USE OF CI/CD IN PROFESSIONAL SW PROJECTS: A CONCRETE EXAMPLE FROM OWN EXPERIENCE

trial lecture
presentation by
Marcus Zinn

Brief CV

Business Roles
- 5,5 years mobile leader (IIOT Mobile Apps)
- 5 years leader for patents (focusing AI and Software patents)
- 12 years software engineering
- …

Hobbies
- Lecturer since 2004 (Mobile Apps, Informatics, Software Engineering, Data Warehouse, ..)
- Support / Supervision for bachelor / master thesis (28)
- IIOT Beekeeping
- AI based image generation (Stable diffusion / comfyUI)
- …

Good morning!

AIMS

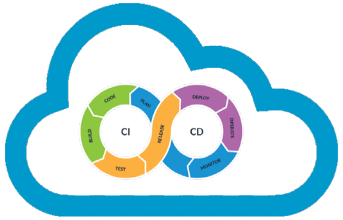
teaching objectives

1. CI/CD explain the concept
2. CI/CD how to (not) start
3. CI/CD todays view

Other objectives

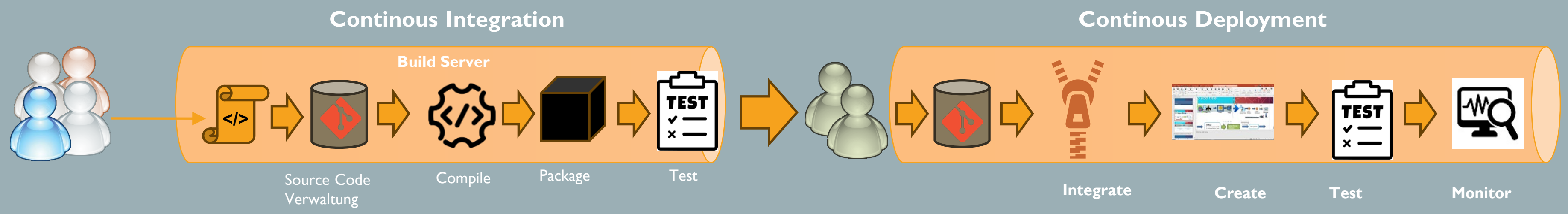
- Show my englisch presentation skills for the professorship
- Show my way of handling lecture topics e.g. exam preparation
- Some ideas for CI/CD bachelor thesis topics

TOPICS



- Part I – Introduction into Continuous Integration/Continuous Deployment
- Part II – Real world scenario „Consolidate 180 Apps into 3“
- Part III – Successful and unsuccessful approaches of CI/CD in the real-world scenario
- Part IV - Best Practices CI/CD in the Cloud + Live Demo
- Part V – Summary and future CI/CD topics

PART I – INTRODUCTION INTO CONTINUOUS INTEGRATION/CONTINUOUS DEPLOYMENT



Continous Integration

Build Server

Source Code Verwaltung

Compile

Package

Test

Continous Deployment

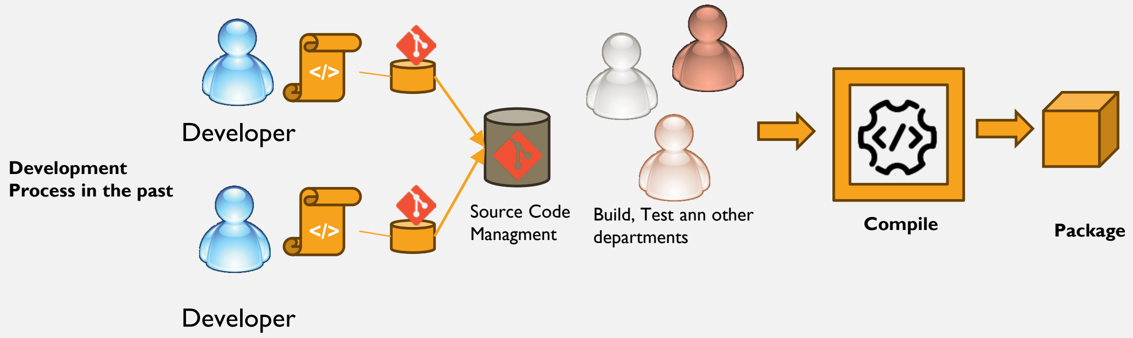
Integrate

Create

Test

Monitor

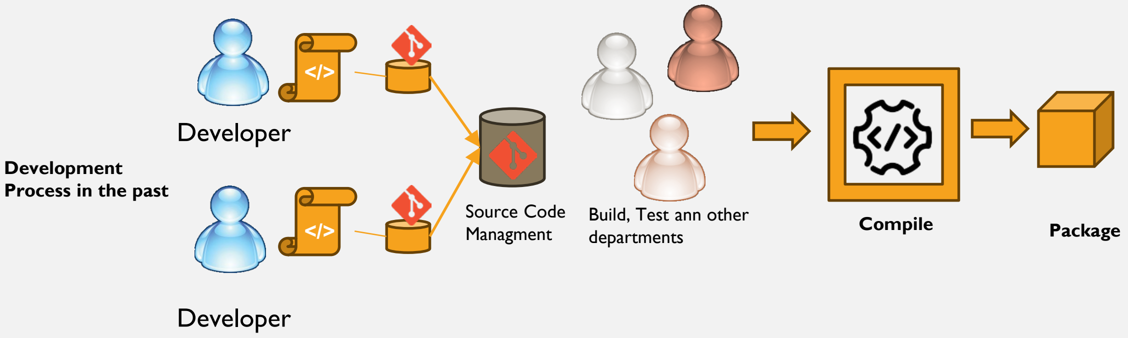
THE (HISTORICAL) CHALLENGES IN DEVELOPMENT PROCESS PART I


Development Process in the past
Developer
Developer
Source Code Managment
Build, Test ann other departments
Compile
Package

Typical challenges in development process

- **Scaling Challenges:** As team size increased, manual integration processes became more cumbersome and harder to manage.
- **Underutilized Resources:** Resources may have been wasted as developers spent time manually integrating code instead of working on new features.
- **Inconsistent Build Environments:** Different development environments and build processes led to inconsistencies and compatibility issues.
- **Difficulty in Code Reuse:** Without a consistent integration strategy, it was challenging to effectively reuse code and maintain shared libraries.
- **Lack of Development Continuity:** Without Continuous Integration, there may have been no clear continuity in the development process, leading to unpredictable outcomes and project delays.

THE (HISTORICAL) CHALLENGES IN DEVELOPMENT PROCESS PART 2

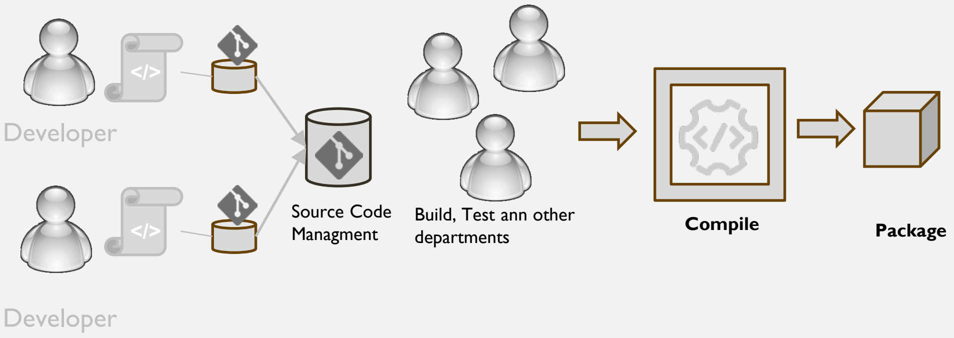

Development Process in the past
Developer
Developer
Source Code Managment
Build, Test ann other departments
Compile
Package

Typical challenges in development process

- **Manual Integration Processes:** Manual integration processes often led to human errors and prolonged development cycles.
- **Code Fragmentation:** Developers often worked on isolated codebases, leading to fragmentation and incompatibility.
- **Difficulty in Debugging:** Bugs were often discovered late in the development cycle, making debugging time-consuming and complex.
- **Lack of Transparency:** There was often no clear view of the status of the code and integrations, leading to uncertainties within the team.
- **Long Feedback Loops:** Without continuous integration, there were long wait times for feedback, affecting efficiency and productivity.

THE (HISTORICAL) CHALLENGES IN DEPLOYMENT PROCESS

Development Process in the past



Developer

Developer

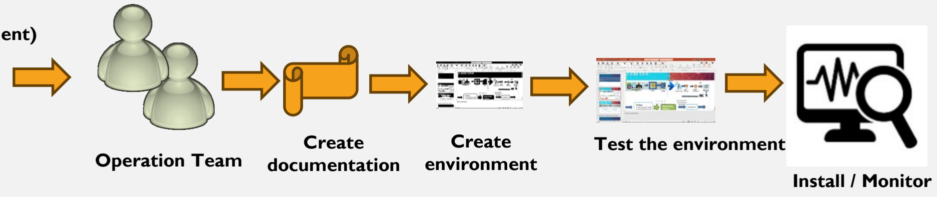
Source Code Managment

Build, Test ann other departments

Compile

Package

Operation (Deplyoment) Process in the past



Operation Team

Create documentation

Create environment

Test the environment

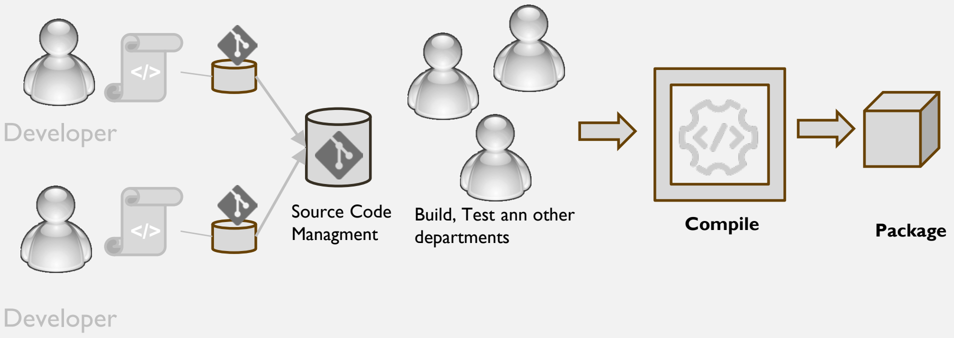
Install / Monitor

Typcial challanges in development process

- **Manual Deployment Processes:** Manual deployment processes were error-prone and time-consuming, leading to delays and inconsistencies in releasing software.
- **Deployment Bottlenecks:** Manual approval processes and dependencies on specific individuals for deployment caused bottlenecks and slowed down the release cycle.
- **Limited Visibility:** Lack of visibility into the deployment process made it difficult to track the status of deployments and identify issues quickly.
- **Risk of Human Error:** Manual deployments increased the risk of human error, such as deploying incorrect versions or configurations, leading to downtime and customer dissatisfaction.
- **Inconsistent Environments:** Differences between development, testing, and production environments often caused deployment issues due to lack of consistency.

THE (HISTORICAL) CHALLENGES IN DEPLOYMENT PROCESS

Development Process in the past



Developer

Developer

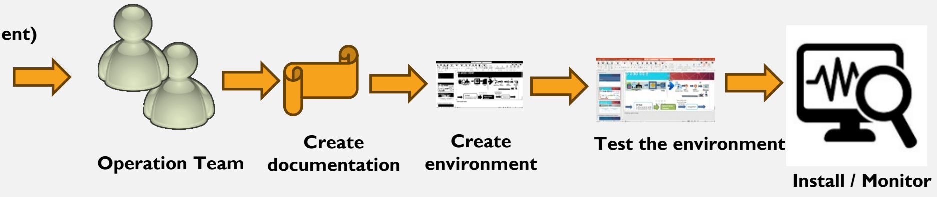
Source Code Managment

Build, Test ann other departments

Compile

Package

Operation (Deplyoment) Process in the past



Operation Team

Create documentation

Create environment

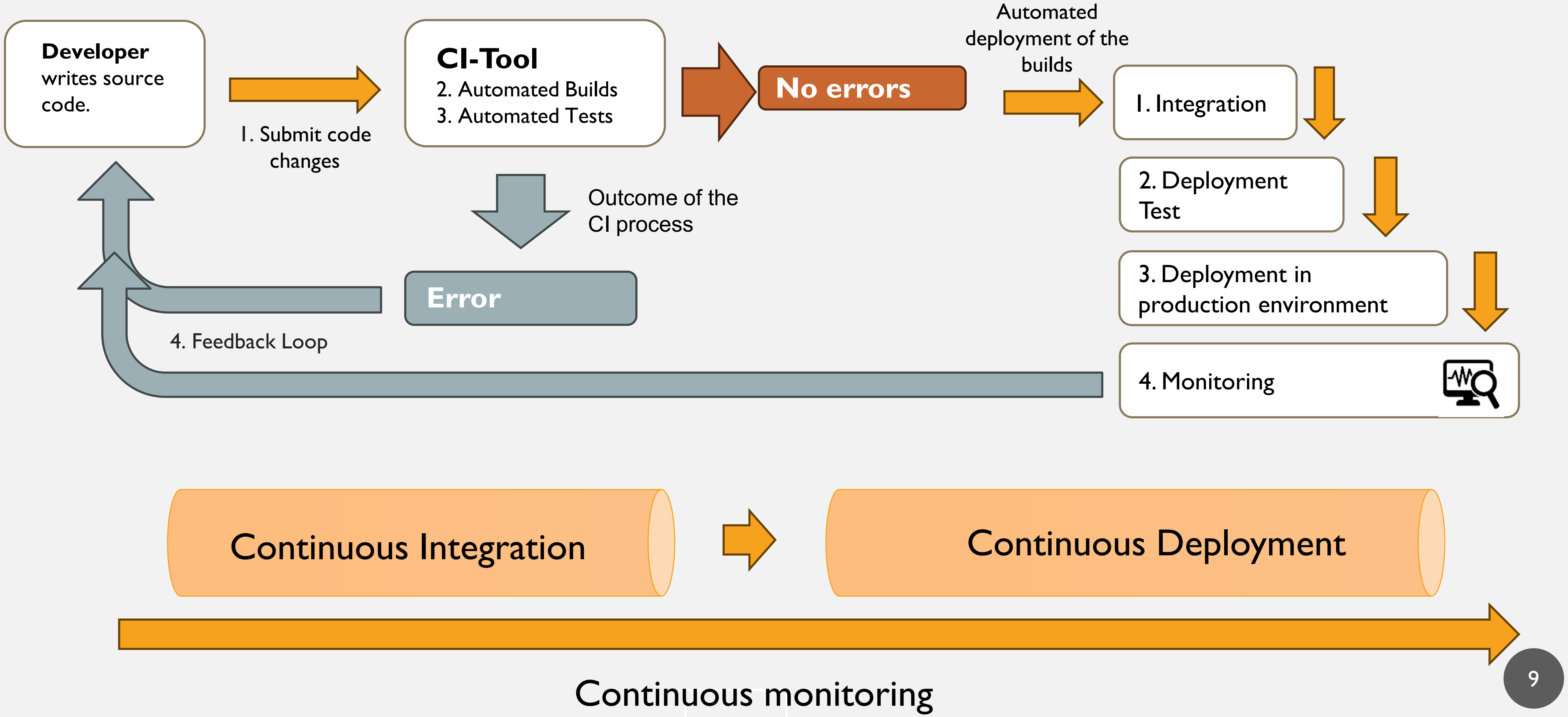
Test the environment

Install / Monitor

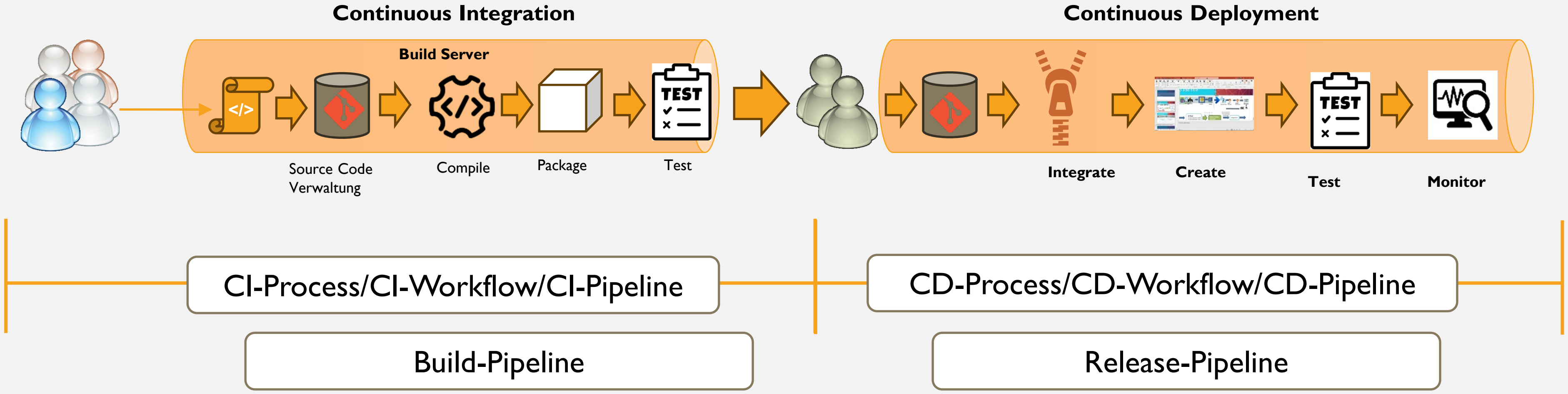
Typcial challanges in development process

1. **Delayed Feedback:** Without Continuous Deployment, feedback on new features and bug fixes was delayed, impacting the ability to iterate quickly and respond to customer needs.
2. **Deployment Rollbacks:** Manual deployment rollbacks were complex and time-consuming, making it challenging to revert to a stable state in case of issues.
3. **Deployment Coordination:** Coordinating deployments across teams and environments was challenging and prone to miscommunication, leading to deployment failures.
4. **Compliance and Security Risks:** Manual deployment processes made it difficult to enforce compliance and security policies consistently across environments.
5. **Limited Release Frequency:** Manual deployment processes limited the frequency of releases, hindering the ability to deliver value to customers rapidly and stay competitive in the market.

CONTINIOUS INTEGRATION/DEPLYEMENT AS SOLUTION CONCEPT



WHAT IS A PIPELINE?



Continuous Integration

Build Server

Source Code Verwaltung

Compile

Package

Test

Continuous Deployment

Integrate

Create

Test

Monitor

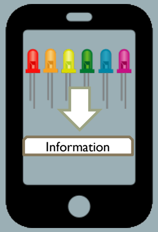
CI-Process/CI-Workflow/CI-Pipeline

Build-Pipeline

CD-Process/CD-Workflow/CD-Pipeline

Release-Pipeline

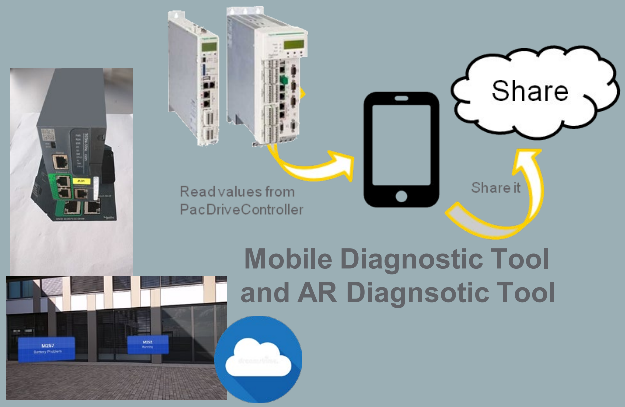
PART II – REAL WORLD SCENARIO „CONSOLIDATE 180 APPS INTO 3“



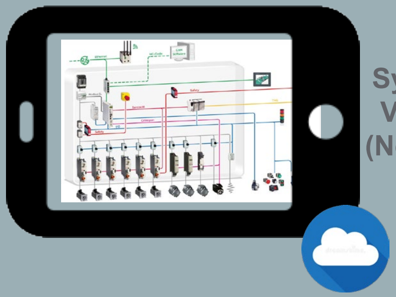
LED Blinking Scanner

MotionSizer

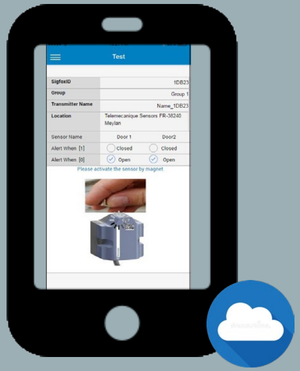
Drive Mobile KeyPad



Mobile Diagnostic Tool and AR Diagnsotic Tool

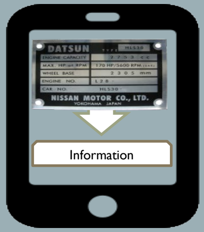


System Viewer (Netplan)



C2C & eXLhoist

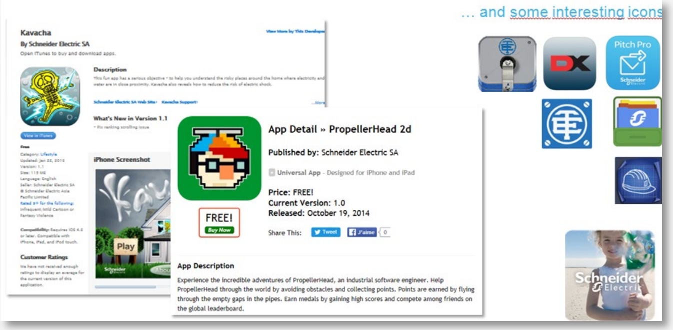
Engine Plate Scanner



WHAT HAPPENED?

- What happened ?
  - Apple called!

- Your 400 apps are
  - Partly not used
  - Partly the same
  - Partly not follow our rules



... and some interesting icons

Kavacha
By Schneider Electric SA
Open iTunes to buy and download apps.

View More by This Developer

Description
This fun app has a serious objective – to help you understand the risky places around the home where electricity and water are in close proximity. Kavacha also reveals how to reduce the risk of electric shock.

Schneider Electric SA Web Site › Kavacha Support › ...More

What's New in Version 1.1
- Fix ranking scrolling issue

View in iTunes

Free
Category: Lifestyle
Updated: Jan 22, 2014
Version: 1.1
Size: 115 MB
Language: English
Seller: Schneider Electric SA
© Schneider Electric Asia Pacific Limited
Rated 9+ for the following:
Infrequent/Mild Cartoon or Fantasy Violence

Compatibility: Requires iOS 4.3 or later. Compatible with iPhone, iPad, and iPod touch.

Customer Ratings
We have not received enough ratings to display an average for the current version of this application.

iPhone Screenshot

App Detail » PropellerHead 2d

Published by: Schneider Electric SA

Universal App - Designed for iPhone and iPad

Price: FREE!
Current Version: 1.0
Released: October 19, 2014

FREE! Buy Now

Share This: Tweet J'aime 0

App Description
Experience the incredible adventures of PropellerHead, an industrial software engineer. Help PropellerHead through the world by avoiding obstacles and collecting points. Points are earned by flying through the empty gaps in the pipes. Earn medals by gaining high scores and compete among friends on the global leaderboard.

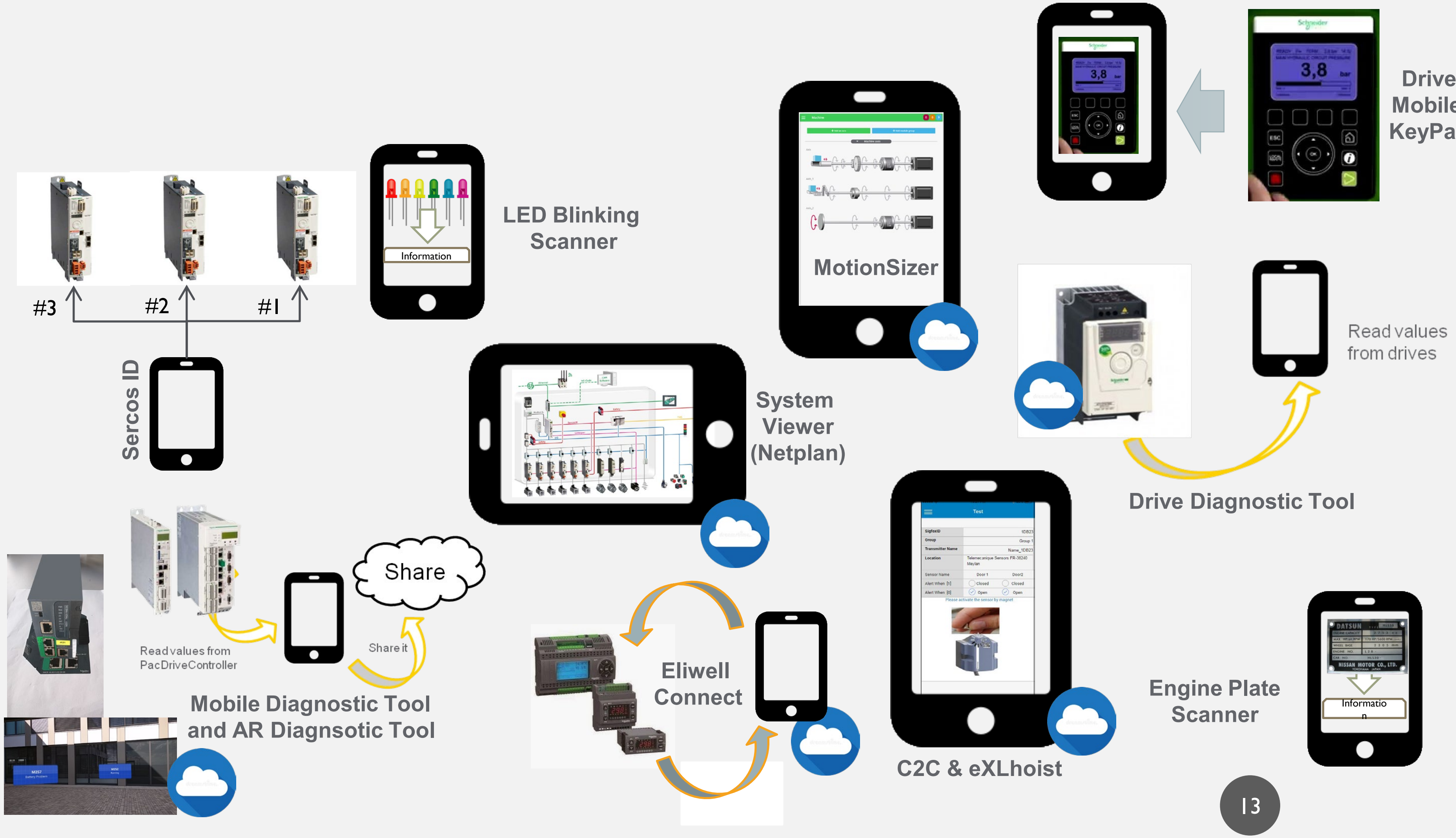
App Examples

- Why is this a challenge?
  - Technology incompatible
  - No Funding
  - Project closed already
  - No Experts
  - No Documentation
  - Architecture incompatible
  - ....

- First solutions
  - Build a corporate group
  - Agreement with apple on consolidation plan

SOLUTION ARCHITECTURE SUMMARY

- **Analyze existing and planned features domains (IIOT/I4.0/COM/DIAG)**
- Define 3 Master apps a solution domains
- Define technology and base architecture
- Create IIOT Framework
- Create App Architecture
- Create MicroService / Cloud Application and Service



LED Blinking Scanner

MotionSizer

Drive Mobile KeyPa

System Viewer (Netplan)

Drive Diagnostic Tool

Mobile Diagnostic Tool and AR Diagnsotic Tool

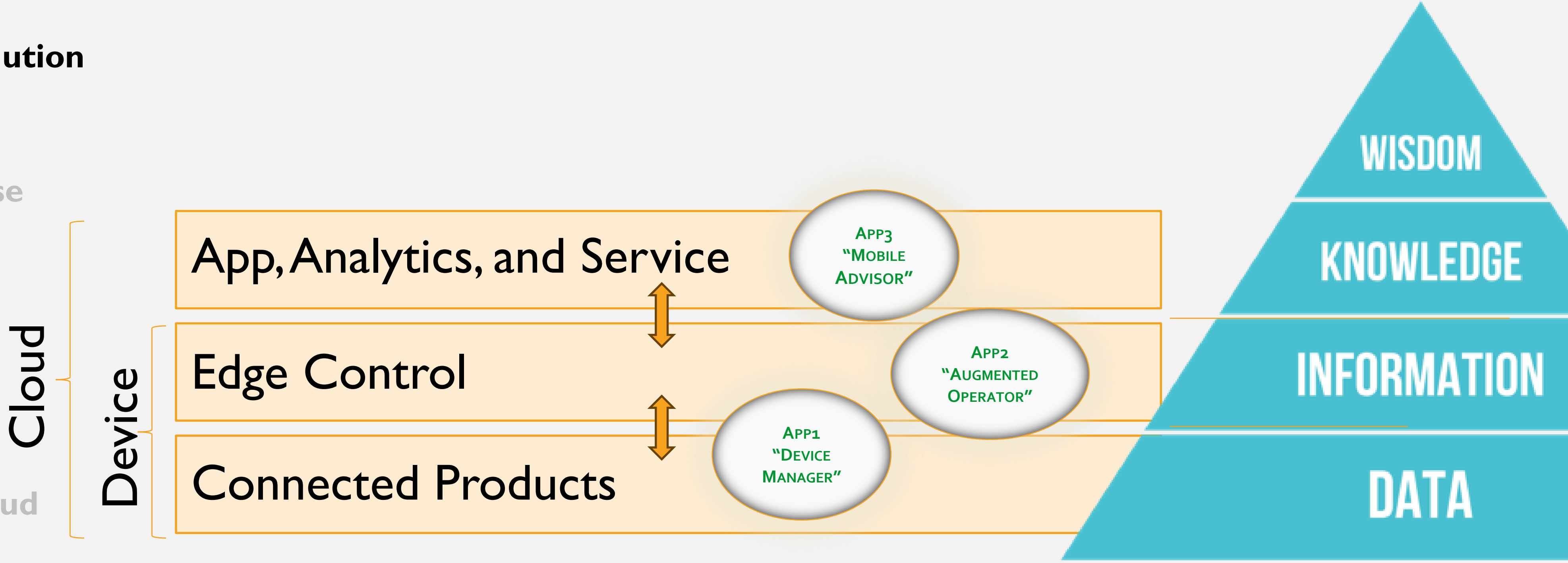
Eliwell Connect

C2C & eXLhoist

Engine Plate Scanner

SOLUTION ARCHITECTURE SUMMARY

- Analyze existing and planned features domains (IIOT/I4.0/COM/DIAG)
- **Define 3 Master apps a solution domains**
- Define technology and base architecture
- Create IIOT Framework
- Create App Architecture
- Create MicroService / Cloud Application and Service



Cloud

Device

App, Analytics, and Service

Edge Control

Connected Products

App3 "Mobile Advisor"

App2 "Augmented Operator"

App1 "Device Manager"

WISDOM

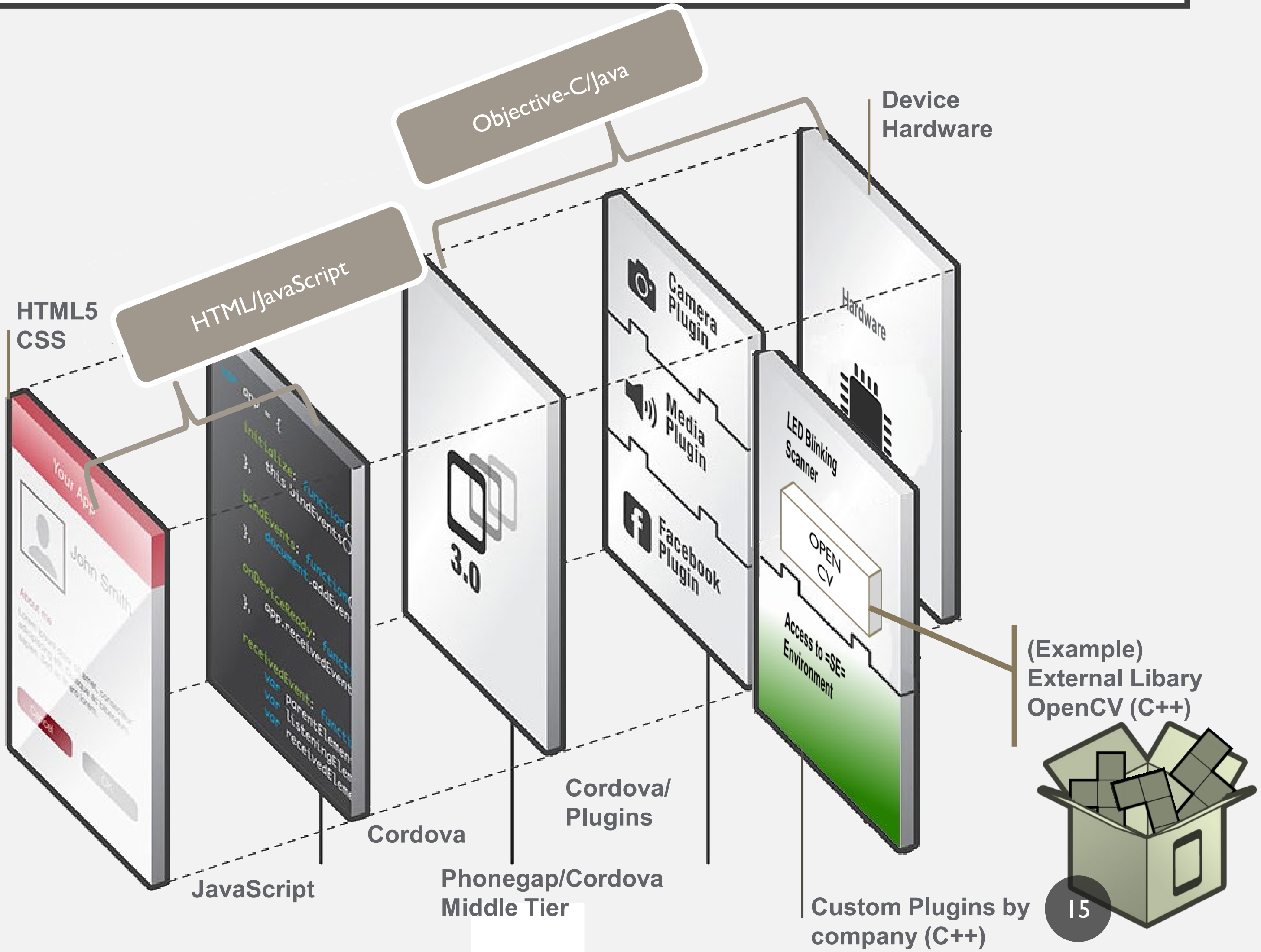
KNOWLEDGE

INFORMATION

DATA

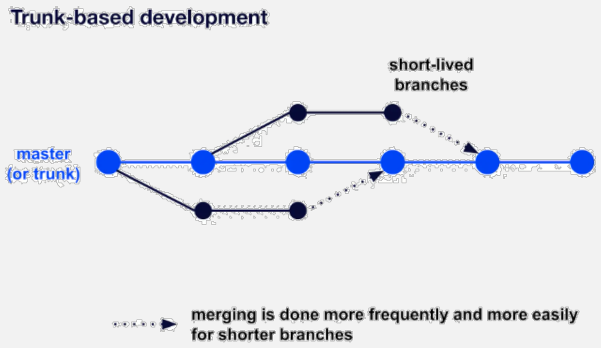
SOLUTION ARCHITECTURE SUMMARY

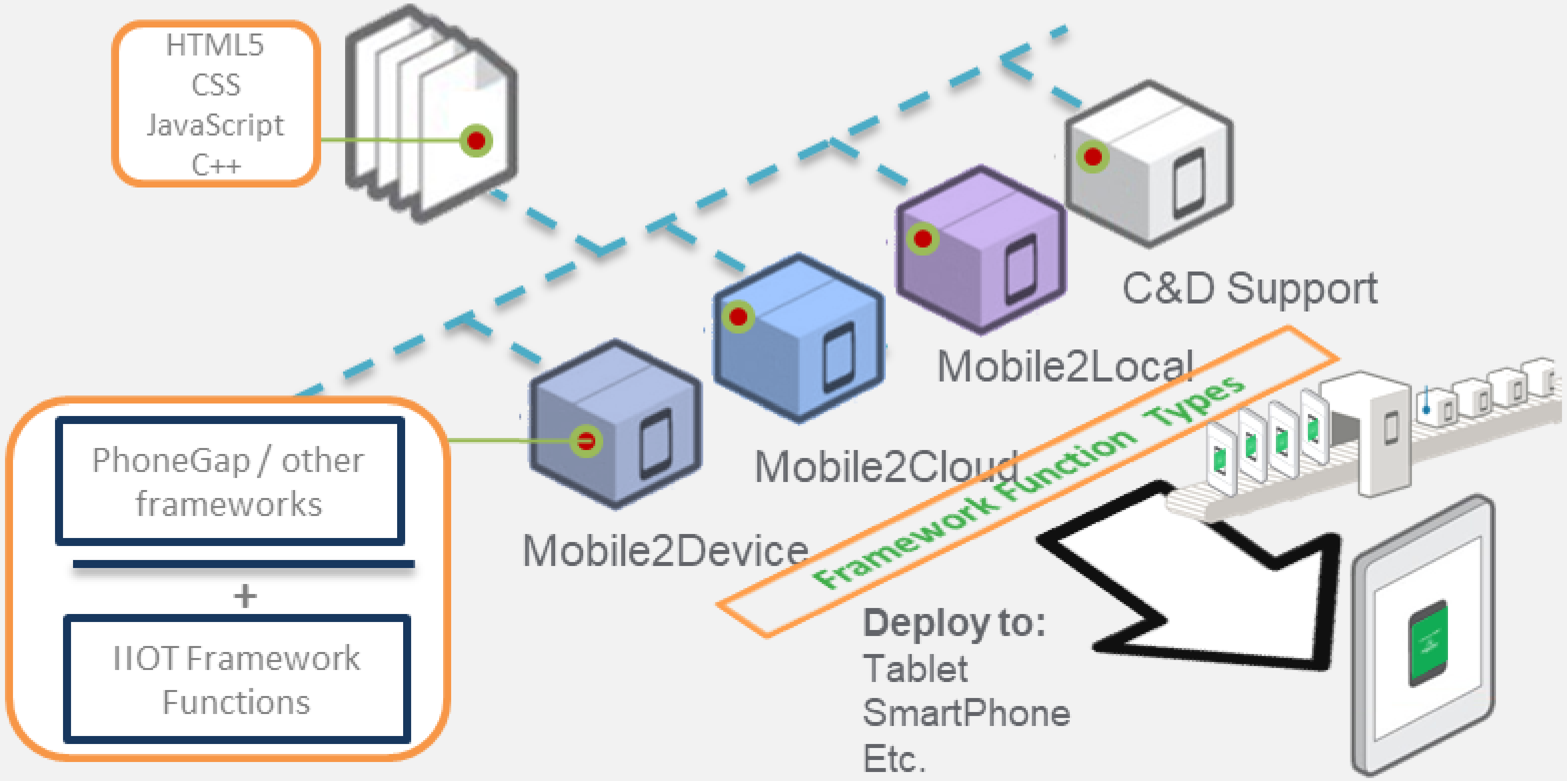
- Analyze existing and planned features domains (IIOT/I4.0/COM/DIAG)
- Define 3 Master apps a solution domains
- **Define technology and base architecture**
- Create IIOT Framework
- Create App Architecture
- Create MicroService / Cloud Application and Service



SOLUTION ARCHITECTURE SUMMARY

- Analyze existing and planned features domains (IIOT/I4.0/COM/DIAG)
- Define 3 Master apps a solution domains
- Define technology and base architecture
- **Create IIOT Framework**
- Create App Architecture
- Create MicroService / Cloud Application and Service


Trunk-based development
short-lived branches
master (or trunk)
merging is done more frequently and more easily for shorter branches


HTML5
CSS
JavaScript
C++

PhoneGap / other frameworks

+

IIOT Framework Functions

Mobile2Device

Mobile2Cloud

Mobile2Local

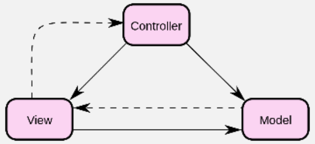
C&D Support

Framework Function Types

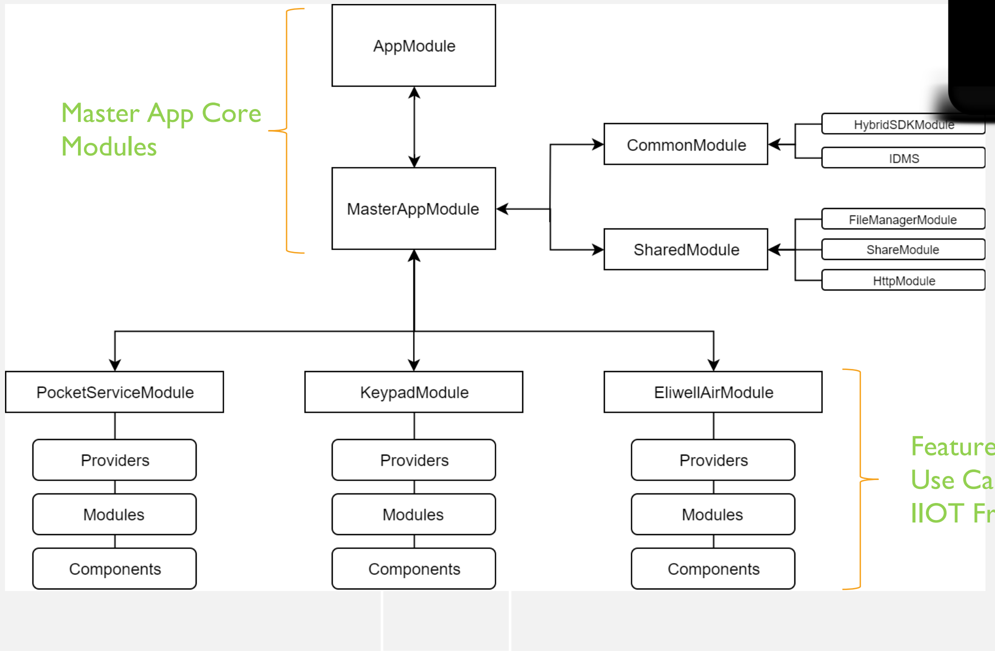
Deploy to:
Tablet
SmartPhone
Etc.

SOLUTION ARCHITECTURE SUMMARY

- Analyze existing and planned features domains (IIOT/I4.0/COM/DIAG)
- Define 3 Master apps a solution domains
- Define technology and base architecture
- Create IIOT Framework
- **Create App Architecture**
- Create MicroService / Cloud Application and Service



Model-View-Control

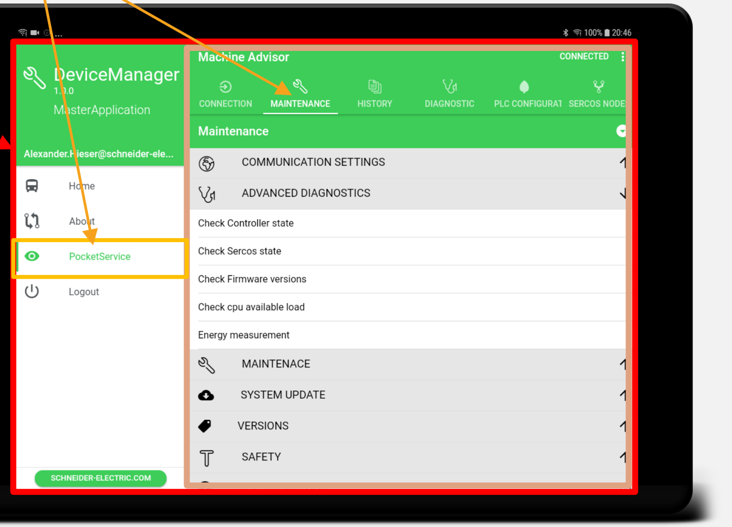


Shared Libaries

Features (Customer Use Case) using IIOT Framework

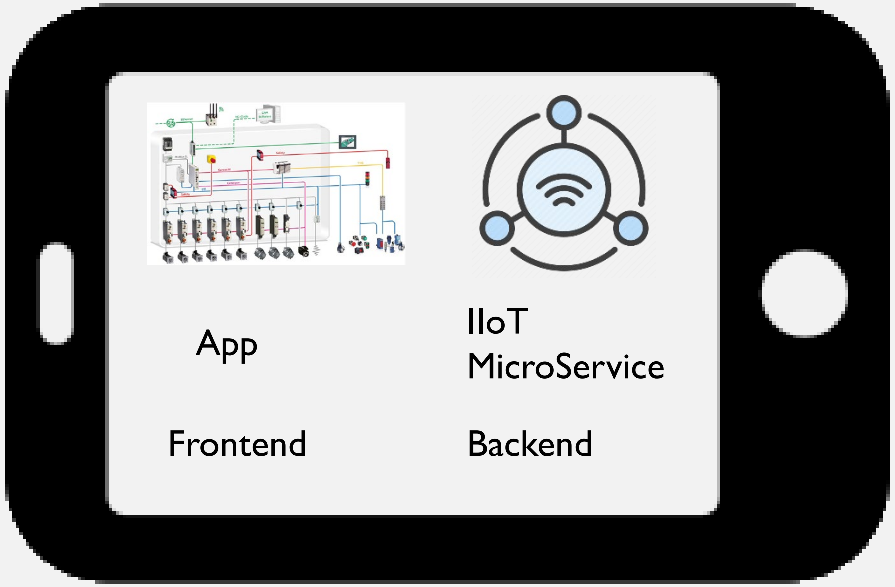
Feature

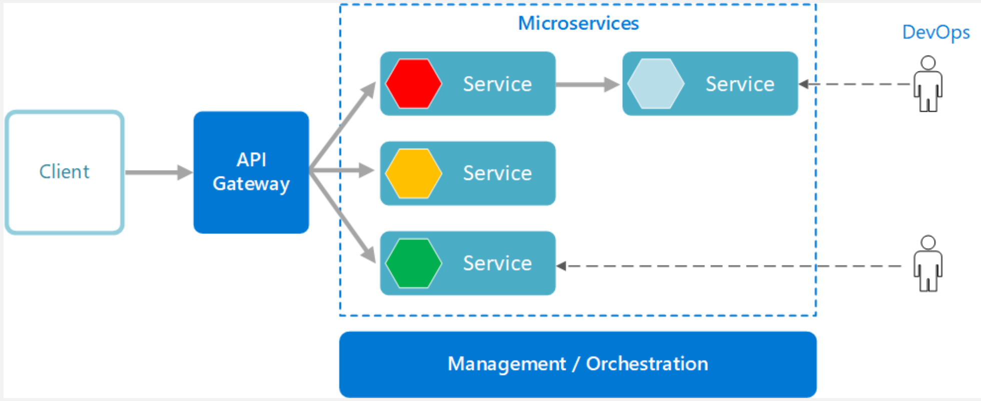
MAPP-Shell



SOLUTION ARCHITECTURE SUMMARY

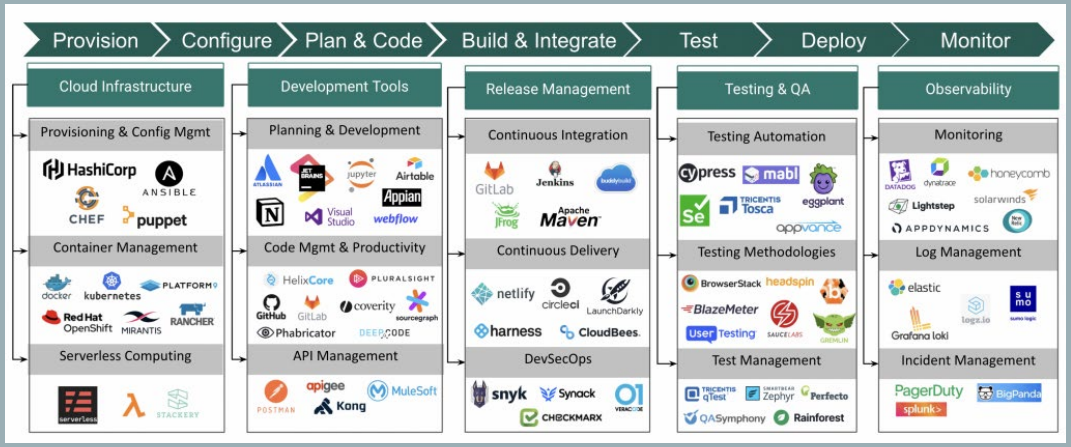
- Analyze existing and planned features domains (IIOT/I4.0/COM/DIAG)
- Define 3 Master apps a solution domains
- Define technology and base architecture
- Create IIOT Framework
- Create App Architecture
- **Create MicroService** / Cloud Application and Service

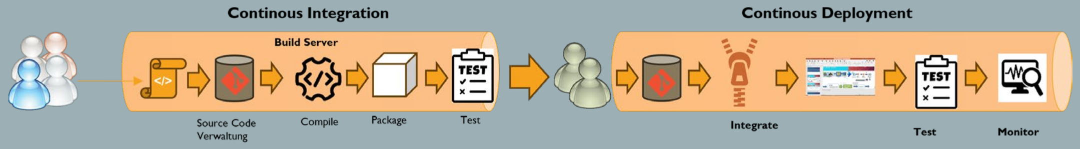




MicroService Architecture (Quelle: Microsoft, 2024)

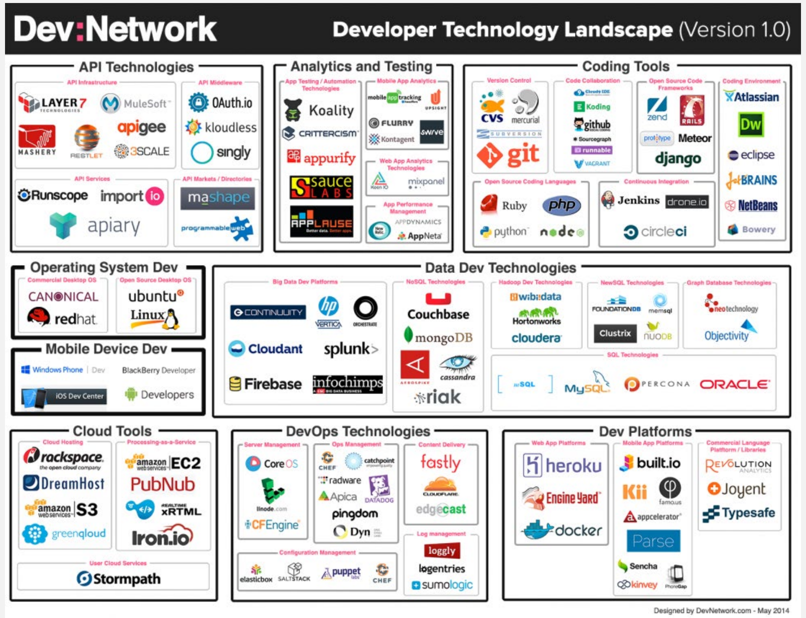
PART III – SUCCESSFUL AND UNSUCCESSFUL APPROACHES OF CI/CD IN THE REAL-WORLD SCENARIO

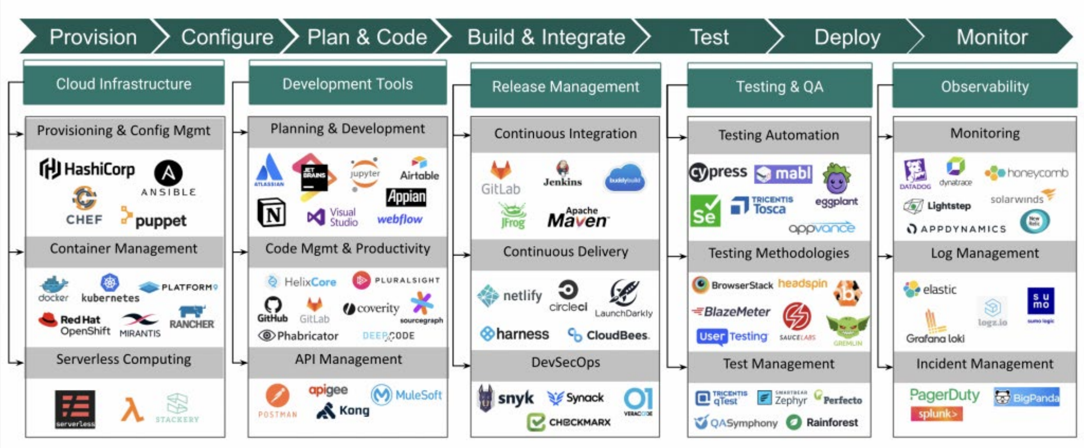




- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.

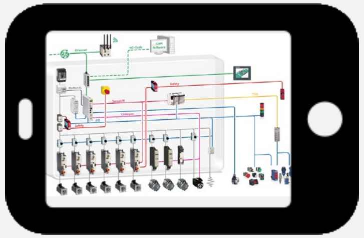
CI/CD IN MOBILE APP DEVELOPMENT - ANALYSIS OF DEVELOPMENT PRACTICES OF OLD & NEW PROJECTS

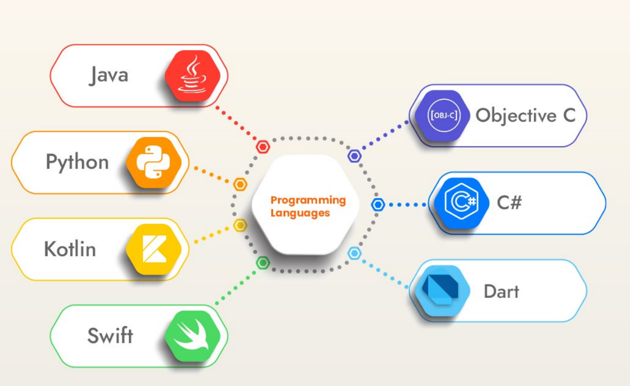




- Analysis of old/new projects

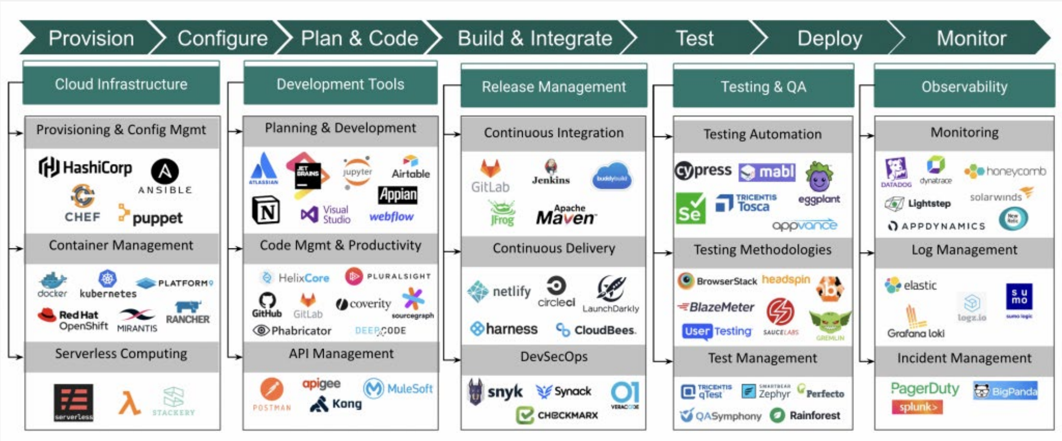
1. Features created
2. Technologies used
3. Toolchain, Dev process & assets used

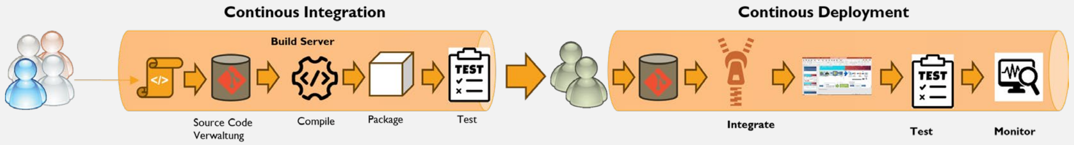




- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.

CI/CD IN MOBILE APP DEVELOPMENT - INTRODUCTION OF CI/CD CONCEPTS



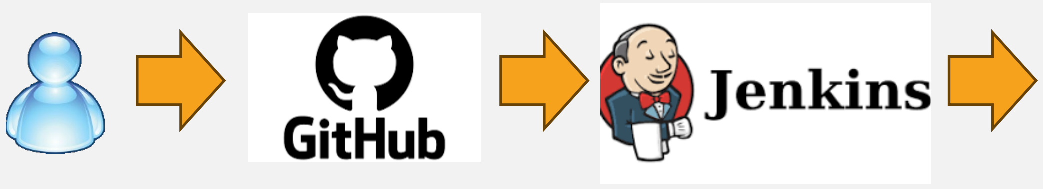


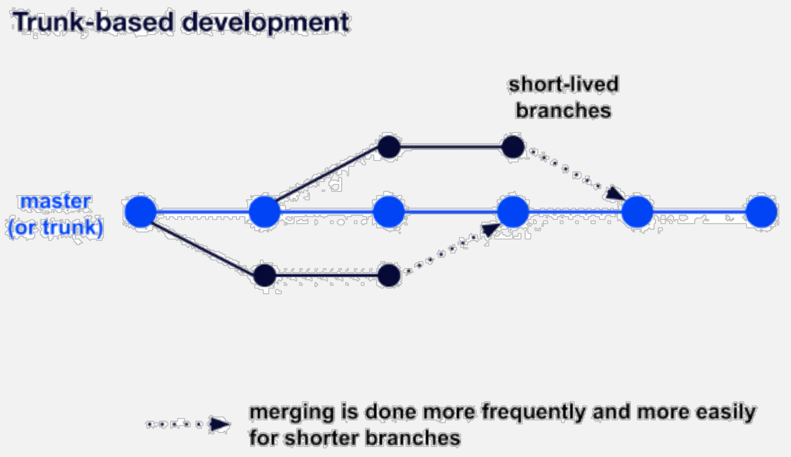
- Define a CI/CD process as draft
  1. Used tools for CI
  2. Used tools for CD
  3. Current state of the art tool analysis
  4. Analyse the added value of each tool
     - Pro: Identification of use cases i.e. for test & build & deploy
     - Con: Tool versions changed and their features

- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs

CI/CD IN MOBILE APP DEVELOPMENT - SELECTION OF SUITABLE TOOLS AND PLATFORMS

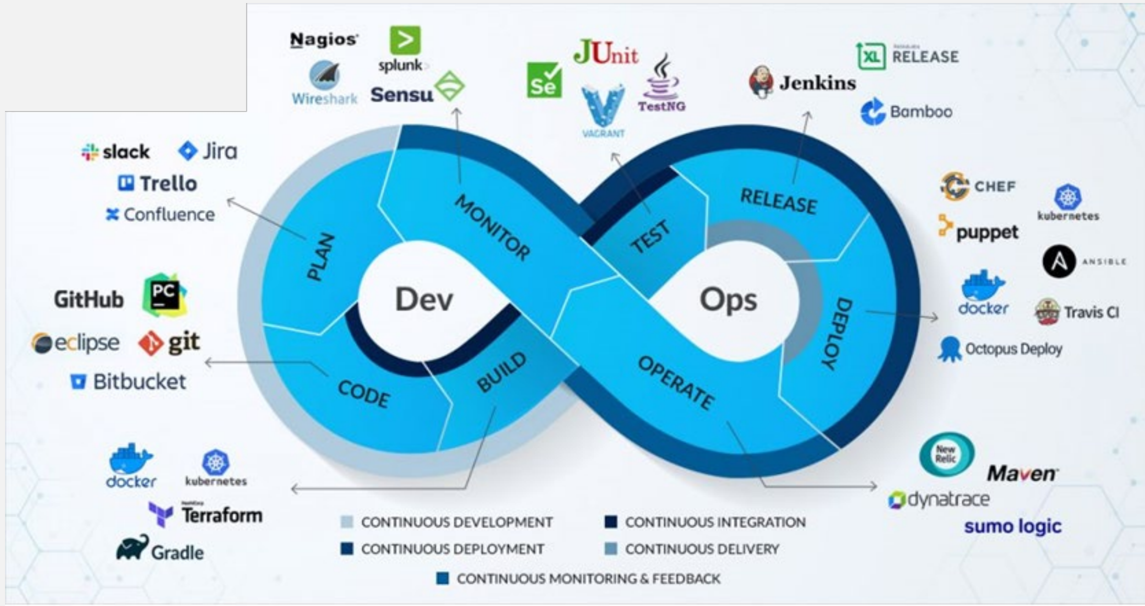
In 2016/17 CI/CD with Jenkins





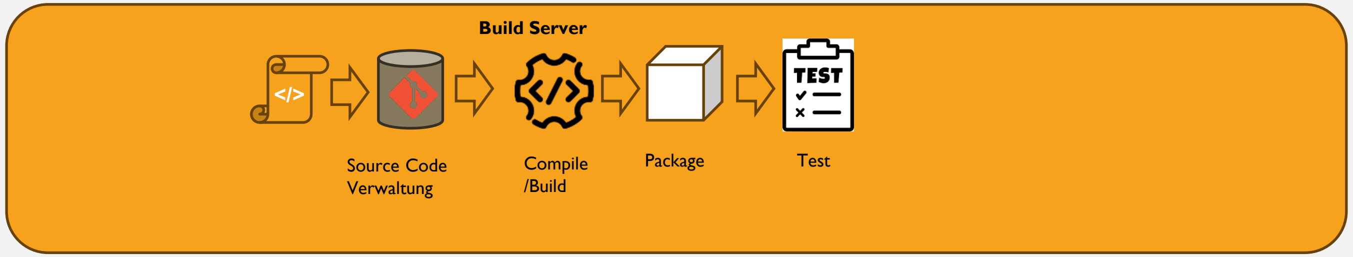
Trunk-based development
short-lived branches
master (or trunk)
merging is done more frequently and more easily for shorter branches

In 2019 change to Devops



- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs
- **Establishment of an Automated Build Pipeline:** Describe the setup of an automated build pipeline that automates the build process for mobile apps, including compilation, testing, and deployment.

CI/CD IN MOBILE APP DEVELOPMENT - ESTABLISHMENT OF AN AUTOMATED BUILD PIPELINE



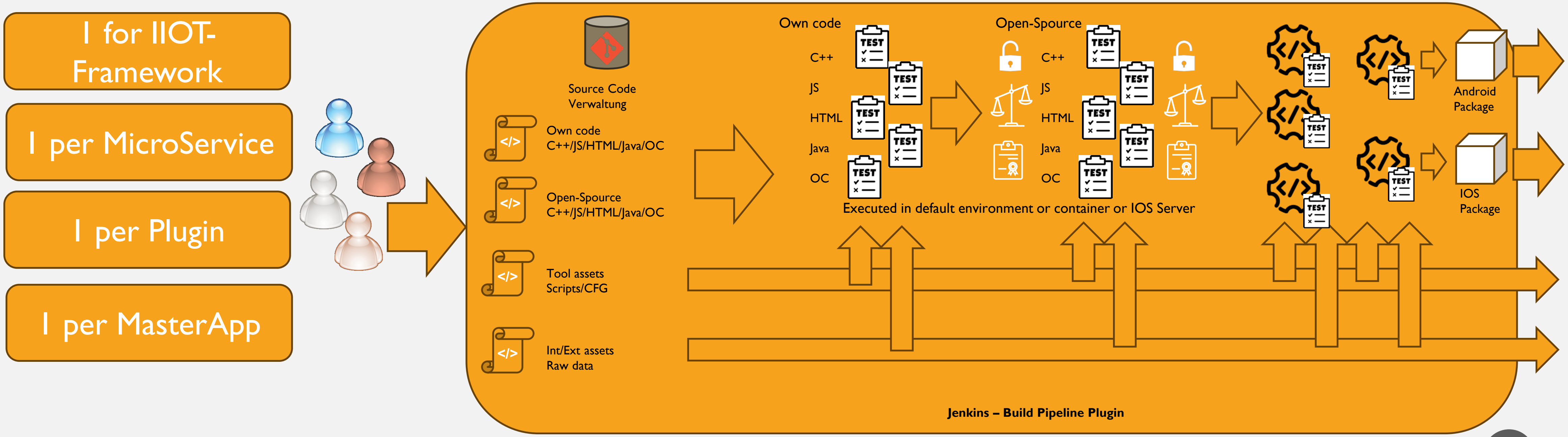
Build Server

Source Code Verwaltung

Compile /Build

Package

Test



1 for IIOT-Framework

1 per MicroService

1 per Plugin

1 per MasterApp

Source Code Verwaltung

Own code C++/JS/HTML/Java/OC

Open-Spource C++/JS/HTML/Java/OC

Tool assets Scripts/CFG

Int/Ext assets Raw data

Own code

C++

JS

HTML

Java

OC

Open-Spource

C++

JS

HTML

Java

OC

Executed in default environment or container or IOS Server

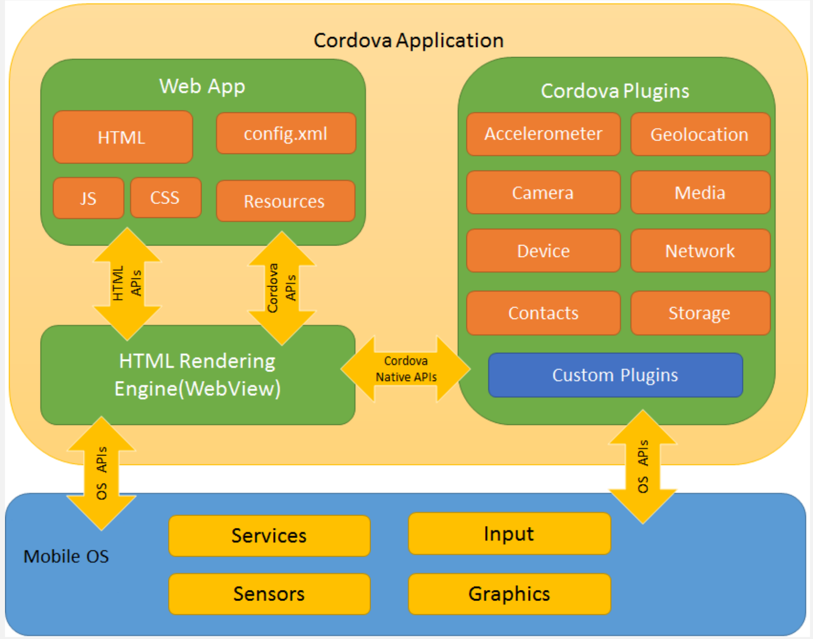
Android Package

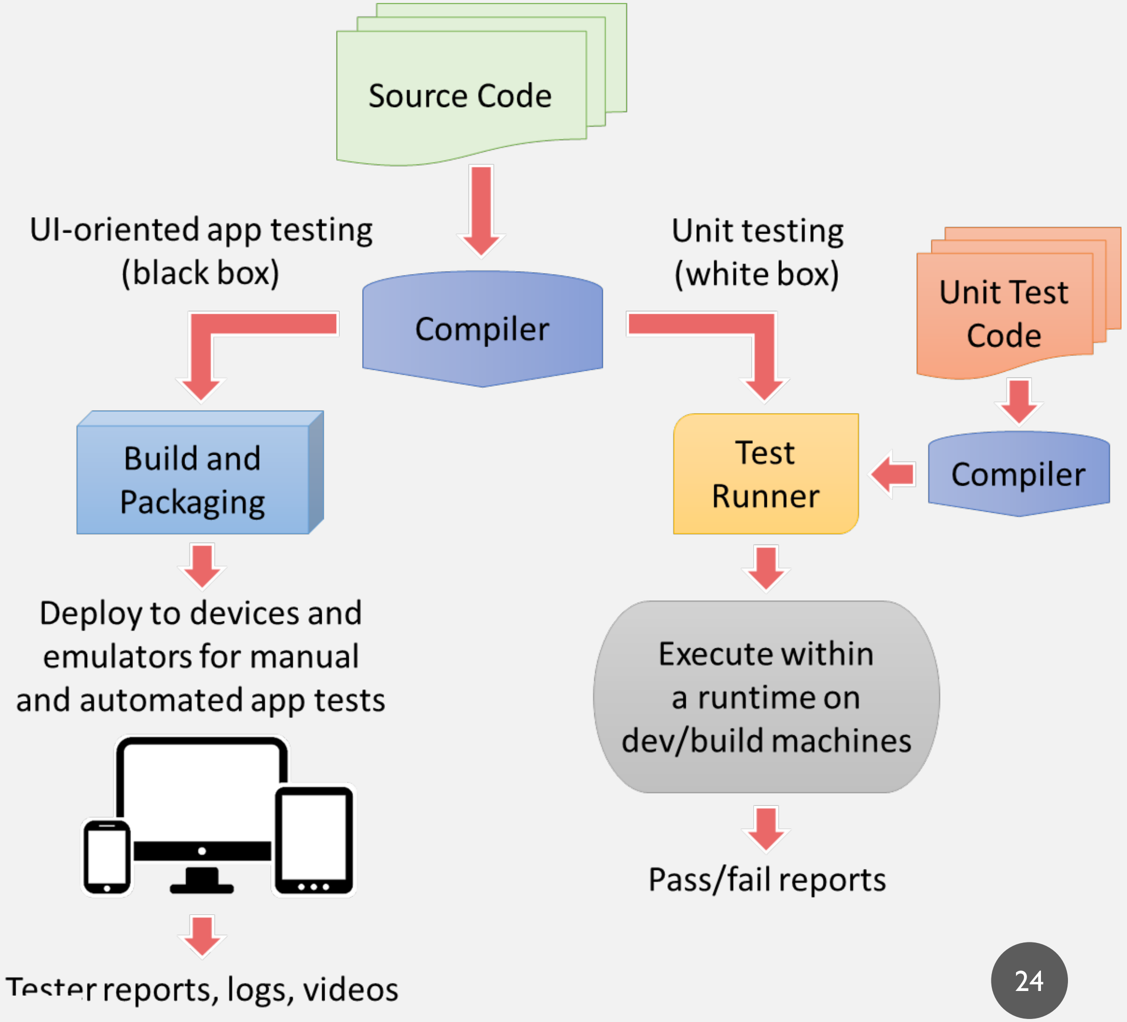
IOS Package

Jenkins – Build Pipeline Plugin

CI/CD IN MOBILE APP DEVELOPMENT - INTEGRATION OF TESTS

- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs
- **Establishment of an Automated Build Pipeline:** Describe the setup of an automated build pipeline that automates the build process for mobile apps, including compilation, testing, and deployment.
- **Integration of Tests:** Explain the integration of automated tests into the CI/CD process to ensure that every code change is thoroughly tested and the quality of the application is maintained.

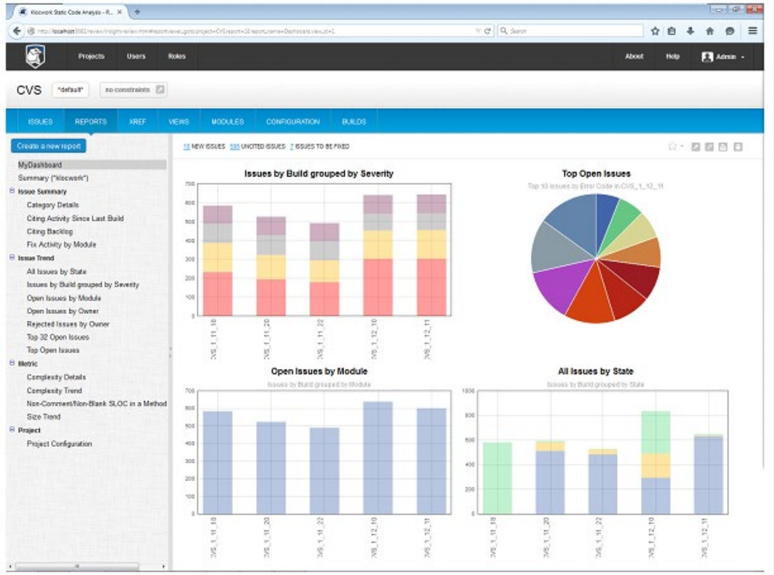


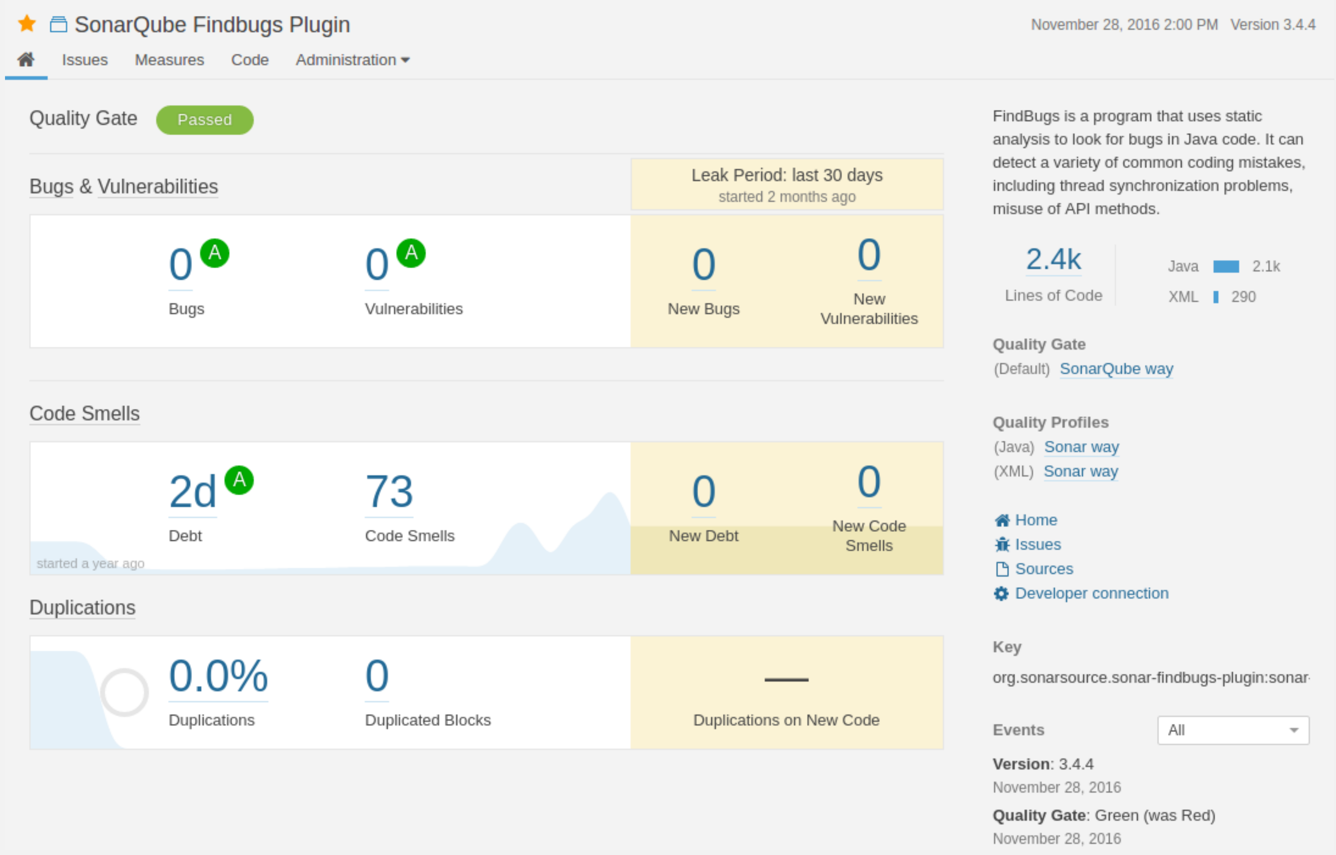


Source Code
UI-oriented app testing (black box)
Compiler
Unit testing (white box)
Unit Test Code
Build and Packaging
Test Runner
Compiler
Deploy to devices and emulators for manual and automated app tests
Execute within a runtime on dev/build machines
Pass/fail reports
Tester reports, logs, videos

CI/CD IN MOBILE APP DEVELOPMENT - IMPLEMENTATION OF CODE QUALITY METRICS

- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs
- **Establishment of an Automated Build Pipeline:** Describe the setup of an automated build pipeline that automates the build process for mobile apps, including compilation, testing, and deployment.
- **Integration of Tests:** Explain the integration of automated tests into the CI/CD process to ensure that every code change is thoroughly tested and the quality of the application is maintained.
- **Implementation of Code Quality Metrics:** Discuss the implementation of code quality metrics and the integration of static code analysis tools to ensure that the code meets the defined quality standards.





★ SonarQube Findbugs Plugin

November 28, 2016 2:00 PM Version 3.4.4

Issues | Measures | Code | Administration ▾

Quality Gate: Passed

Bugs & Vulnerabilities

Leak Period: last 30 days
started 2 months ago

Bugs	Vulnerabilities	New Bugs	New Vulnerabilities
0 A	0 A	0	0

Code Smells

Debt	Code Smells	New Debt	New Code Smells
2d A	73	0	0

started a year ago

Duplications

Duplications	Duplicated Blocks	Duplications on New Code
0.0%	0	—

FindBugs is a program that uses static analysis to look for bugs in Java code. It can detect a variety of common coding mistakes, including thread synchronization problems, misuse of API methods.

2.4k Lines of Code

Java	2.1k
XML	290

Quality Gate
(Default) SonarQube way

Quality Profiles
(Java) Sonar way
(XML) Sonar way

Home
Issues
Sources
Developer connection

Key
org.sonarsource.sonar-findbugs-plugin:sonar

Events: All

Version: 3.4.4
November 28, 2016

Quality Gate: Green (was Red)
November 28, 2016

CI/CD IN MOBILE APP DEVELOPMENT - VERSIONING AND RELEASE AND DEPLOYMENT MANAGEMENT

- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs
- **Establishment of an Automated Build Pipeline:** Describe the setup of an automated build pipeline that automates the build process for mobile apps, including compilation, testing, and deployment.
- **Integration of Tests:** Explain the integration of automated tests into the CI/CD process to ensure that every code change is thoroughly tested and the quality of the application is maintained.
- **Implementation of Code Quality Metrics:** Discuss the implementation of code quality metrics and the integration of static code analysis tools to ensure that the code meets the defined quality standards.
- **Versioning and Release Management:** Explain the importance of version control and release management in a CI/CD workflow for mobile apps and how these processes can be effectively implemented.

- Using Github and Jenkins for Versioning and Release
  - Configure Build Pipeline with Github ReleaseNotes
  - Configure Google/Apple Certification for uploading
  - NOT USED:Automated upload in google/apple store. Coporate team request review stage gate
  - Manual upload files by coporate team with deployment information by deploynebt pipline

- Using app stores for deployment managment
  - Prepare releases
  - Manual „Go“ Button

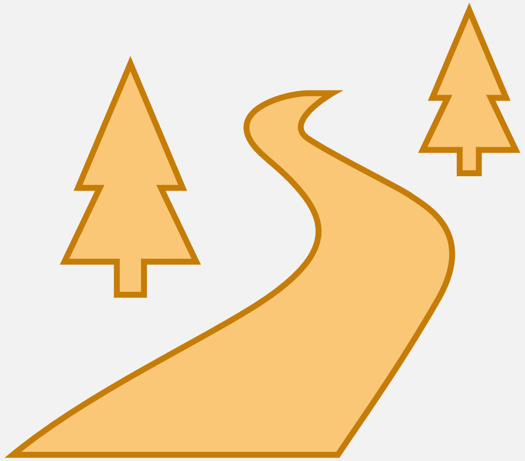
Available on the App Store

GET IT ON Google Play

- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs
- **Establishment of an Automated Build Pipeline:** Describe the setup of an automated build pipeline that automates the build process for mobile apps, including compilation, testing, and deployment.
- **Integration of Tests:** Explain the integration of automated tests into the CI/CD process to ensure that every code change is thoroughly tested and the quality of the application is maintained.
- **Implementation of Code Quality Metrics:** Discuss the implementation of code quality metrics and the integration of static code analysis tools to ensure that the code meets the defined quality standards.
- **Versioning and Release Management:** Explain the importance of version control and release management in a CI/CD workflow for mobile apps and how these processes can be effectively implemented.
- **Training and Support for Development Teams:** Emphasize the importance of training and support for development teams in the adoption of new CI/CD practices and tools to ensure a smooth transition.

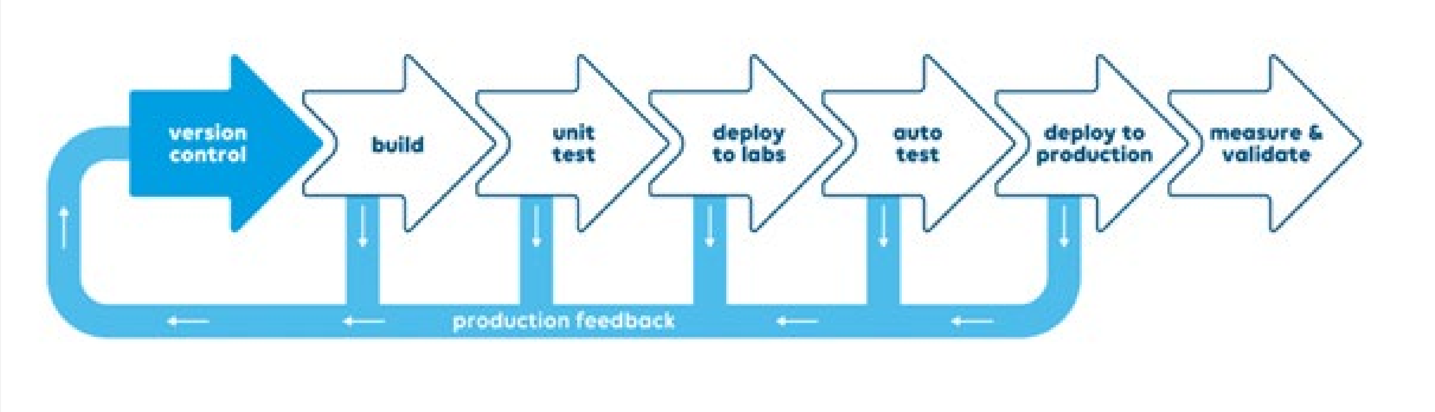
CI/CD IN MOBILE APP DEVELOPMENT - TRAINING AND SUPPORT FOR DEVELOPMENT TEAMS

- Added courses for elemental tools to training path
- Added courses for technology i.e. cordovae , JS, etc
- Added courses for IIOT Framework
- Live learning course for CI/CD



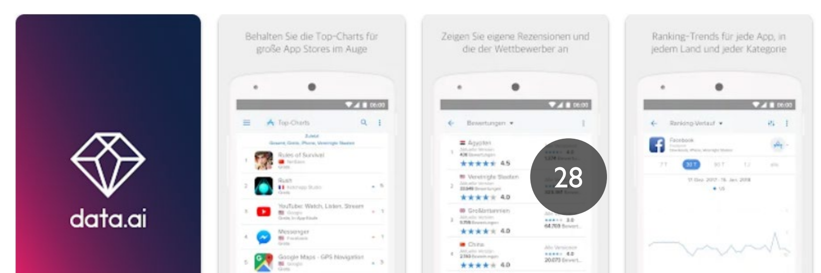
- **Analysis of Current Development Practices:** Begin by conducting an analysis of the current processes and challenges in app development within your company or organization.
- **Introduction of CI/CD Concepts:** Explain the fundamentals of Continuous Integration (CI) and Continuous Deployment (CD), as well as their benefits for software development, especially for applications.
- **Selection of Suitable Tools and Platforms:** Introduce various tools and platforms that are suitable for implementing CI/CD in mobile app development. Consider aspects such as support for mobile platforms, integration with existing development environments, and costs
- **Establishment of an Automated Build Pipeline:** Describe the setup of an automated build pipeline that automates the build process for mobile apps, including compilation, testing, and deployment.
- **Integration of Tests:** Explain the integration of automated tests into the CI/CD process to ensure that every code change is thoroughly tested and the quality of the application is maintained.
- **Implementation of Code Quality Metrics:** Discuss the implementation of code quality metrics and the integration of static code analysis tools to ensure that the code meets the defined quality standards.
- **Versioning and Release Management:** Explain the importance of version control and release management in a CI/CD workflow for mobile apps and how these processes can be effectively implemented.
- **Training and Support for Development Teams:** Emphasize the importance of training and support for development teams in the adoption of new CI/CD practices and tools to ensure a smooth transition.
- **Monitoring and Feedback Loops:** Discuss the implementation of monitoring and feedback mechanisms to continuously monitor the status of the CI/CD pipeline and identify potential areas for improvement.

CI/CD IN MOBILE APP DEVELOPMENT - MONITORING AND FEEDBACK LOOPS

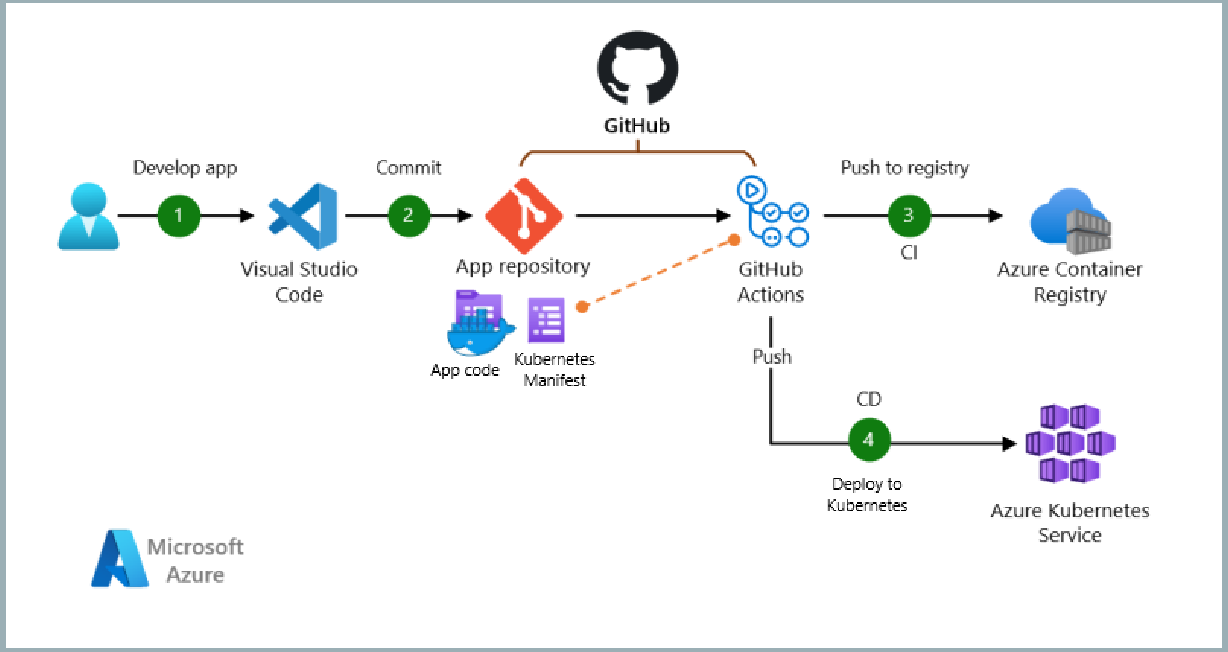


- Using Jenkins to
  - Get feedback from tool in CI and CD i.e. Junit, SonarCobe

- Using monitor features for deployment
  - Google Analytics
  - App Annie (Business)


data.ai
Behalten Sie die Top-Charts für große App Stores im Auge
Zeigen Sie eigene Rezensionen und die der Wettbewerber an
Ranking-Trends für jede App, in jedem Land und jeder Kategorie

PART IV - BEST PRACTICES CI/CD IN THE CLOUD

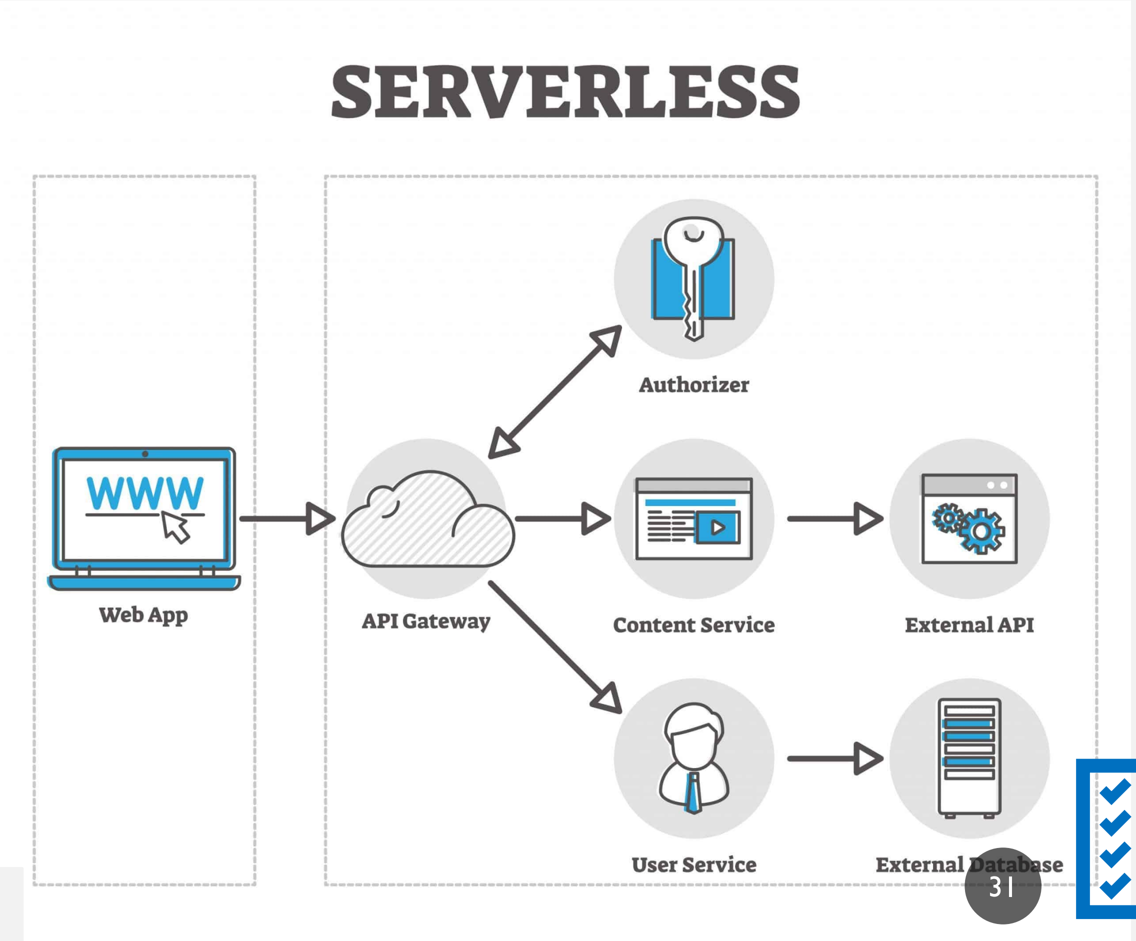


RELEVANCE FOR THE EFFICIENCY AND QUALITY OF SOFTWARE DEVELOPMENT PROCESSES

- Modularity
  - Division into independent modules.
  - More efficient work for developers.
  - Faster development, testing, and deployment.
  - Reduced impact of changes.
  - Facilitated maintainability and scalability.
- Cohesion
  - Clear and specific tasks in modules.
  - Reduction of unwanted dependencies and complexity.
  - Increase in efficiency.
  - Better structured code and simpler testing procedures.
  - Enhancement of software quality and maintainability.
- Scalability
  - Response to increasing demands.
  - Rapid adaptation to spikes in load.
  - Maintenance of performance and stability.
  - Reliable functioning under increasing stress.
  - Sustaining high quality.
- Automation
  - Reduction of manual effort and human errors.
  - Enhancement of efficiency and consistent results.
  - Release of time for developing new features and quality improvements.
  - Acceleration of development processes.
  - Faster time to market for software products.

CI/CD ARCHITECTURE PATTERNS IN THE CLOUD

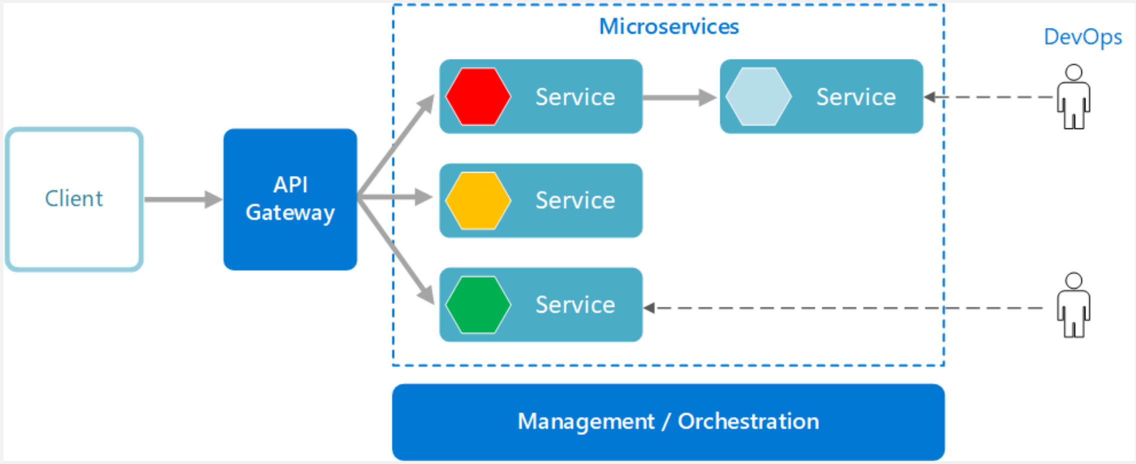
- Serverless Architecture:
  - Deployment of functions or microservices in independent units without having to worry about the underlying infrastructure.
  - Granular scalability and precise resource utilization.
  - Suitable for event-driven or scalable applications, offering high flexibility.
- Advantages for CI/CD:
  - Rapid deployment (of services).
  - Easily integrable into CI/CD pipelines as they consist of small, independent functions.
  - Automated build and deployment scripts can be used to deploy functions.
  - CI/CD pipelines can conduct automatic tests for each function to ensure quality.
  - Cost optimization (pay-as-you-go pricing model).
  - Scalability (granular scaling at the function level).



Serverless-Architectur (Source Kofi-group, 2024)

CI/CD ARCHITECTURE PATTERNS IN THE CLOUD

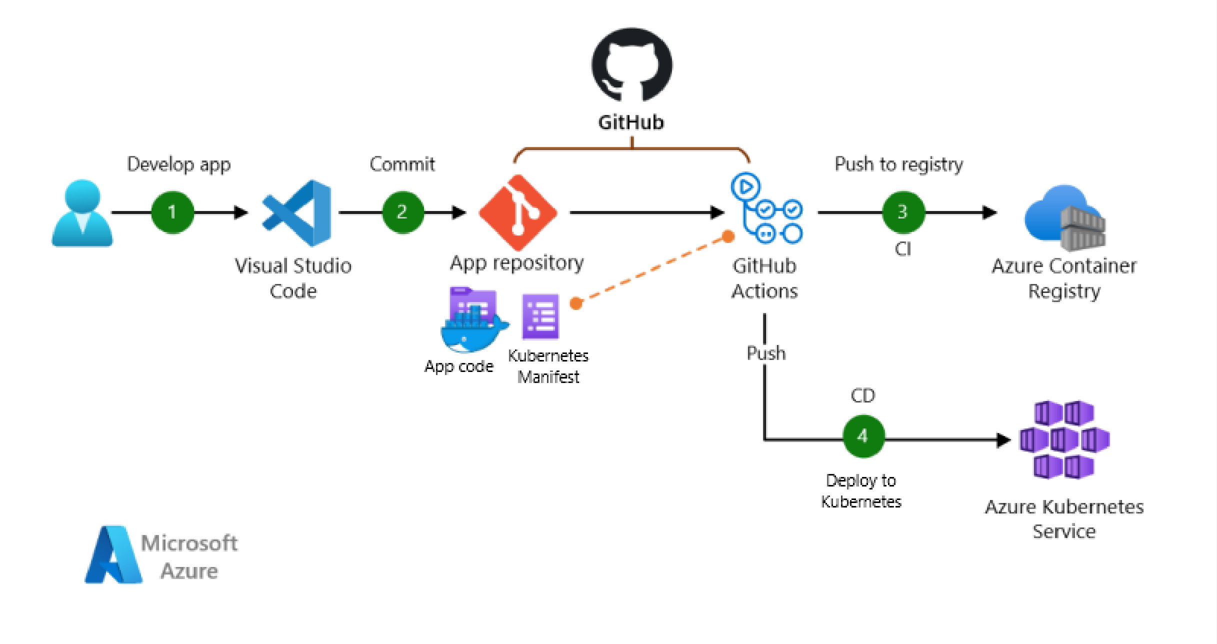
- Microservice Architecture:
  - Microservice architecture involves breaking down an application into smaller, independent services or microservices, each fulfilling a specific function.
  - Independent development and deployment: Each microservice can be developed, tested, deployed, and scaled independently, enhancing the maintainability, scalability, and flexibility of the application.
  - Standardized communication: Microservices communicate via standardized interfaces such as APIs and can be implemented in various programming languages and technologies.
- Advantages for CI/CD:
  - Independent deployment: By dividing into microservices, updates or new features can be deployed independently without affecting other parts of the system.
  - Scalability: Microservices can be scaled separately to respond to changing requirements or spikes in load, enabling efficient resource utilization.
  - Technological diversity: Each microservice can be implemented in a different technology or programming language, fostering flexibility and innovation.



MicroService Architectur (Source: Microsoft, 2024)

CI/CD ARCHITECTURE PATTERNS IN THE CLOUD

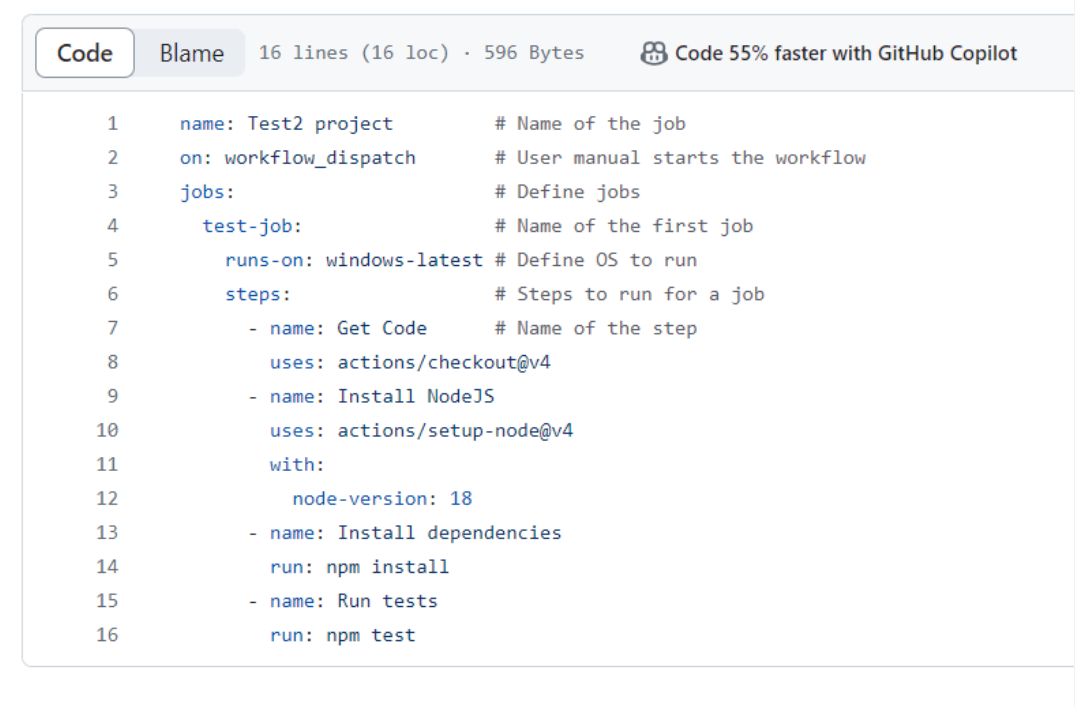
- Container orchestration:
  - Container orchestration systems like Kubernetes enable the management and orchestration of containers in distributed environments.
  - Containers provide consistent deployment of applications across different environments and reduce dependency on the underlying infrastructure.
  - Kubernetes allows for the automation of container deployment, scaling, load balancing, and recovery, thereby improving operational efficiency and reliability.
- Benefits **CI/CD:**
  - Container orchestration systems like Kubernetes facilitate the integration of containers into CI/CD pipelines.
  - Developers can use automated scripts to build, test, and deploy container images to Kubernetes clusters.
  - CI/CD pipelines can also perform automatic rollbacks if tests fail or issues arise.
  - Consistent deployment, Automated scaling, High availability



Push-based Architecture using GitHub Actions for CI and CD.

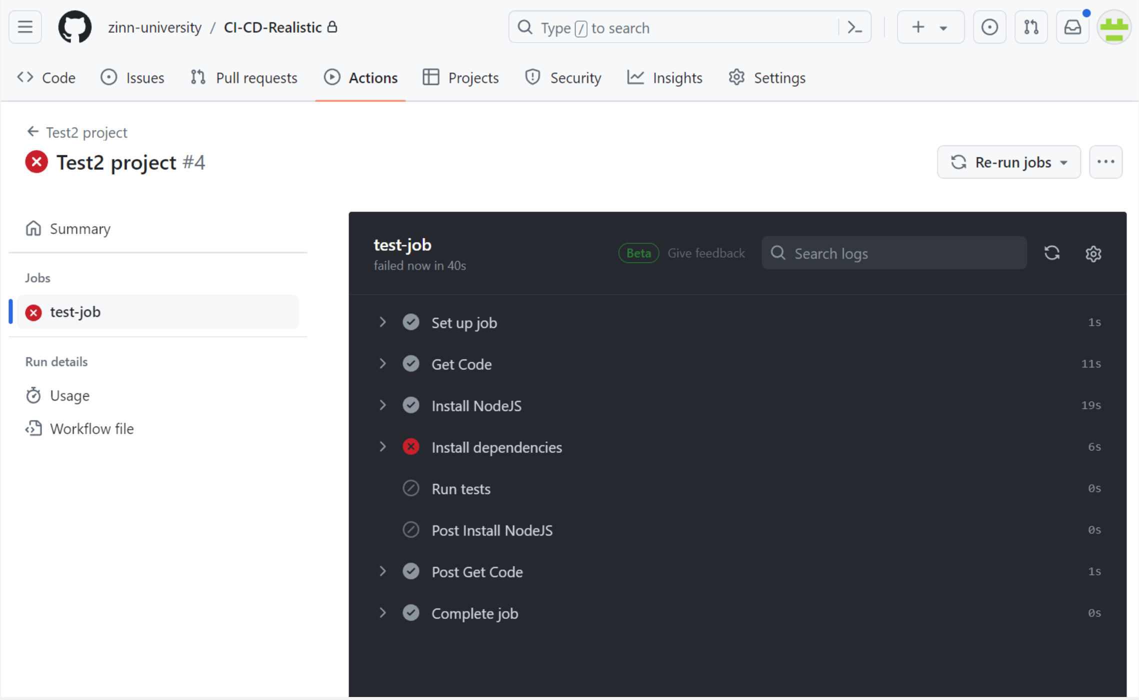
(Source: Microsoft, 2024)

SHOWCASE „GITHUB ACTIONS“



Code
Blame
16 lines (16 loc) · 596 Bytes
Code 55% faster with GitHub Copilot

```yaml
name: Test2 project          # Name of the job
on: workflow_dispatch        # User manual starts the workflow
jobs:                        # Define jobs
  test-job:                  # Name of the first job
    runs-on: windows-latest # Define OS to run
    steps:                   # Steps to run for a job
      - name: Get Code       # Name of the step
        uses: actions/checkout@v4
      - name: Install NodeJS
        uses: actions/setup-node@v4
        with:
          node-version: 18
      - name: Install dependencies
        run: npm install
      - name: Run tests
        run: npm test
```



Test2 project #4

test-job

test-job
failed now in 40s

```
Set up job              1s
Get Code                11s
Install NodeJS          19s
Install dependencies    6s
Run tests               0s
Post Install NodeJS     0s
Post Get Code           1s
Complete job            0s
```

PART V – SUMMARY AND FUTURE CI/CD TOPICS

CURRENT REASEARCH TOPICS IN CI/CD

Development of Cloud Technologies

Cloud platforms will continue to evolve to offer more features and services that enhance the automation, scalability, and security of CI/CD processes.

Integration of AI and ML

Artificial Intelligence (AI) and Machine Learning (ML) will increasingly be integrated into CI/CD pipelines to enable automatic error detection, deployment optimization, and prediction of performance issues.

Micro-Frontends

Similar to microservices, micro-frontends will emerge to divide frontend applications into small, independent components, leading to more flexible deployment and update processes.

Infrastructure as Code (IaC)

IaC is already being used in many CI/CD pipelines, but this trend is expected to further strengthen. The use of tools like Terraform or Ansible for automating infrastructure configuration enables consistent deployment of infrastructure resources across different environments.

USE OF CI/CD IN PROFESSIONAL SW PROJECTS: A CONCRETE EXAMPLE FROM OWN EXPERIENCE

trial lecture
presentation by
Marcus Zinn

Brief CV

Business Roles

- 5,5 years mobile leader (IIOT Mobile Apps)
- 5 years leader for patents (focusing AI and Software patents)
- 12 years software engineering
- …

Hobbies

- Lecturer since 2004 (Mobile Apps, Informatics, Software Engineering, Data Warehouse, ..)
- Support / Supervision for bachelor / master thesis (28)
- IIOT Beekeeping
- AI based image generation (Stable diffusion / comfyUI)
- …

Thank you!