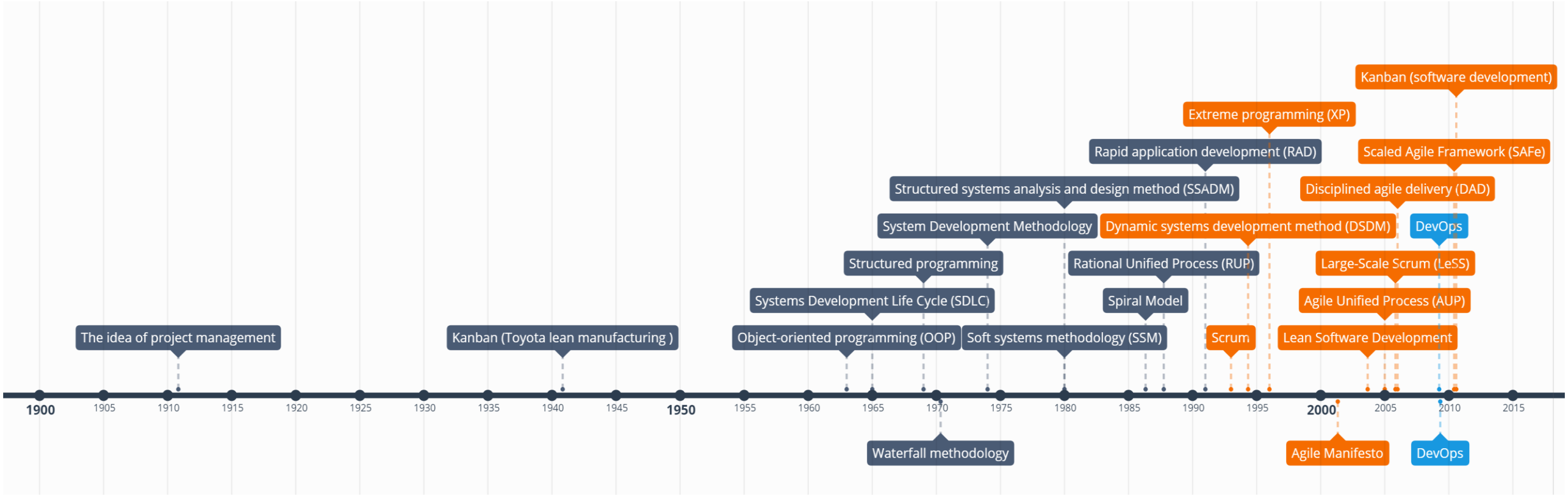
Sebastian Copei (sebastian.copei@iee.fraunhofer.de)

DevOps Technologies – Vorlesung 2 – WS 24/25

Lernziele für Heute

1. Historie
2. Desktop vs. Cloud; Wie verteile ich meine Software?
3. Vorteile / Nachteile Cloud
4. Was macht DevOps aus?
5. Was bedeutet das für eine Softwarearchitektur?
6. Docker
7. Kubernetes

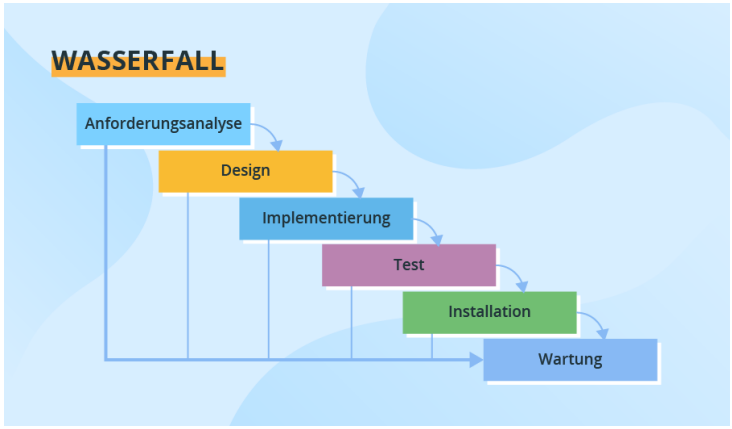
Historie

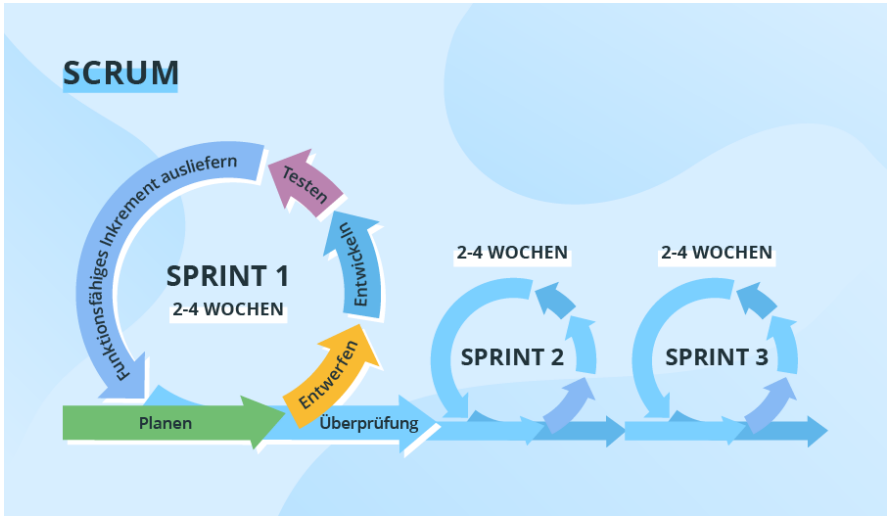


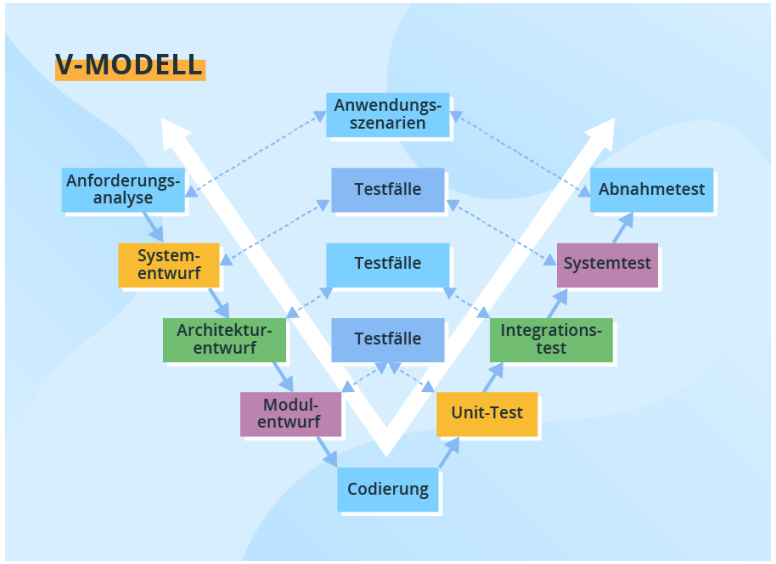
[1] https://almbok.com/kb/software_development_process_history

Fraunhofer IEE

Historie



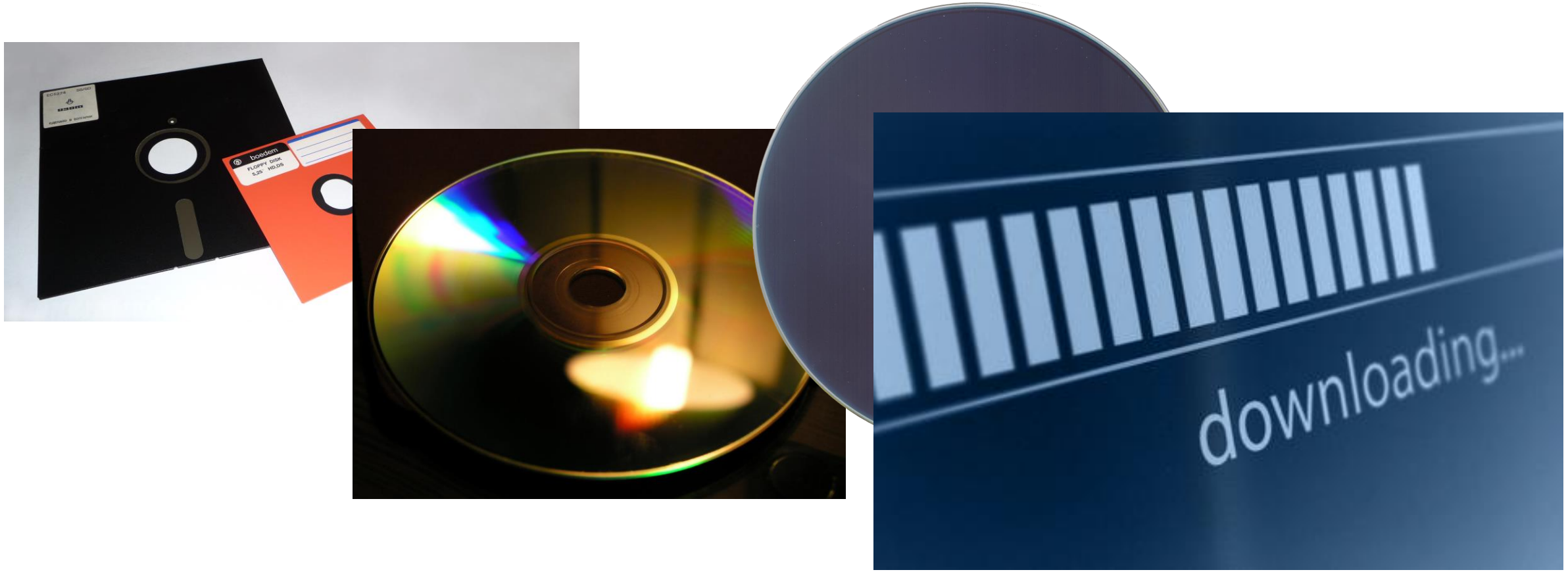




[2] https://www.scnsoft.de/blog/vorgehensmodelle-der-softwareentwicklung

Fraunhofer IEE

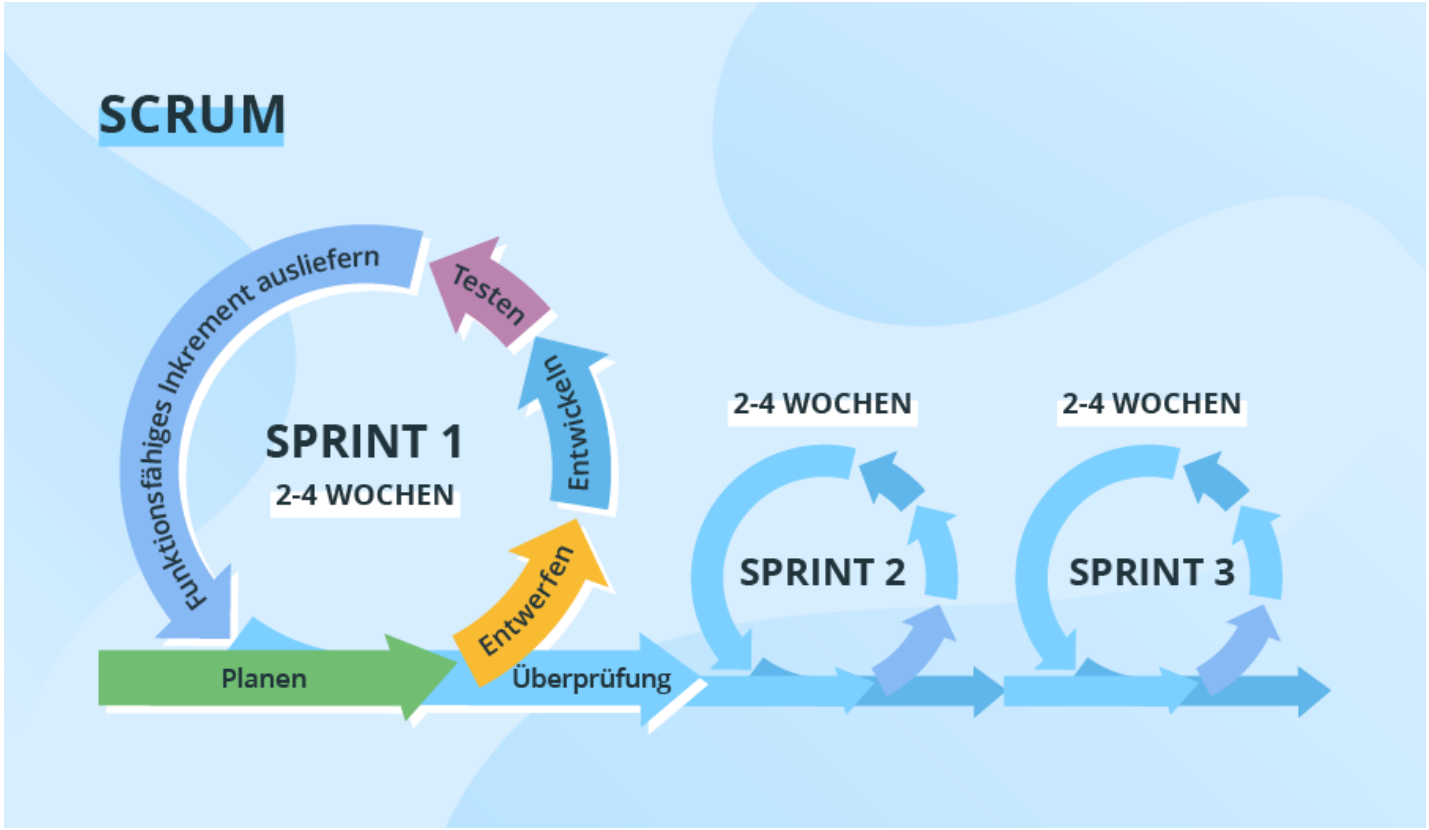
Desktop vs. Cloud; Wie verteile ich meine Software?



Fraunhofer IEE

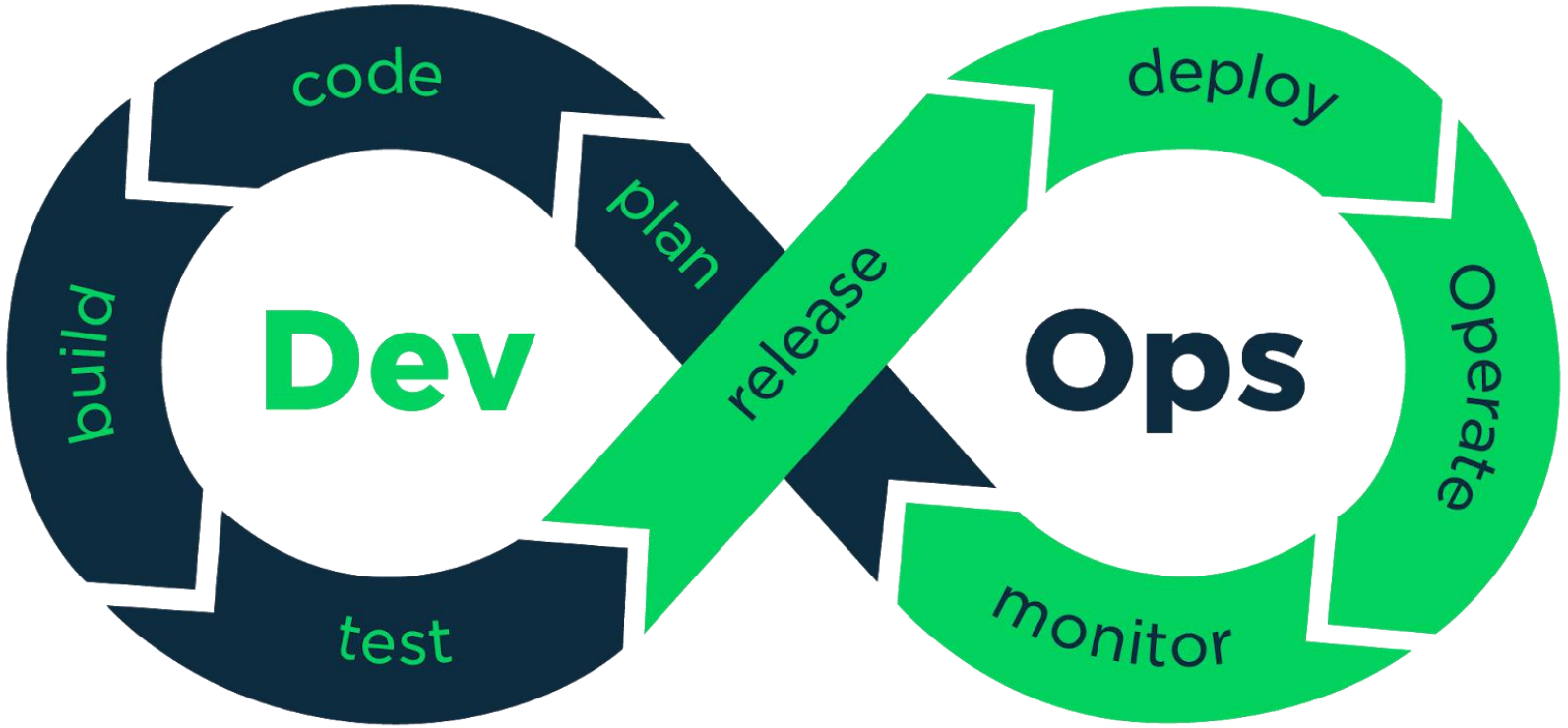
Vorteile / Nachteile Cloud

Was macht DevOps aus?



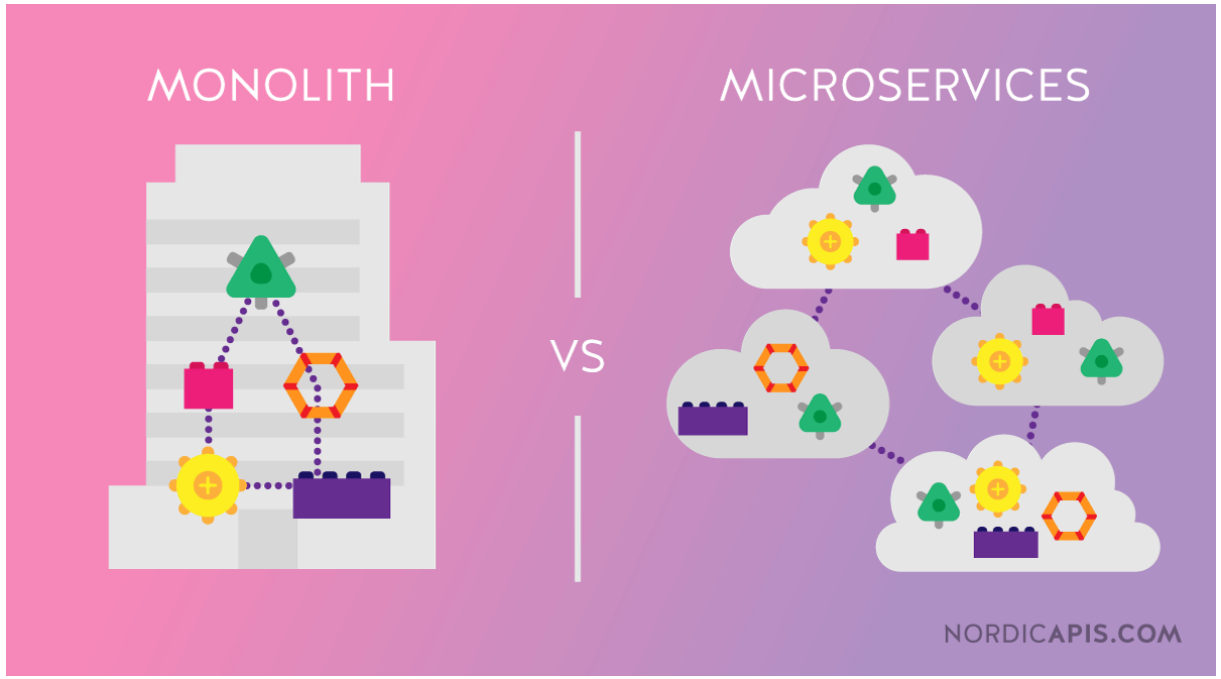
Fraunhofer IEE

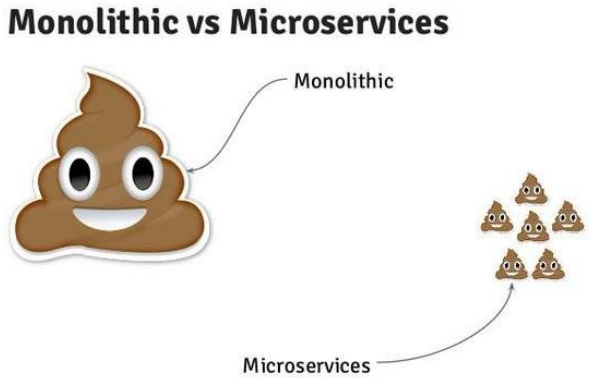
Was macht DevOps aus?



Fraunhofer IEE

Was bedeutet das für eine Softwarearchitektur?

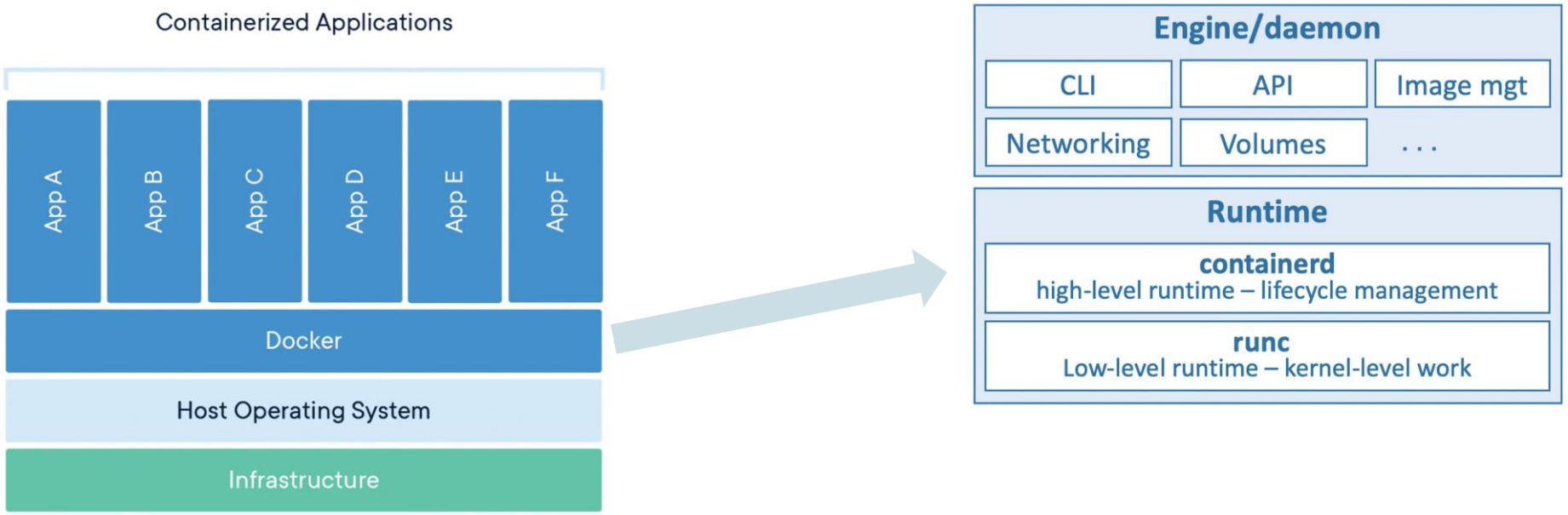




[3] https://nordicapis.com/should-you-start-with-a-monolith-or-microservices/
[4] https://www.linkedin.com/pulse/microservices-principles-converting-monolith-arati-joshi/

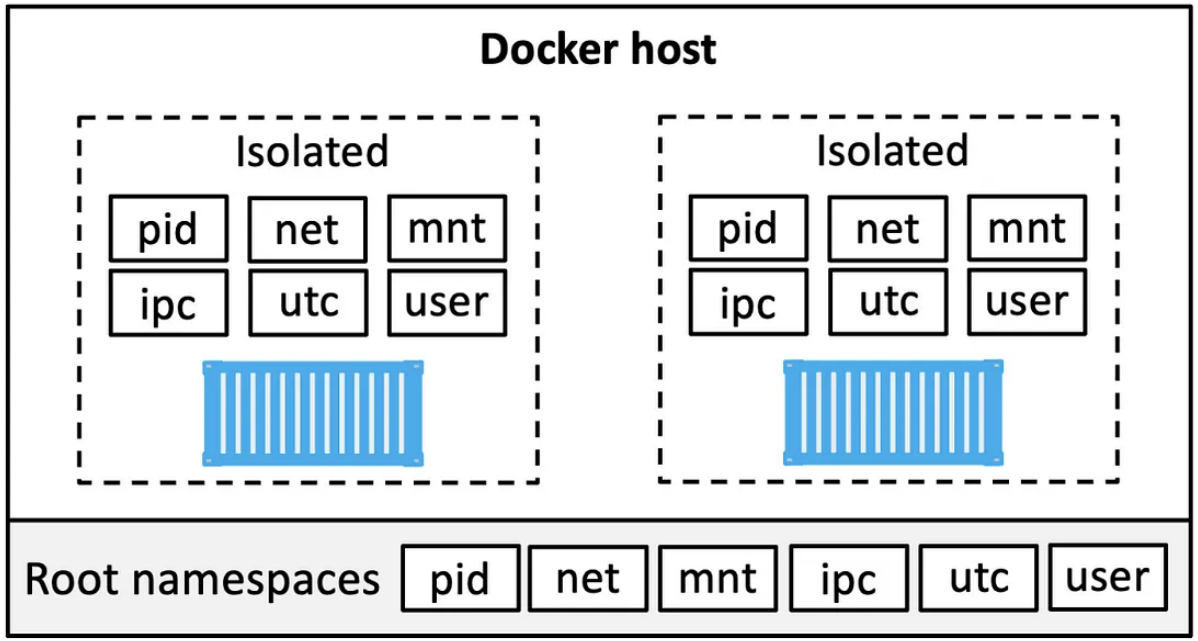
Fraunhofer IEE

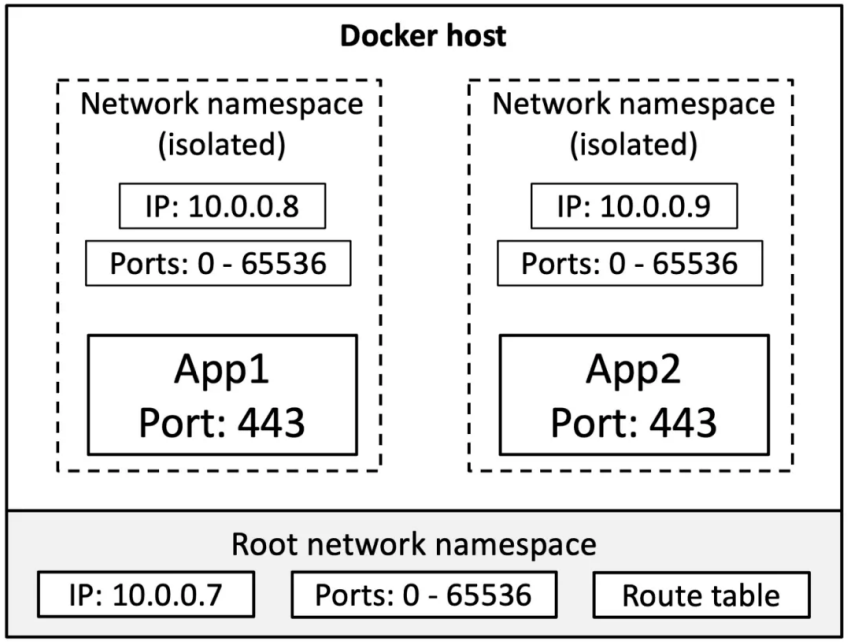
Docker



[5] https://medium.com/@dmosyan/deep-dive-into-docker-containers-architecture-and-features-530a937f4c87

Docker





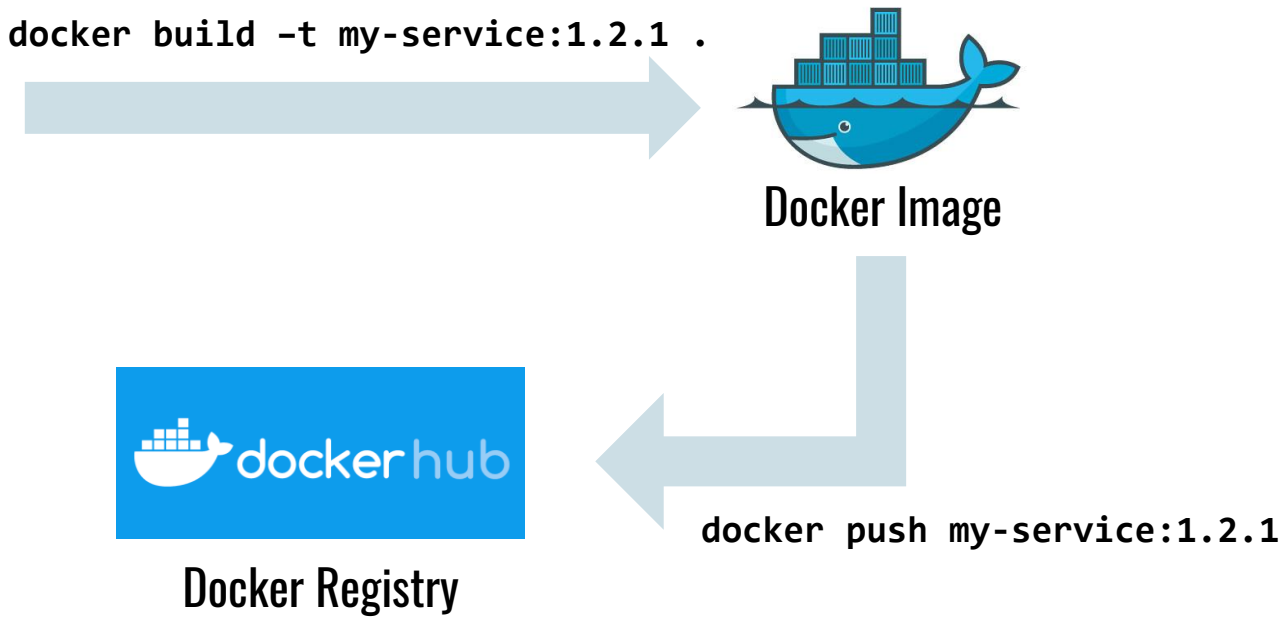
[6] https://medium.com/@dmosyan/deep-dive-into-docker-containers-architecture-and-features-530a937f4c87

Fraunhofer IEE

Docker

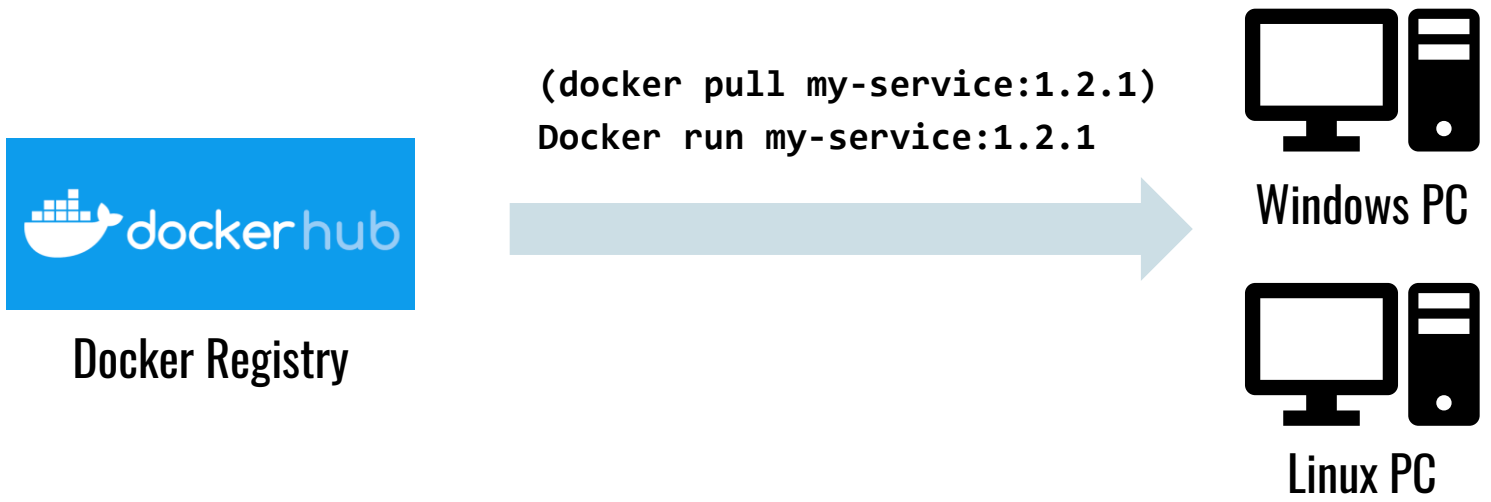
```
FROM node:20.15.0-alpine

WORKDIR /usr/src/app

COPY . .

RUN npm ci
RUN npm run build

EXPOSE 3000
CMD [“npm“, “run“, “start:prod“]
```

Dockerfile



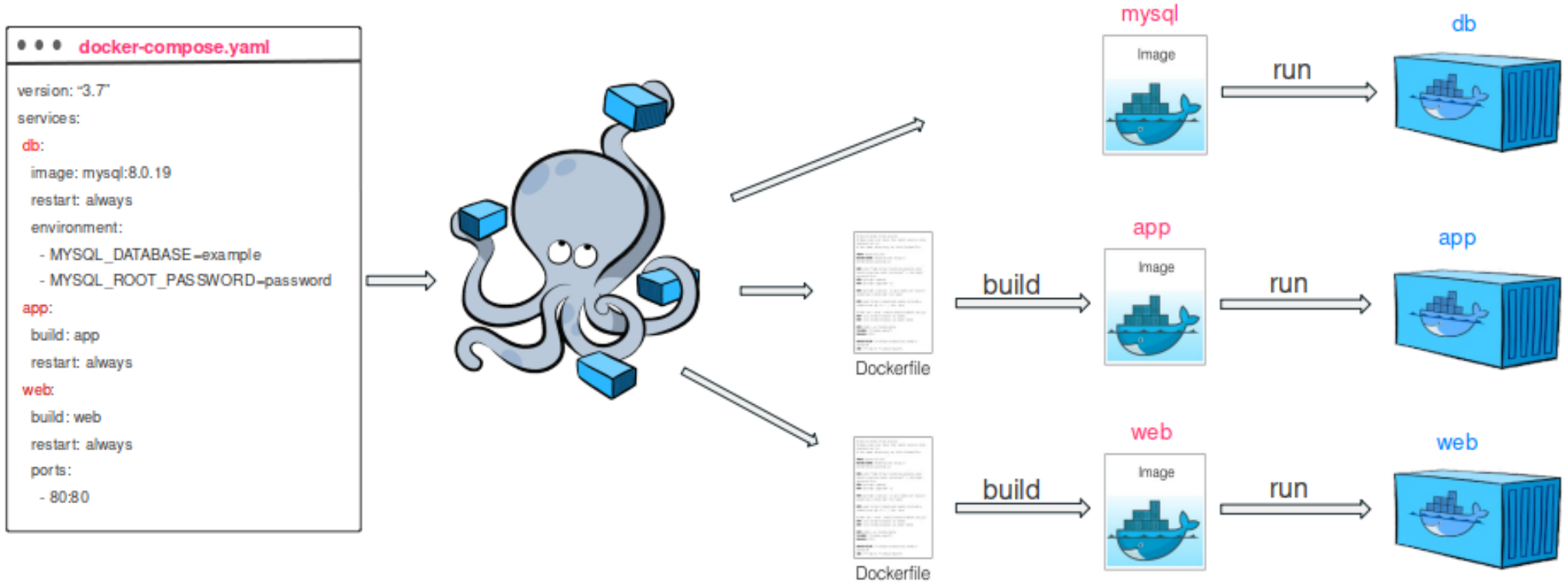
[7] https://www.ionos.de/digitalguide/server/knowhow/dockerfile/
[8] https://www.linode.com/docs/guides/how-to-use-dockerfiles/

Docker



[7] https://www.ionos.de/digitalguide/server/knowhow/dockerfile/
[8] https://www.linode.com/docs/guides/how-to-use-dockerfiles/

Docker



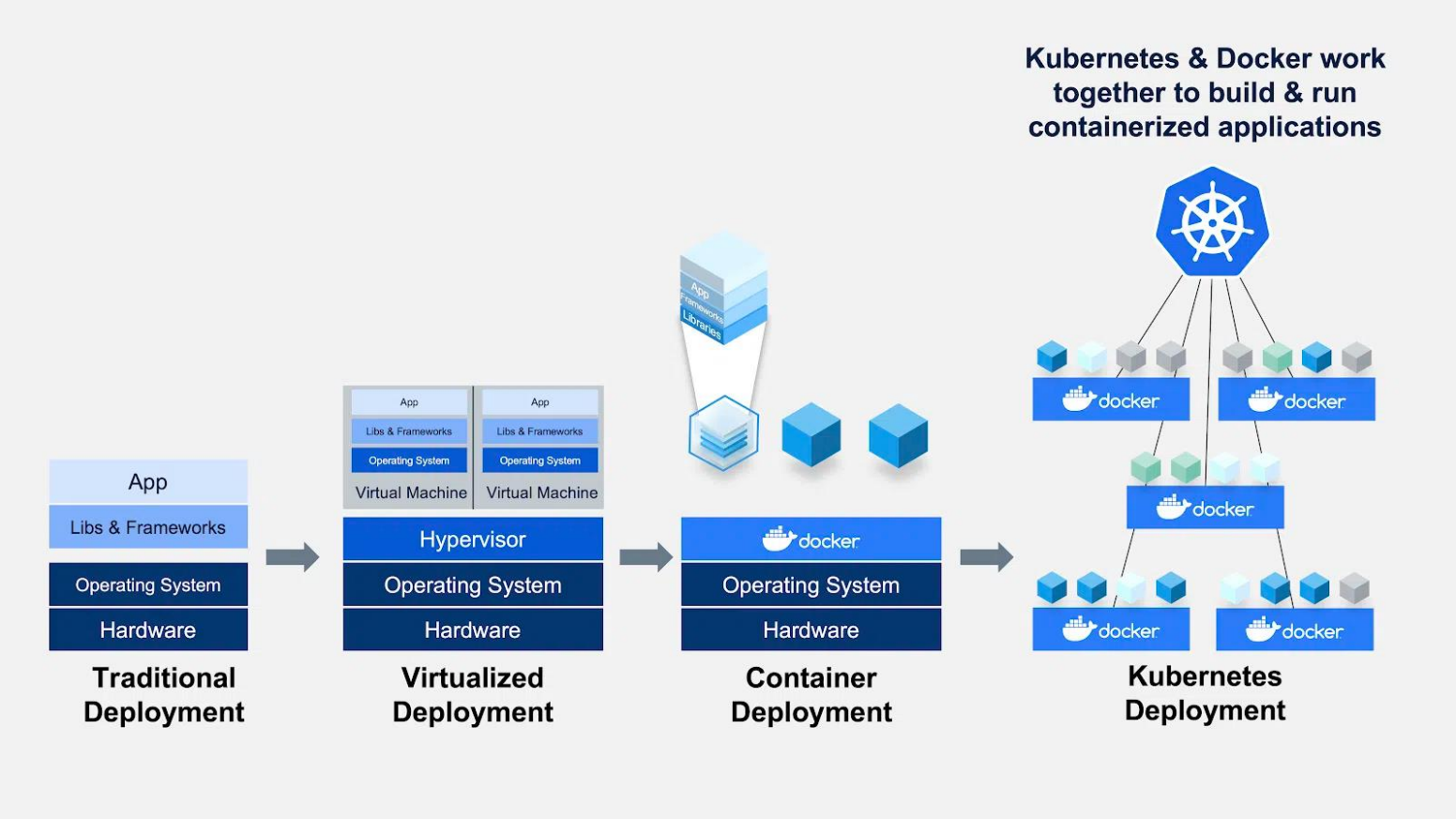
[9] https://code.visualstudio.com/docs/containers/docker-compose

Fraunhofer IEE

Docker - Zusammenfassung

- **kernal namespaces** fassen rudimentäre OS-Funktionen zusammen
  - PID
  - NET
  - MNT
  - IPC
  - UTS
  - User
- Jeder **Container** bekommt eigene **namespace**
- **cgroups** werden verwendet, um die Ressourcennutzung eines **Containers** zu steuern
- Ein **Docker Image** wird einmal aus von einem **Dockerfile** erstellt kann über eine **Registry** geteilt werden
- Eine gestartete Instanz eines **Docker Images** wird **Docker Container** genannt
- Viele **Docker Container** Definitionen können in einer **Docker Compose** zu einem **Stack** zusammengefasst werden

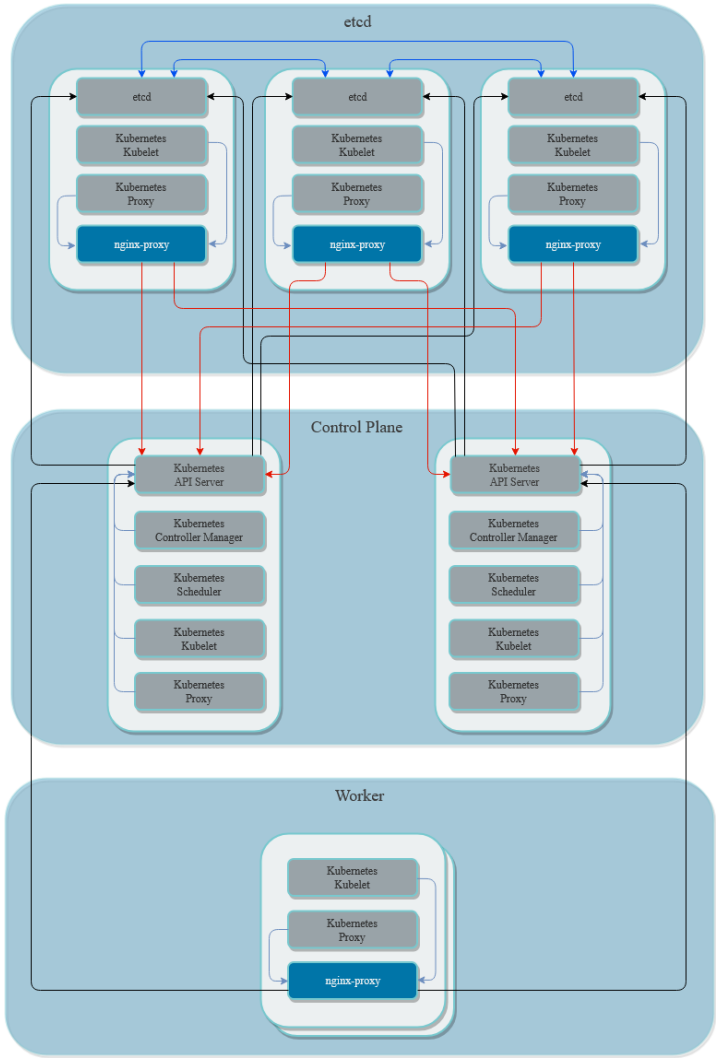
Kubernetes



[10] https://www.docker.com/blog/top-questions-docker-kubernetes-competitors-or-together/

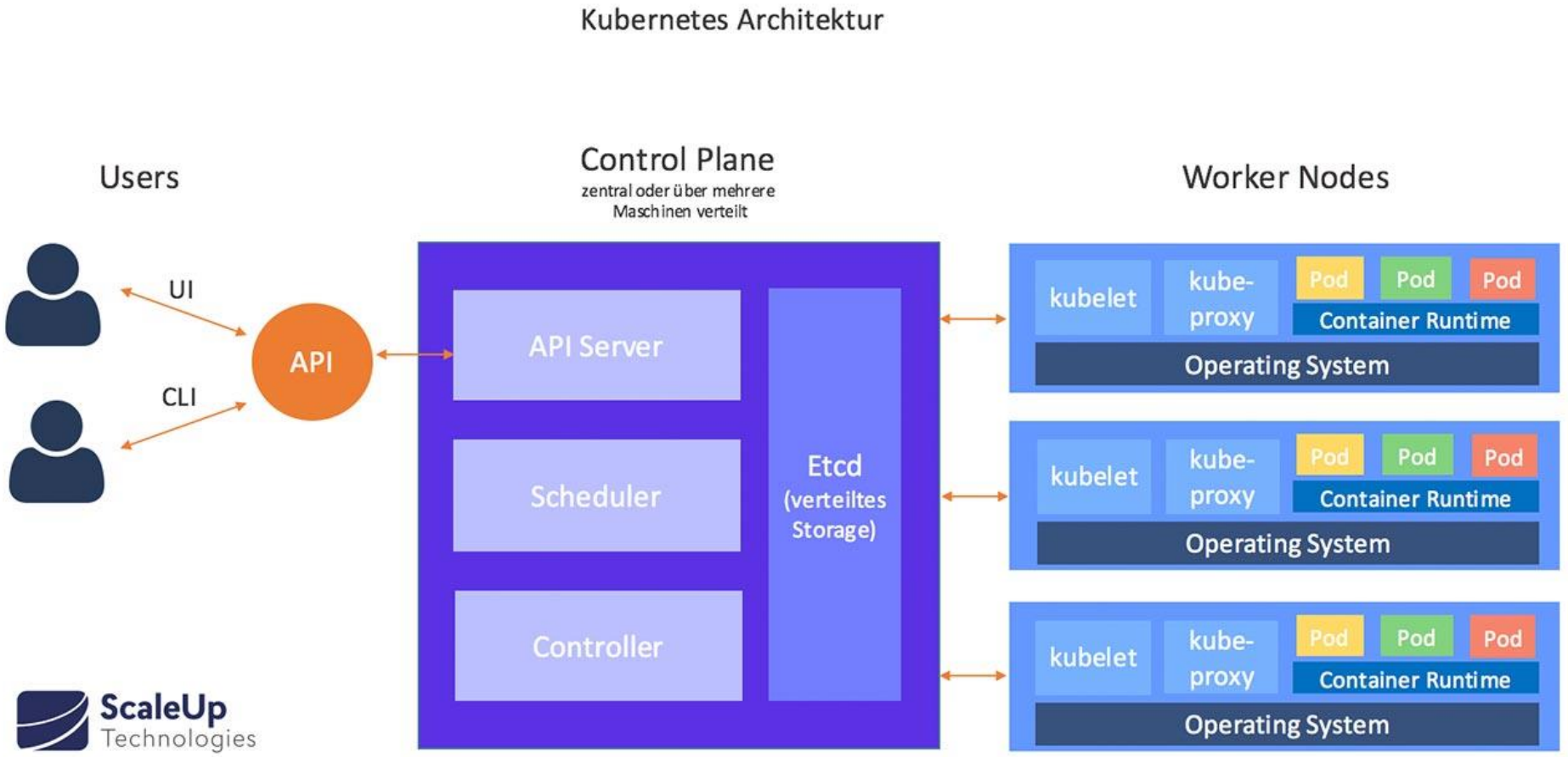
Kubernetes

- Ein Cluster besteht aus mehreren Servern (Nodes)
- Jeder Node hat eine Aufgabe (Roles)
  - Rolle: etcd
    - Verteilte Datenbank die den State des Clusters speichert
  - Rolle: Control Plane
    - Stellt APIs, Scheduler und Verwaltungsdienste bereit
  - Rolle: Worker
    - Führt Container aus



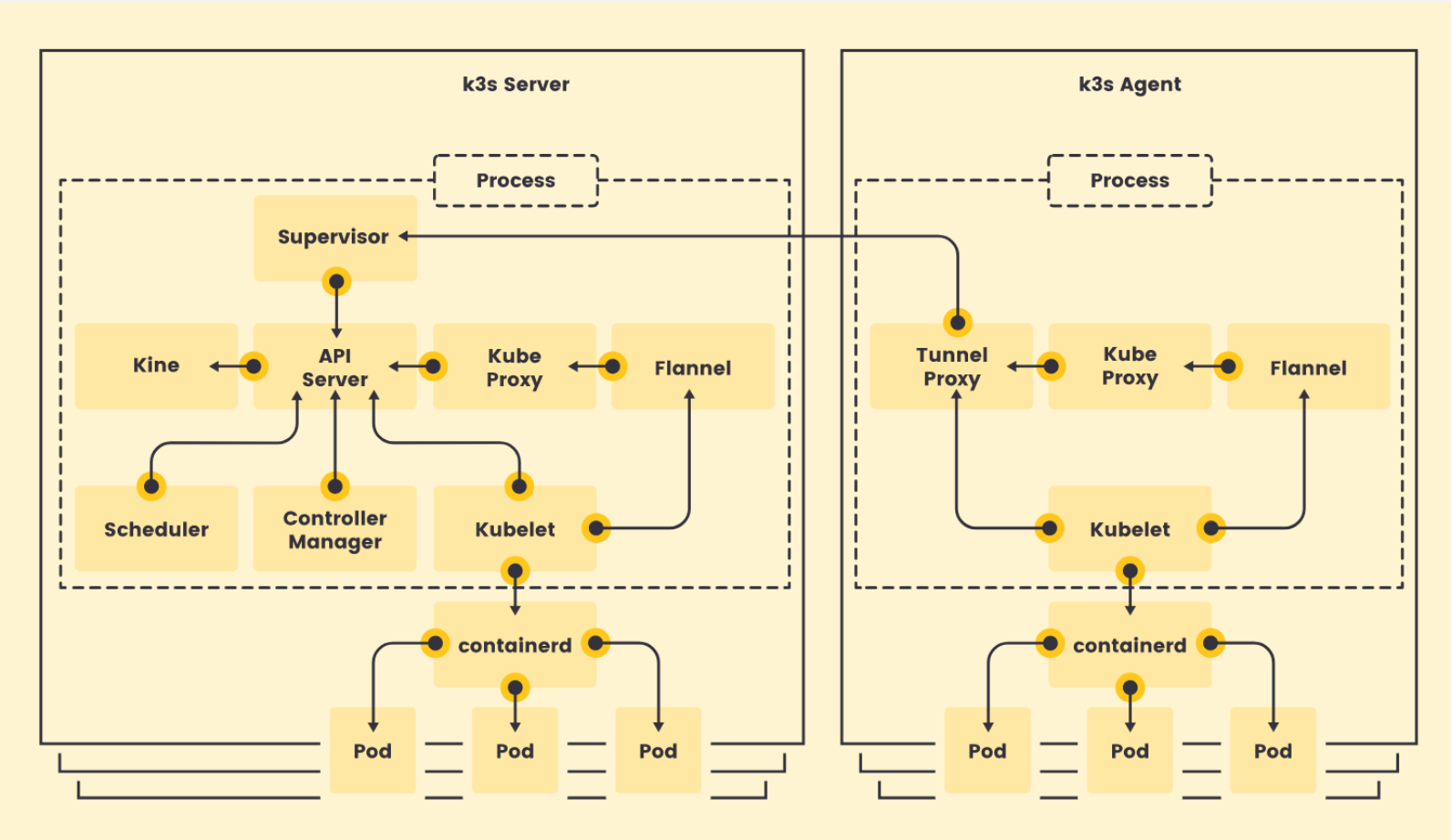
[11] https://ranchermanager.docs.rancher.com/how-to-guides/new-user-guides/kubernetes-clusters-in-rancher-setup/checklist-for-production-ready-clusters/roles-for-nodes-in-kubernetes

Kubernetes



[12] https://www.scaleuptech.com/blog/kubernetes-architektur/

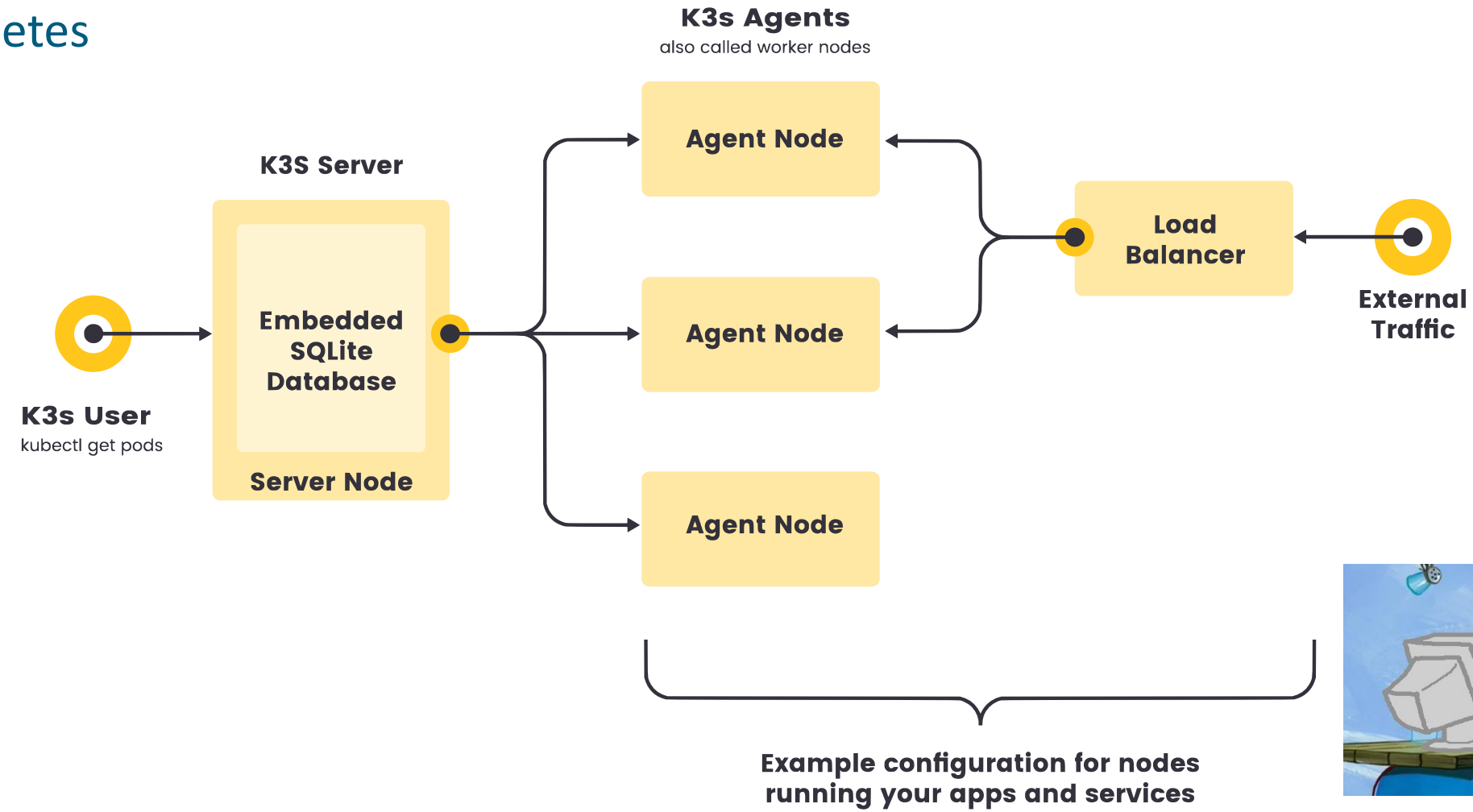
Kubernetes



[13] https://docs.k3s.io/architecture

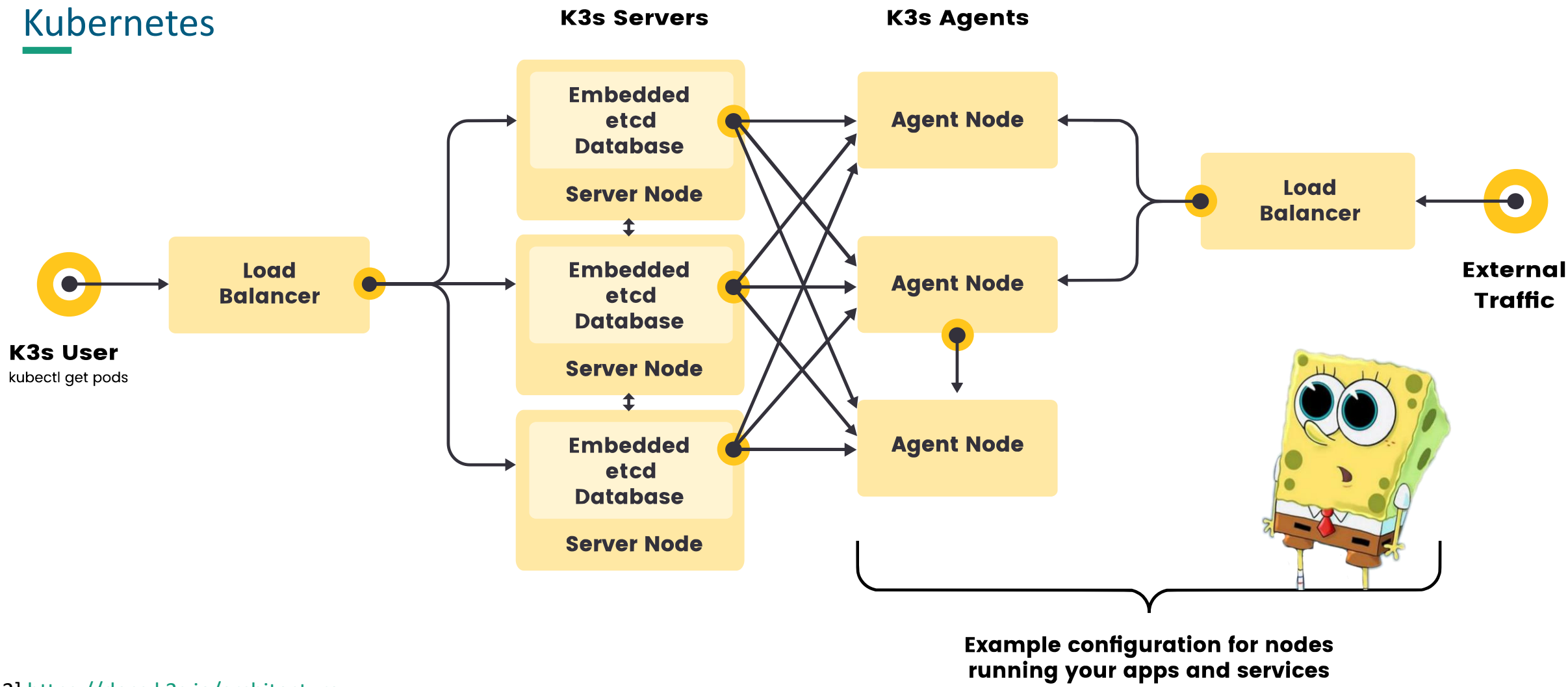
Fraunhofer IEE

Kubernetes



[13] https://docs.k3s.io/architecture

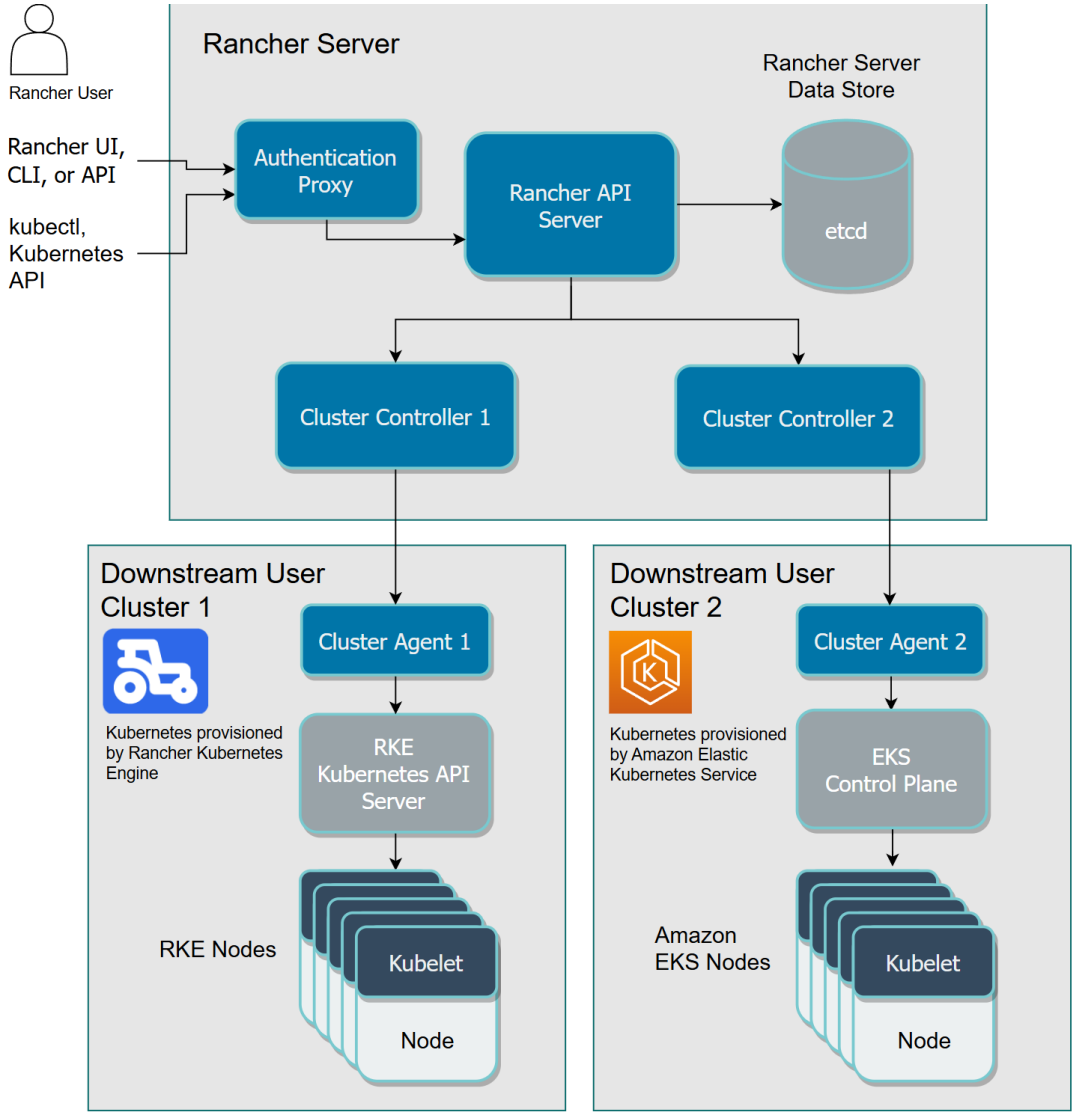
Kubernetes



[13] https://docs.k3s.io/architecture

Fraunhofer IEE

Kubernetes



[14] https://ranchermanager.docs.rancher.com/reference-guides/rancher-manager-architecture/rancher-server-and-components

Fraunhofer IEE

Kubernetes

```
FROM node:20.15.0-alpine

WORKDIR /usr/src/app

COPY . .

RUN npm ci
RUN npm run build

EXPOSE 3000
CMD [“npm“, “run“, “start:prod“]
```

Dockerfile

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-service
  namespace: production
  labels:
    app: my-service
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-service
  template:
    metadata:
      labels:
        app: my-service
    spec:
      containers:
        - image: my-service:0.1.0
          name: my-service
```

Kubernetes Manifest

[15] https://faun.pub/understanding-the-kubernetes-manifest-e96d680f2a11

Kubernetes - Zusammenfassung

- Mit **Kubernetes** können **Container** auf mehrere **Nodes** verteilt und betrieben werden
- Ein **Kubernetes Cluster** besteht aus mehreren Servern mit unterschiedlichen **Roles**
  - Role: **etcd**
    - Verteilte Datenbank die den State des Clusters speichert
  - Role: **Control Plane**
    - Stellt APIs, Scheduler und Verwaltungsdienste bereit
  - Role: **Worker**
    - Führt Container aus
- **Kubernetes** gibt es in verschiedenen Ausprägungen wie **K3S**, **minikube**, **EKS** (Amazon), …
- Für einen produktiv Einsatz empfiehlt es sich **Kubernetes** in einem **High-Availability** setup zu betreiben
  - Mehrere **Nodes** mit den gleichen **Roles**
  - Mehrere **Cluster** mit für unterschiedliche Aufgaben
- **Resourcen** in einem **Cluster** werden durch **Manifest** Dateien **deklarativ** beschrieben

Lernziele für Heute

1. Historie
2. Desktop vs. Cloud; Wie verteile ich meine Software?
3. Vorteile / Nachteile Cloud
4. Was macht DevOps aus?
5. Was bedeutet das für eine Softwarearchitektur?
6. Docker
7. Kubernetes