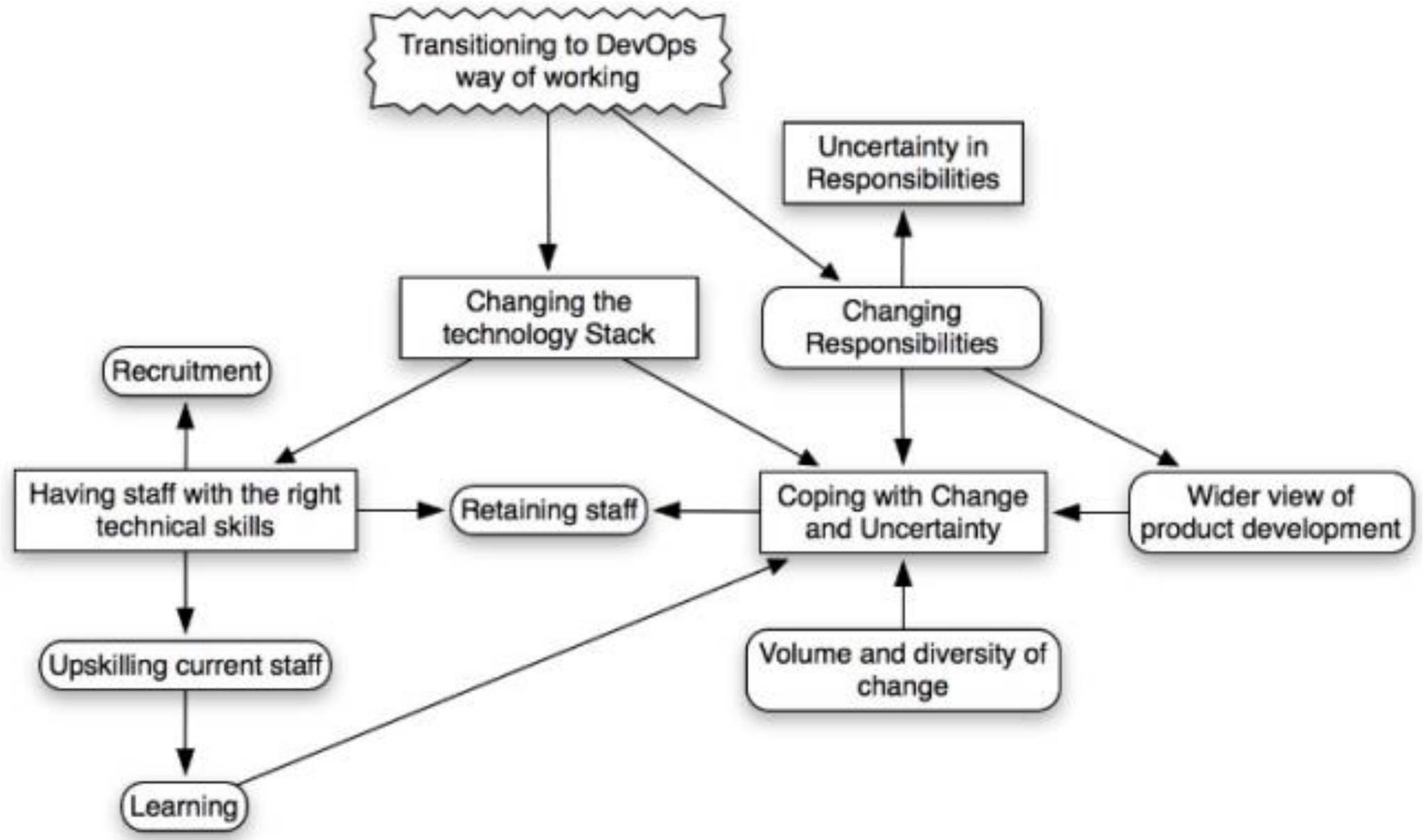

Results
Continuous Integration
Build
Test
CI Server
Continuous Delivery
Acceptance Test
Manual
Production
Continuous Deployment
Acceptance Test
Auto
Production
Commit
Developers
Source Repository
Results

Benefits realised from DevOps adoption





Challenges in Adopting DevOps



DevOps Aims to Increase

- The pace of innovation
- Responsiveness to business needs
- Collaboration software quality

Git

Conoscete Git?

Cosa è?

Scenario 1

- Your program is working
- You change “just one thing”
- Your program breaks
- You change it back
- Your program is still broken--why?
- Has this ever happened to you?

Scenario 2

- You change one part of a program. It works.
- Your co-worker changes another part. It works.
- You put them together. It doesn’t work
  - Some change in one part must have broken something in the other part
  - What were all the changes?

Scenario 3

- You make a number of improvements to a class
- Your co-worker makes a number of different improvements to the same class
- How can you merge these changes?

Version control systems

- Keeps multiple (older and newer) versions of everything (not just source code)
- Requests comments regarding every change
- Allows “check in” and “check out” of files so you know which files someone else is working on
- Displays differences between versions

Why use formal version control?

- History of changes
- Able to go back
- No worries about breaking things that work
- Merging changes from multiple people

What are Git and GitHub?

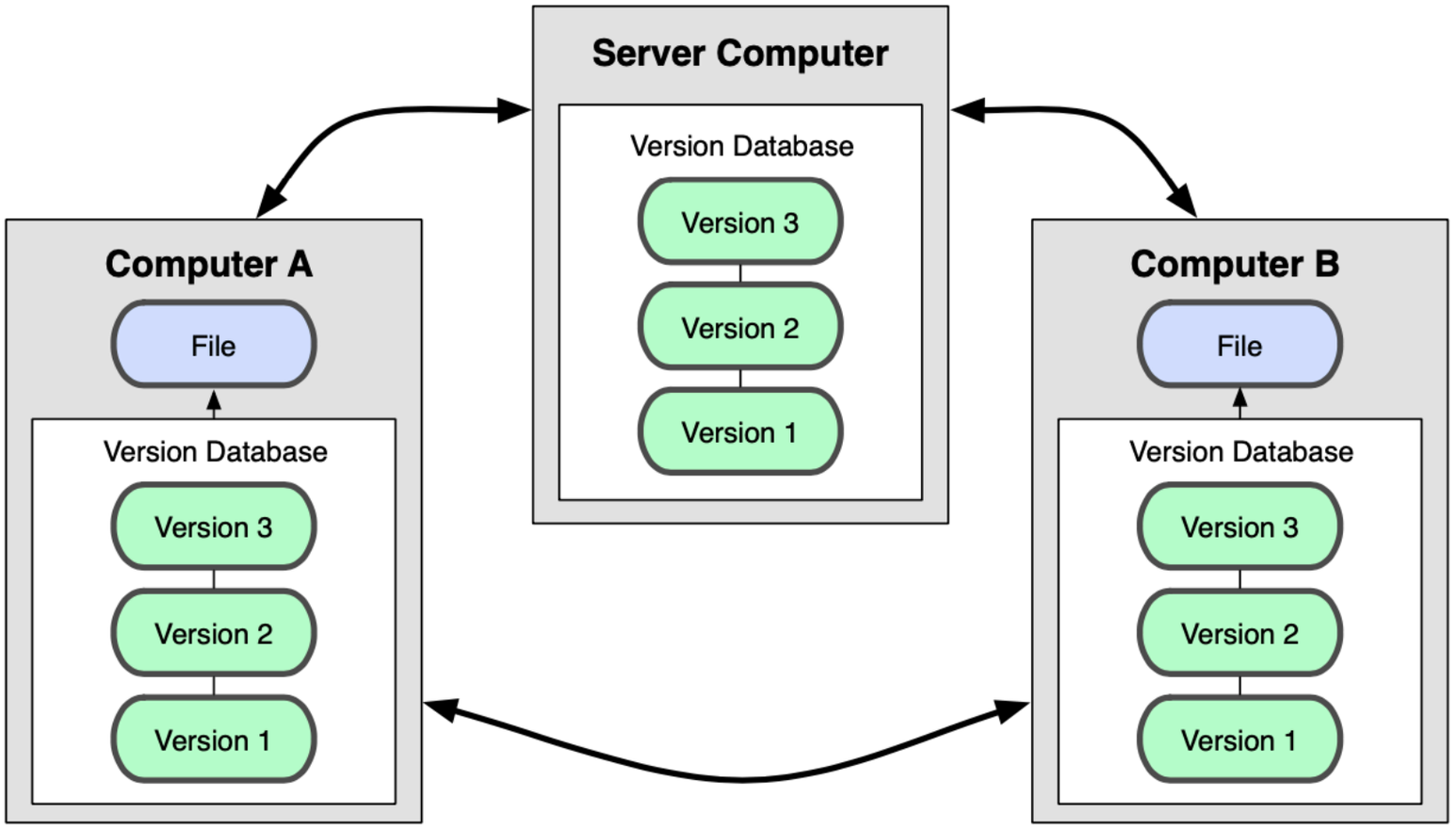
Git is a free and open source **distributed version control system** designed to handle everything from small to very large projects with speed and efficiency

GitHub is a web-based **Git repository hosting service**, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

Git history and concept

- Originally written for Linux kernel development.
  - All Linux kernel developers used to be able to use the proprietary Bitkeeper version control system for free.
  - In 2005 there were further restrictions put on Bitkeeper so that it wasn't as free as it used to be.
    - Linus Torvalds was uneasy with the situation and decided to write his own tool.
- Basically a versioned file system
  - The version control system is developed on top of this
  - More than 130 commands
  - Distributed repository model can be used centrally
  - Very fast!

Git
Distributed VCS



Basic usage

- Change some files
    - See what you've changed git status
        - git diff
        - git log
- Indicate what changes to save
    - git add
- Commit to those changes
    - git commit
- Push the changes to GitHub
    - git push
- Pull changes from your collaborator
    - git pull

Example of repository

mauropelucchi / unibg_mobile_and_cloud_2020

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security | Insights | Settings

Mobile and Cloud Technologies 2020

Edit

Manage topics

1 commit | 1 branch | 0 packages | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

mauropelucchi Initial commit — Latest commit 0747bc7 7 days ago

README.md — Initial commit — 7 days ago

README.md

unibg_mobile_and_cloud_2020

Mobile and Cloud Technologies 2020

Go!

Open VS Code…

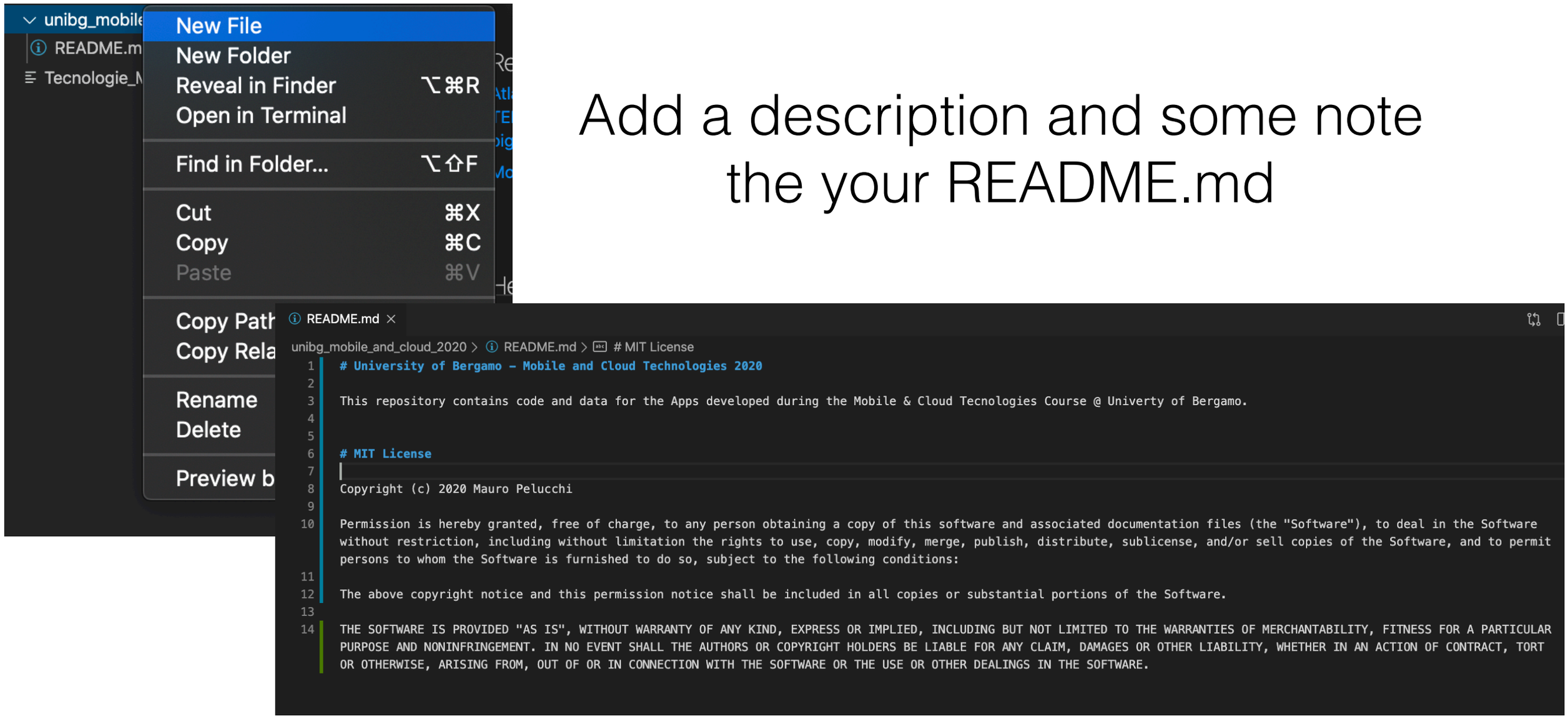
Go!
git clone <url>

Open VS Code…

and clone your repository

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) shuttle:Tecnologie Cloud e Mobile 2020 mauropelucchi$
(base) shuttle:Tecnologie Cloud e Mobile 2020 mauropelucchi$
(base) shuttle:Tecnologie Cloud e Mobile 2020 mauropelucchi$ git clone https://github.com/mauropelucchi/unibg_mobile_and_cloud_2020.git
```

Create a README.md file

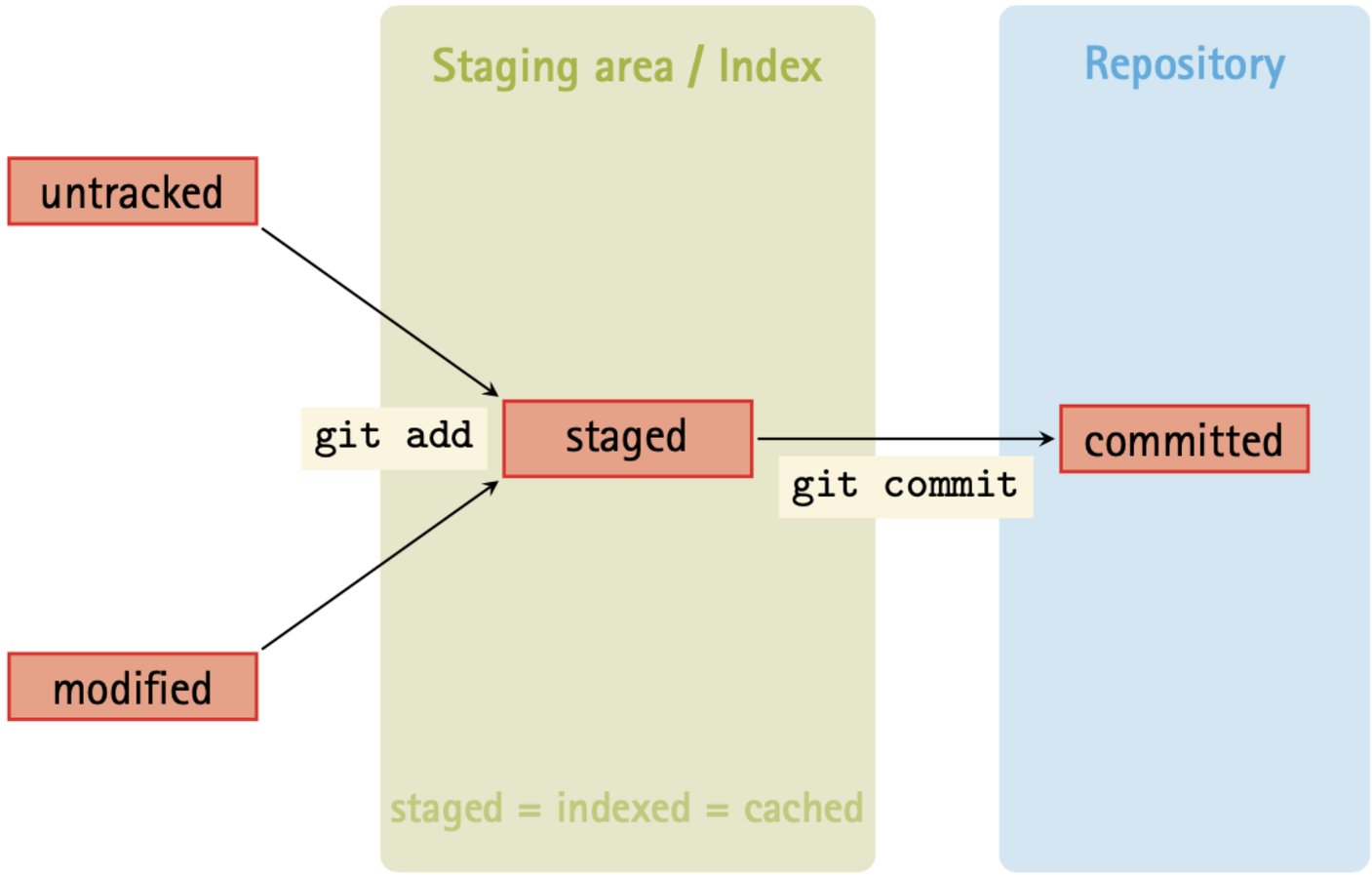


Add a description and some note the your README.md

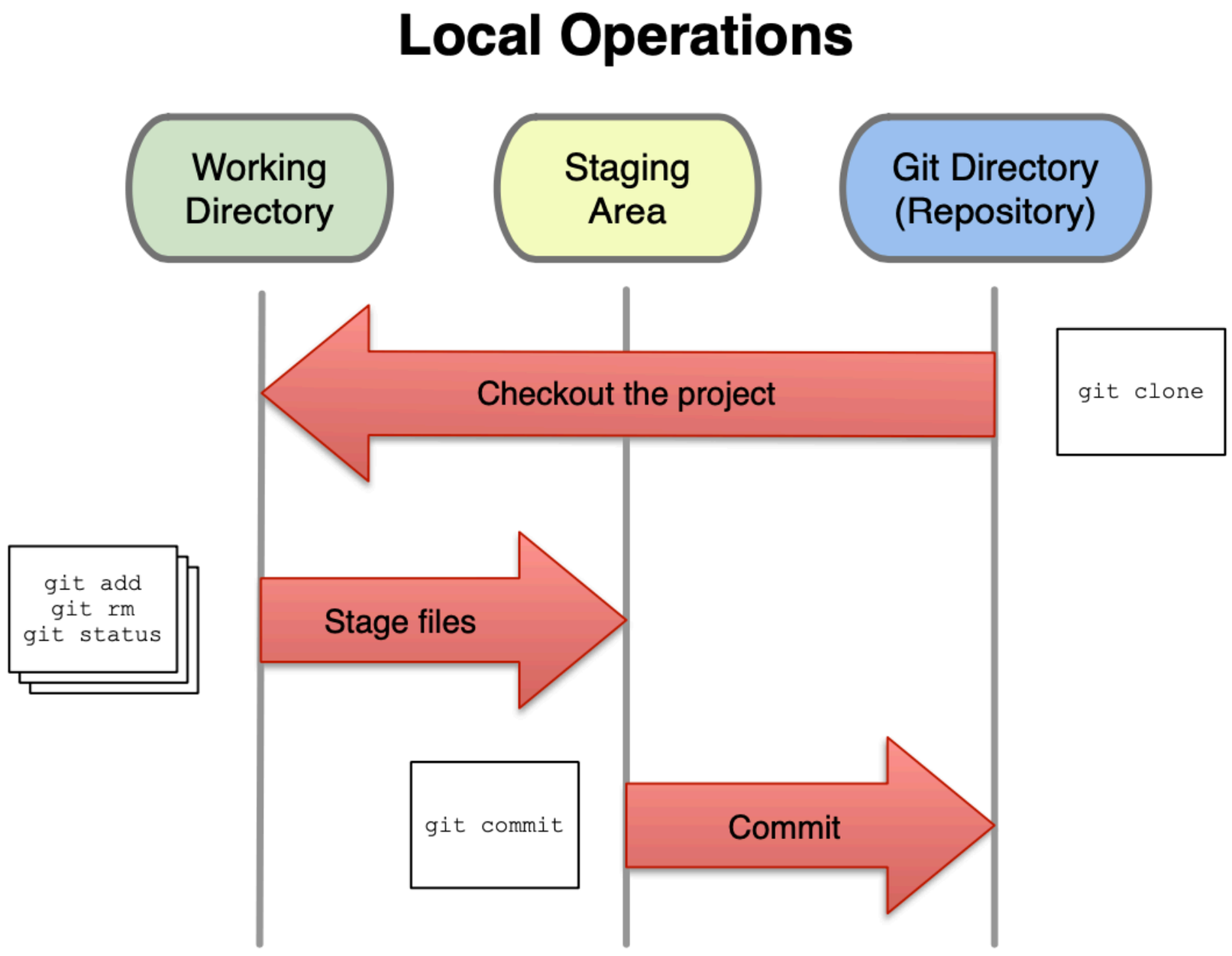
Incorporate into repository
git add FILE

```
(base) shuttle:Tecnologie Cloud e Mobile 2020 mauropelucchi$
(base) shuttle:Tecnologie Cloud e Mobile 2020 mauropelucchi$
(base) shuttle:Tecnologie Cloud e Mobile 2020 mauropelucchi$ cd unibg_mobile_and_cloud_2020/
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$ git add README.md
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$
```

Adds files to the staging area.
If a file is added once, Git is aware of it and will show it as modified instead of untracked if you have made a modification.
A modified file is not automatically staged. You have to call git add every time you want to commit the changes



The Three (local) States



Incorporate into repository
git commit -m “your message”

```
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$ git commit -m "Initial commit for Unibg 2020 course"
[master 09404a0] Initial commit for Unibg 2020 course
 1 file changed, 14 insertions(+), 2 deletions(-)
 rewrite README.md (100%)
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$
```

- The -m argument allows one to enter a message
- Without -m, git will spawn a text editor
- Use a meaningful message
- Message can have multiple lines, but make 1st line an overview

Best practice

- Use frequent, small commits
- Don't get out of sync with your collaborators
- Commit the sources, not the derived files
- Use a .gitignore file to indicate files to be ignored

git status

```
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$
```

Push the changes to GitHub
git push

```
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 987 bytes | 987.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/mauropelucchi/unibg_mobile_and_cloud_2020.git
   0747bc7..09404a0  master -> master
(base) shuttle:unibg_mobile_and_cloud_2020 mauropelucchi$
```

Check your repository

mauropelucchi / unibg_mobile_and_cloud_2020

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security | Insights | Settings

Mobile and Cloud Technologies 2020

Edit

Manage topics

2 commits | 1 branch | 0 packages | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

mauropelucchi Initial commit for Unibg 2020 course — Latest commit 09404a0 5 minutes ago

README.md — Initial commit for Unibg 2020 course — 5 minutes ago

README.md

University of Bergamo - Mobile and Cloud Technologies 2020

This repository contains code and data for the Apps developed during the Mobile & Cloud Tecnologies Course @ Univerty of Bergamo.

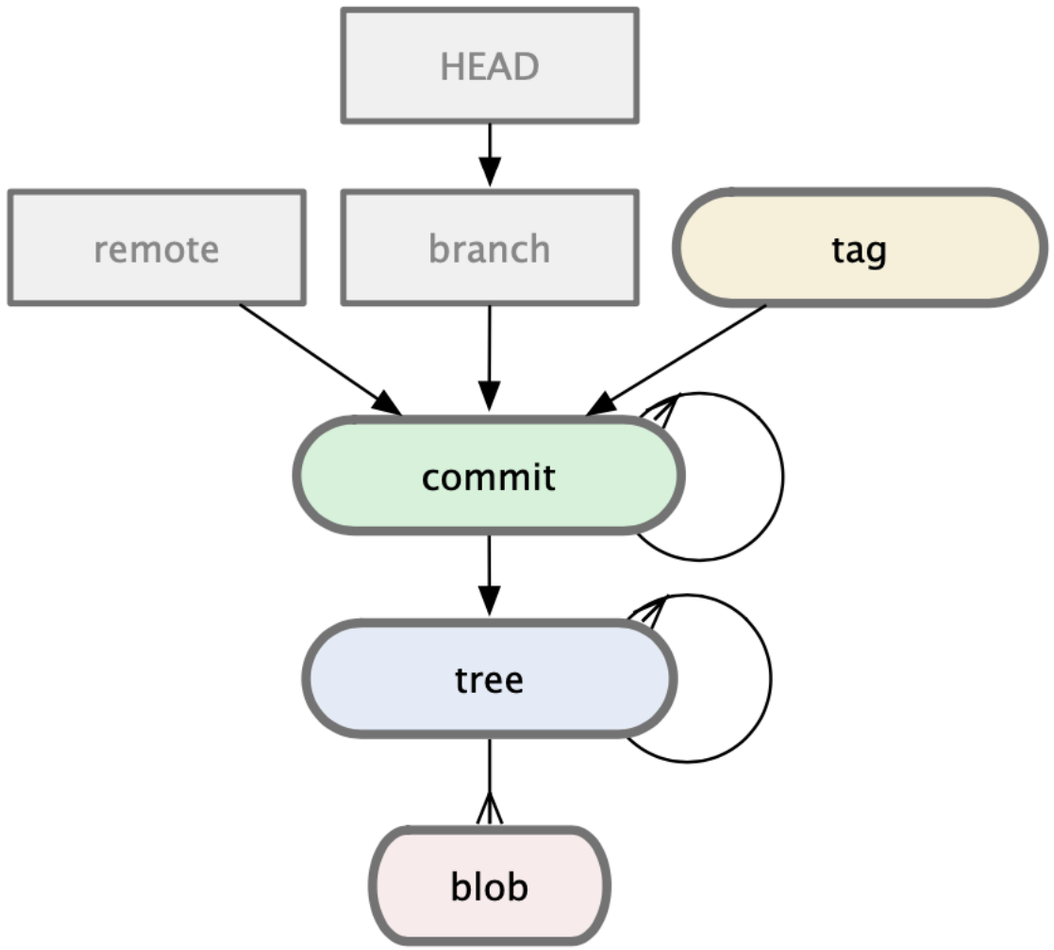
MIT License

Copyright (c) 2020 Mauro Pelucchi

Suggest a change to a repo

- Go to the repository
  - http://github.com/someone/repo
- Fork the repository
  - Click the "Fork" button
- Clone your version of it
  - git clone git@github.com:username/repo
- Change things locally, git add, git commit
- Push your changes to your GitHub repository
  - git push
- Go to your GitHub repository
  - Click "Pull Requests" and "New pull request"

Git Data Model



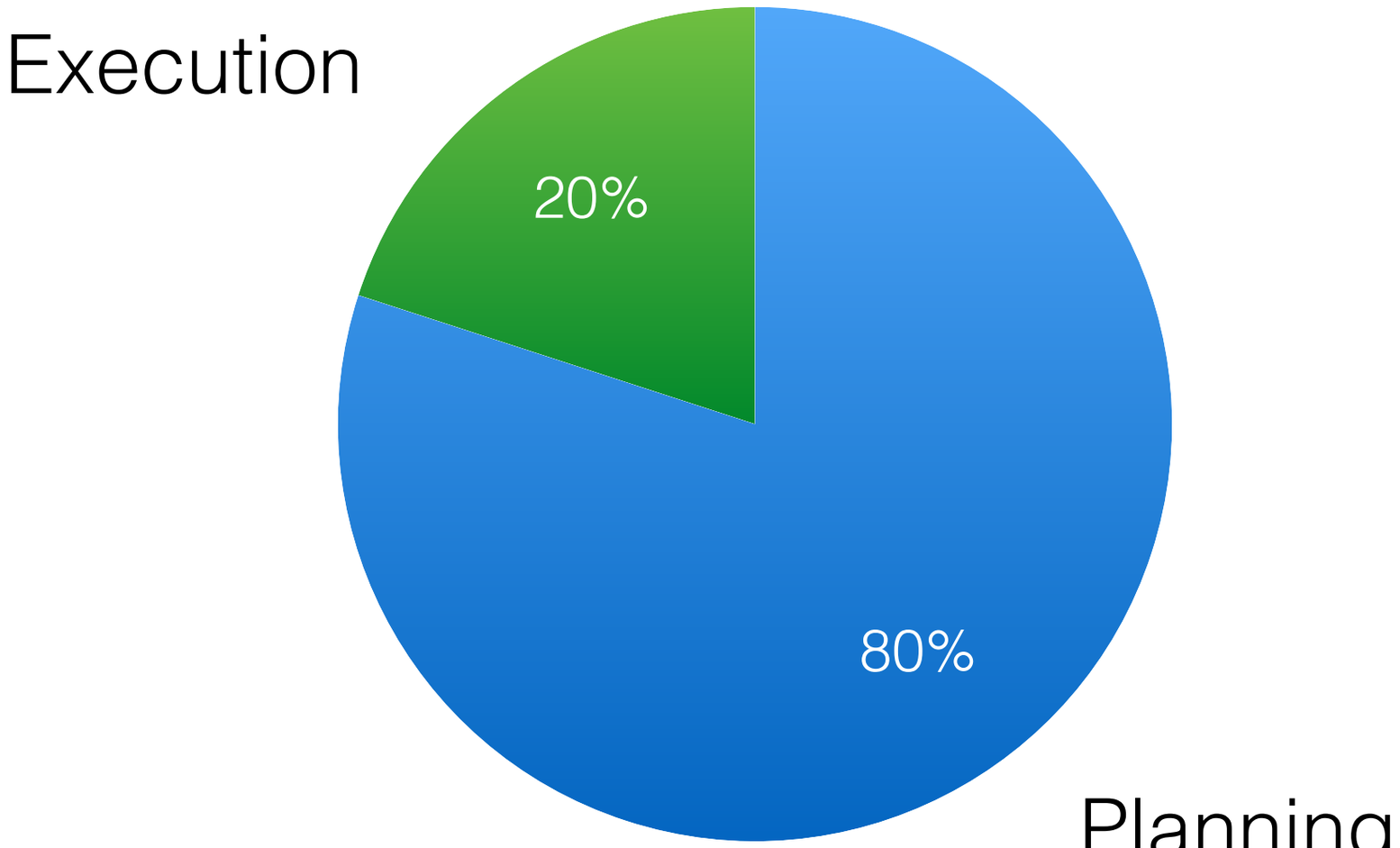
Agile basics

Cosa è un progetto e quali sono le sue caratteristiche?

Contemporary business and science treat as a project
(or program)
any undertaking, carried out individually or collaboratively and possibly involving research or design, that is carefully planned (usually by a project team) to achieve a particular aim

Wikipedia

Traditional approach



Agile Manifesto (2001)

www.agilemanifesto.org

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over **processes and tools**

Working software over **comprehensive documentation**

Customer collaboration over **contract negotiation**

Responding to change over **following a plan**

That is, while there is value in the items on the right, we value the items on the left more.

Agile principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

4. Business people and developers must work together daily throughout the project.

5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Agile principles

7. Working software is the primary measure of progress.

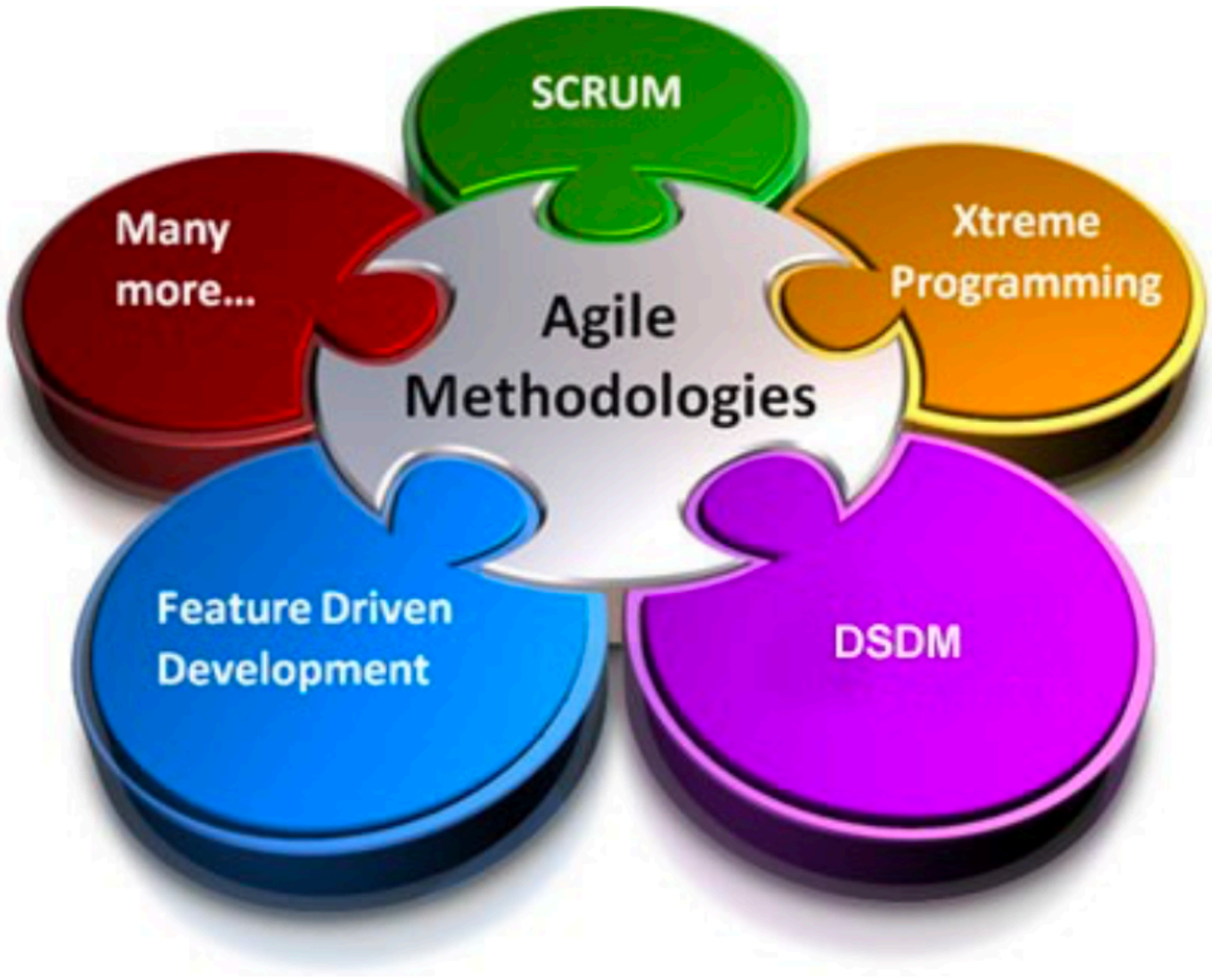
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

9. Continuous attention to technical excellence and good design enhances agility.

10. Simplicity--the art of maximizing the amount of work not done--is essential.

11. The best architectures, requirements, and designs emerge from self-organizing teams.

12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

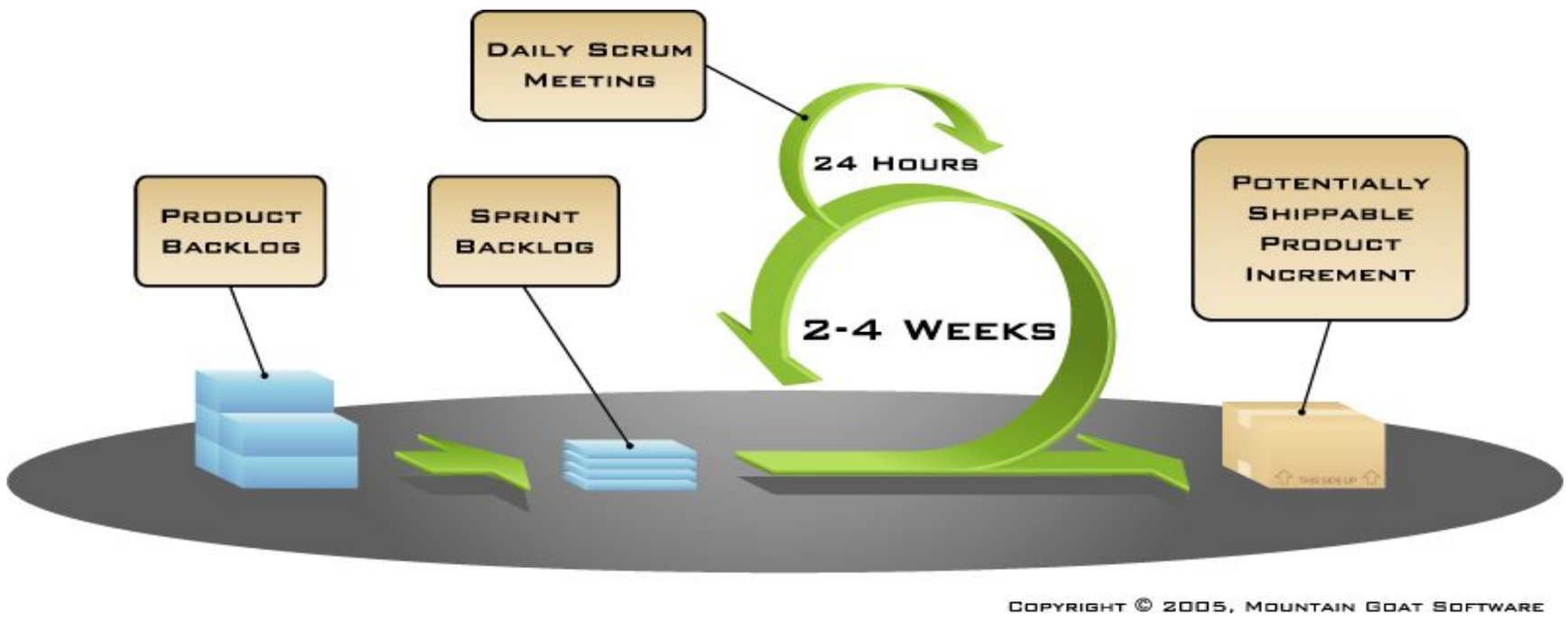

SCRUM
Many more...
Agile Methodologies
Xtreme Programming
Feature Driven Development
DSDM

Scrum

- Scrum is an Agile process;
- Used to manage complex projects since 1990;
- Delivers business functionality in 30 days;
- Business sets the priorities;
- Teams self-organize to determine the best way to deliver the highest priority features.
- Scalable to distributed, large, and long projects;
- Extremely simple but very hard!

Scrum Framework

- Sprint planning –> Definition of Done
- Sprint review – The demo
- Sprint retrospective
- Daily scrum meeting



Team

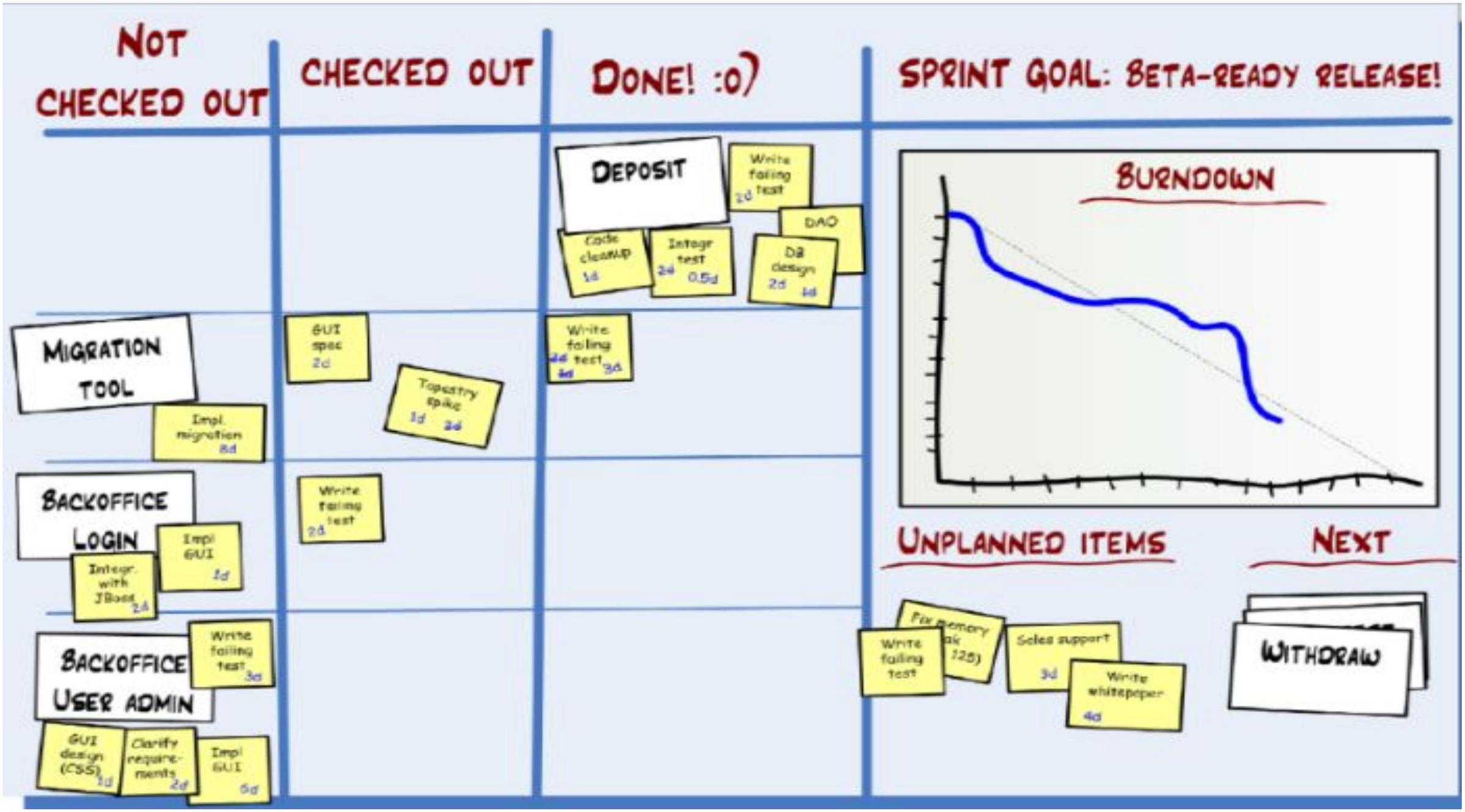
- Seven (plus/minus two) members -> Pizza Team
- Is cross-functional (Skills in testing, coding, architecture etc.)
- Selects the Sprint goal and specifies work results
- Has the right to do everything within the boundaries of the project guidelines to reach the Sprint goal
- Organizes itself and its work
- Demos work results to the Product Owner.

Team

- Scrum Master
  - Ensure that the team is fully functional and productive
  - Enable close cooperation across all roles and functions
  - Remove barriers
  - Shield the team from external interferences during the Sprint
  - Ensure that the process is followed, including issuing invitations to Daily Scrum, Sprint Review and Sprint Planning meetings.
- Product Owner
  - Define the features of the product.
  - Decide on release date and content.
  - Be responsible for the profitability of the product (ROI).
  - Prioritize features according to market value.
  - Adjust features and priority every iteration, as needed
  - Accept or reject work results.


Scrum Elements - Roles
PRODUCT OWNER
SCRUM DEV TEAM
SCRUMMASTER

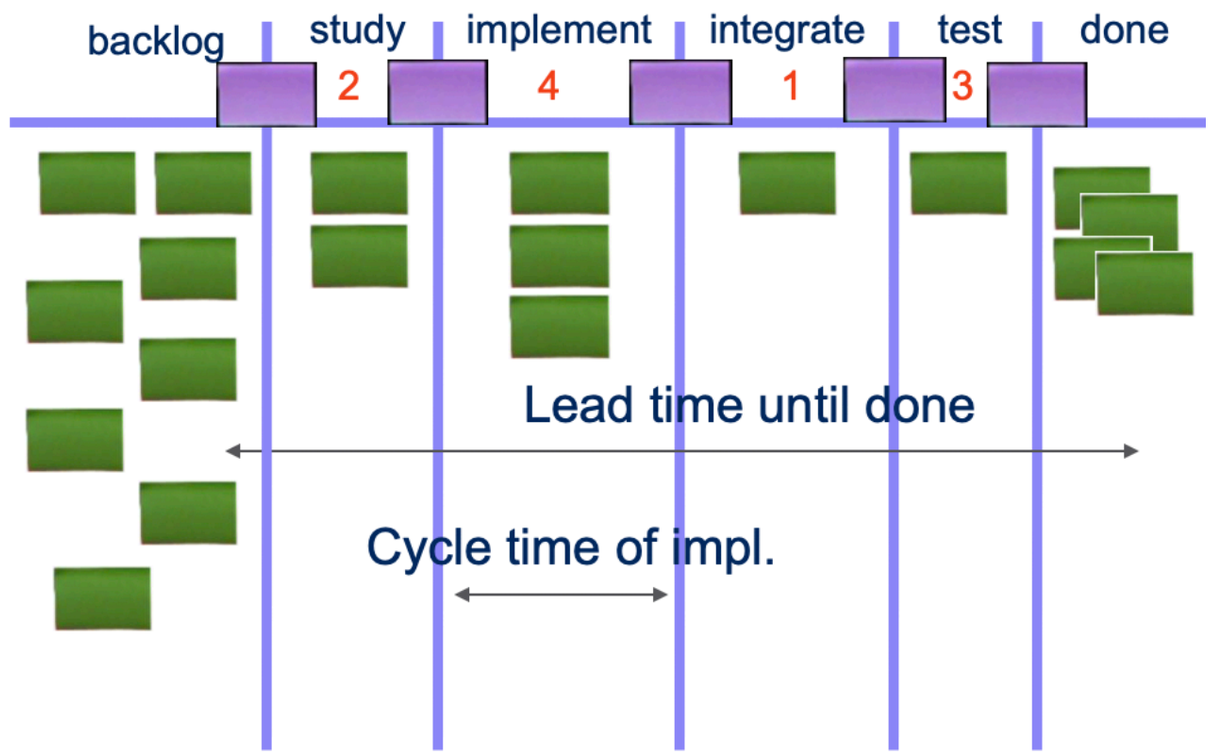
Burndown Chart



Agile at the Team Level - Kanban

Kanban

- Visualize the workflow
- Limit Work In Progress (WIP)
- Measure and manage flow
- Make process policies explicit
- Improve Collaboratively (using models/scientific method)



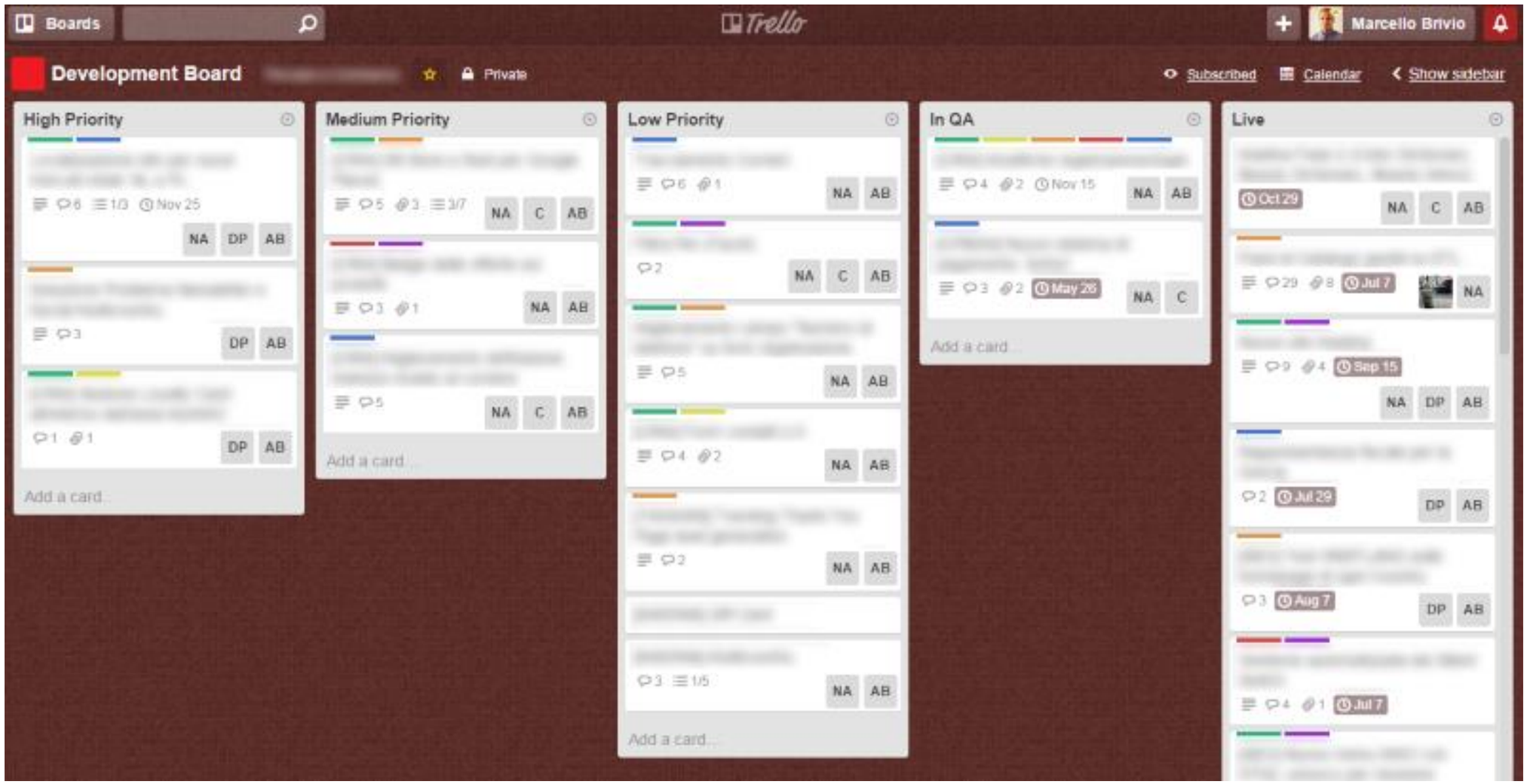
Achieve Agile Principles with DevOps Techniques

Agile Principle	DevOps techniques
1. Highest priority is satisfy the customer through early and ***continuous delivery*** of software	Continuous Delivery & Deployment
2. Welcome ***changing requirements***, even late in development	Continuous Integration and Continuous Feedback
3. Deliver working software ***frequently***, from a couple of weeks to a couple of months	Continuous Integration Continuous Deployment Continuous Feedback
4. Business people and developers must work together daily throughout the project	Integrated Development Environment

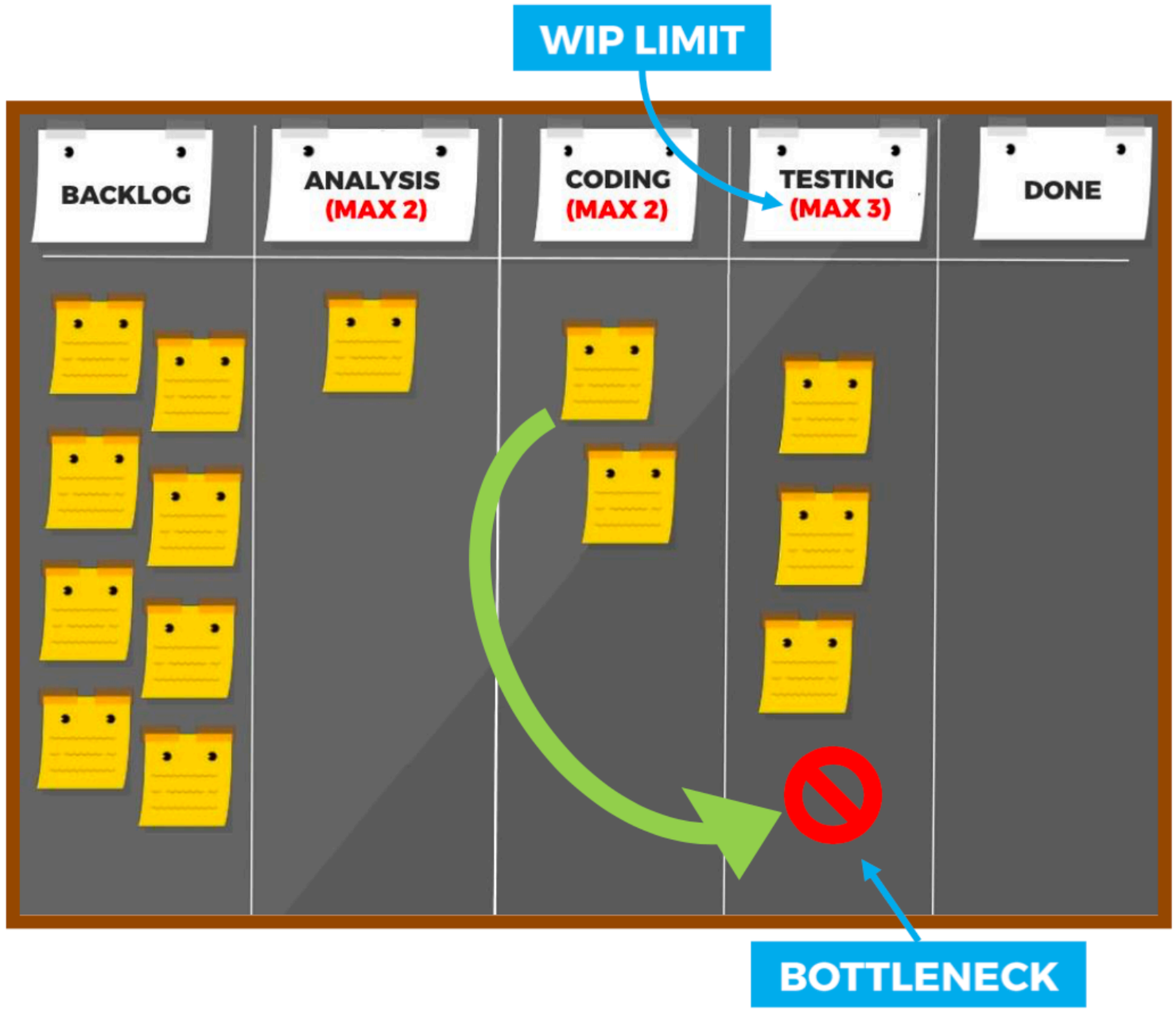
Team communication

«THE EMAIL KILLER»

Trello - Kanban



Trello - Kanban



Documentation

Design as a Service

Why service design?

The Progression of Economic Value

Source: Pine and Gilmore, 'The Experience Economy'

Service design

Best practice

- Stay focused on the customer
- Stay alert for and quickly adopt trends
- Make high-quality, high-velocity decisions

5 fundamentals

Systems

Value

People

Journeys

Proposition

Proviamo a disegnare un nuovo servizio….

TEDx

TEDx

TED is a nonprofit devoted to spreading ideas, usually in the form of short, powerful talks (18 minutes or less). TED began in 1984 as a conference where Technology, Entertainment and Design converged, and today covers almost all topics — from science to business to global issues — in more than 100 languages. Meanwhile, independently run TEDx events help share ideas in communities around the world.

https://www.ted.com/talks

3300+ talks to stir your curiosity

Find just the right one

Search talks... | Topics | Languages | Duration | **More**

Sort by: Newest

Noah Charney
The art forger who

Yifat Susskind
In uncertain times, think

Alexandra Auer
The intangible effects of

Elizabeth Gilbert
It's OK to feel

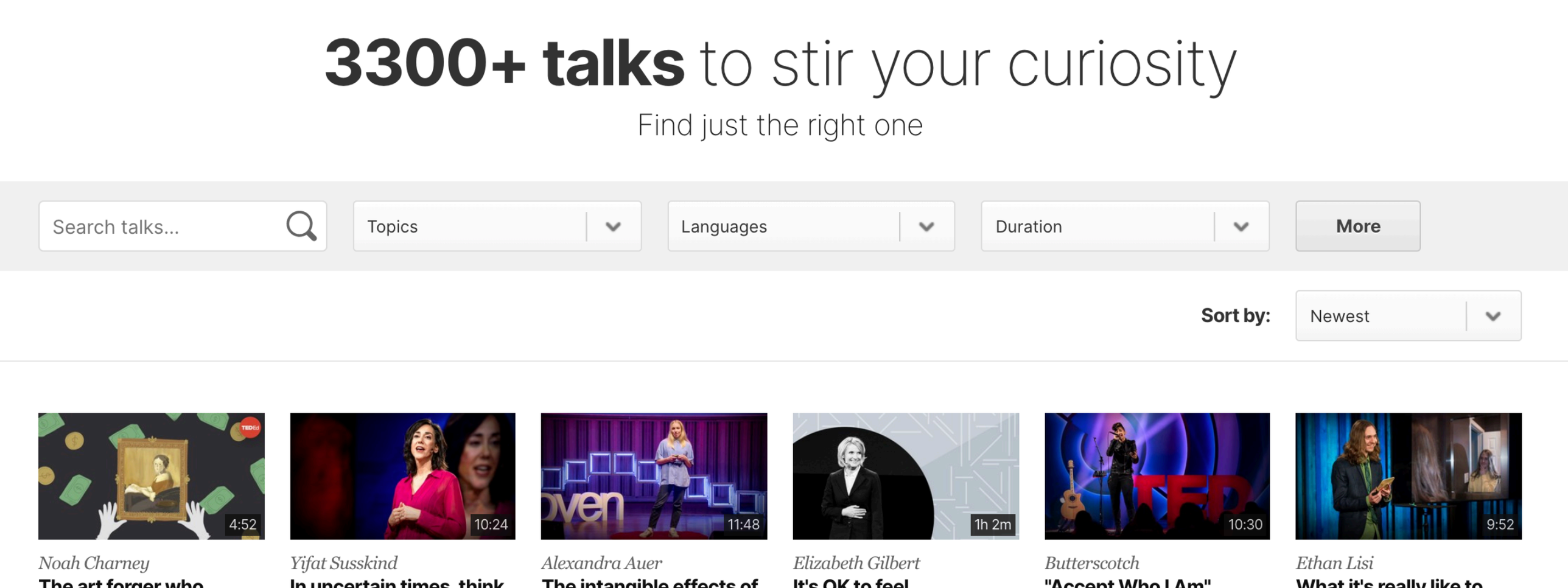
Butterscotch
"Accept Who I Am"

Ethan Lisi
What it's really like to

TEDx

https://github.com/mauropelucchi/tedx_dataset

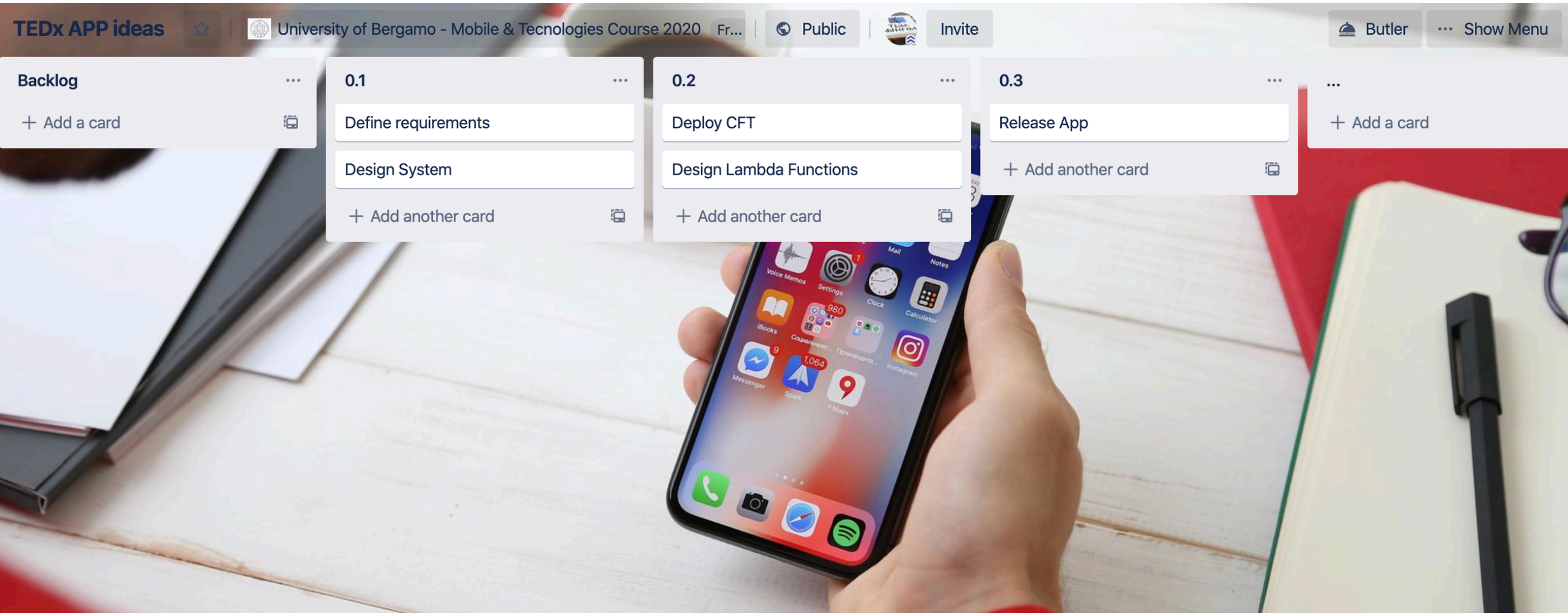
TEDx Dataset contains information about 3,000k audio-video recordings of TED Talks uploaded to the official TED.com website (update to 2020/04/04). The dataset is obtained with a scraper based on Selenium Web Driver for academic and teaching purpose. The data and the scraper has been scraped from the official TED Website and is available under the MIT license.



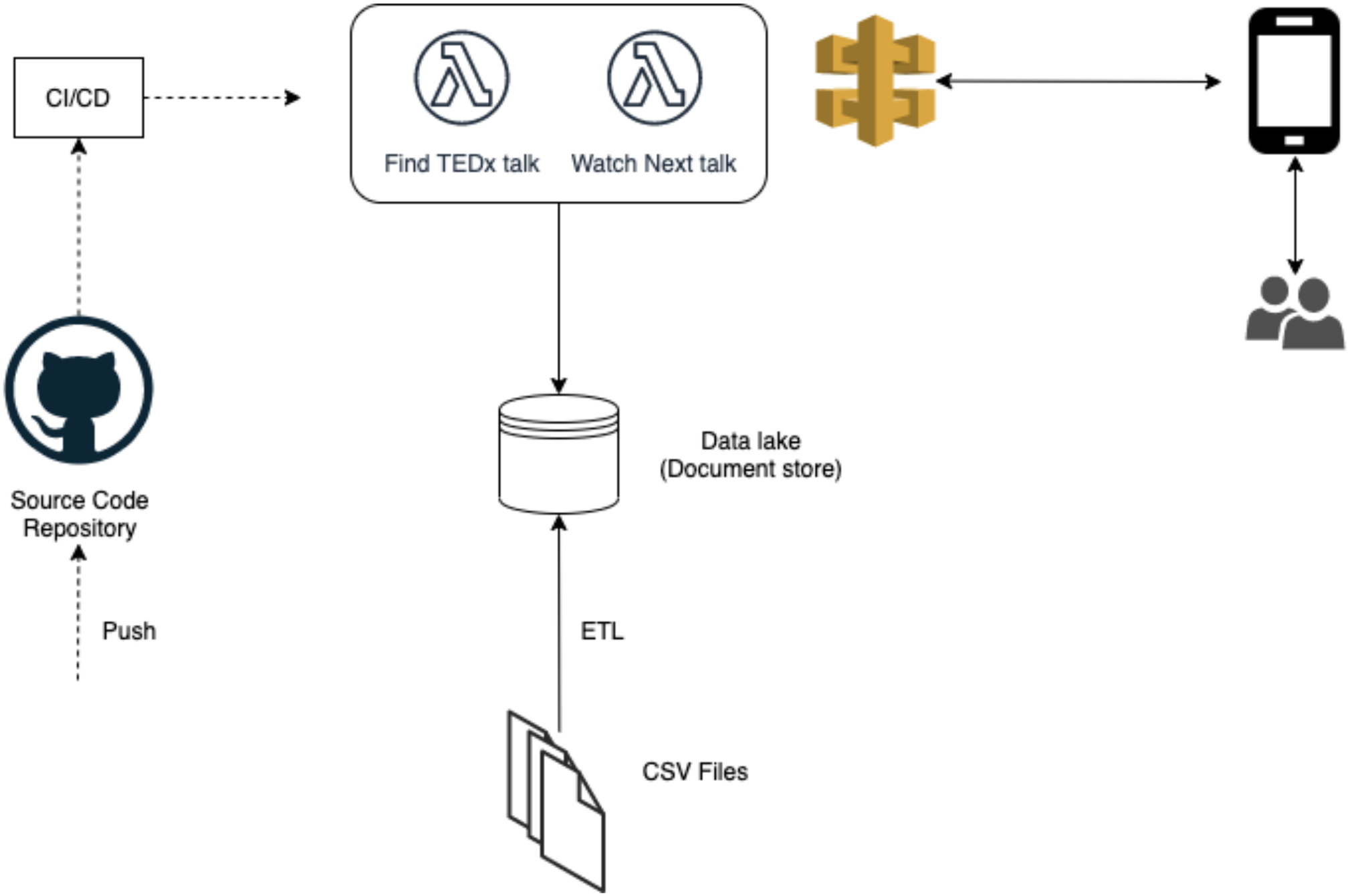
Define our idea… with Trello

www.trello.com

https://trello.com/b/5eMo7bSH



Define our systems


CI/CD
Find TEDx talk
Watch Next talk
Source Code
Repository
Push
Data lake
(Document store)
ETL
CSV Files

draw.io

Define our sprints… with Trello

www.trello.com

https://trello.com/b/supH2hqQ