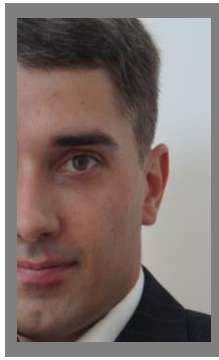
Microservice Architekturen in der Software-Entwicklung anhand einer aktuellen Containertechnologie



Michael Roth

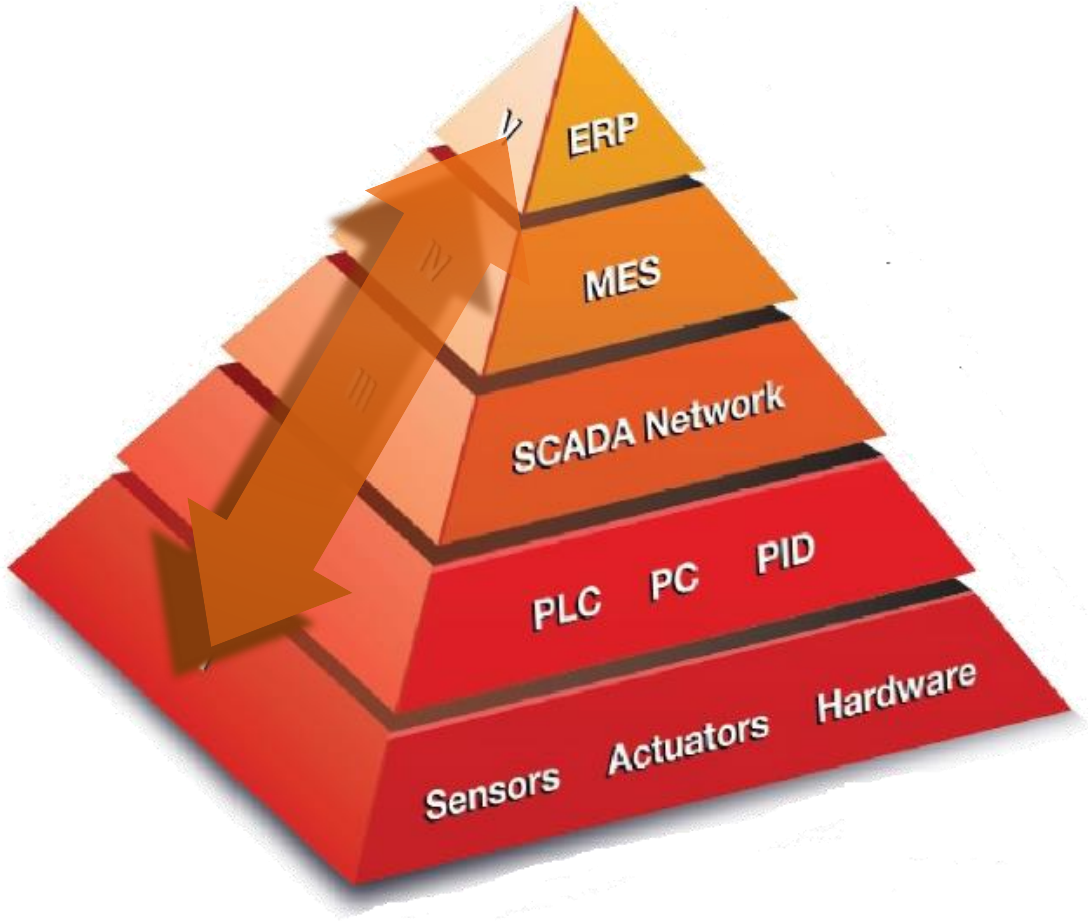
PhD Theses in Experimental Software Engineering

Vol. 59

Michael Roth

Qualitative Reliability Analysis of Software-controlled System using State/Event Fault Trees

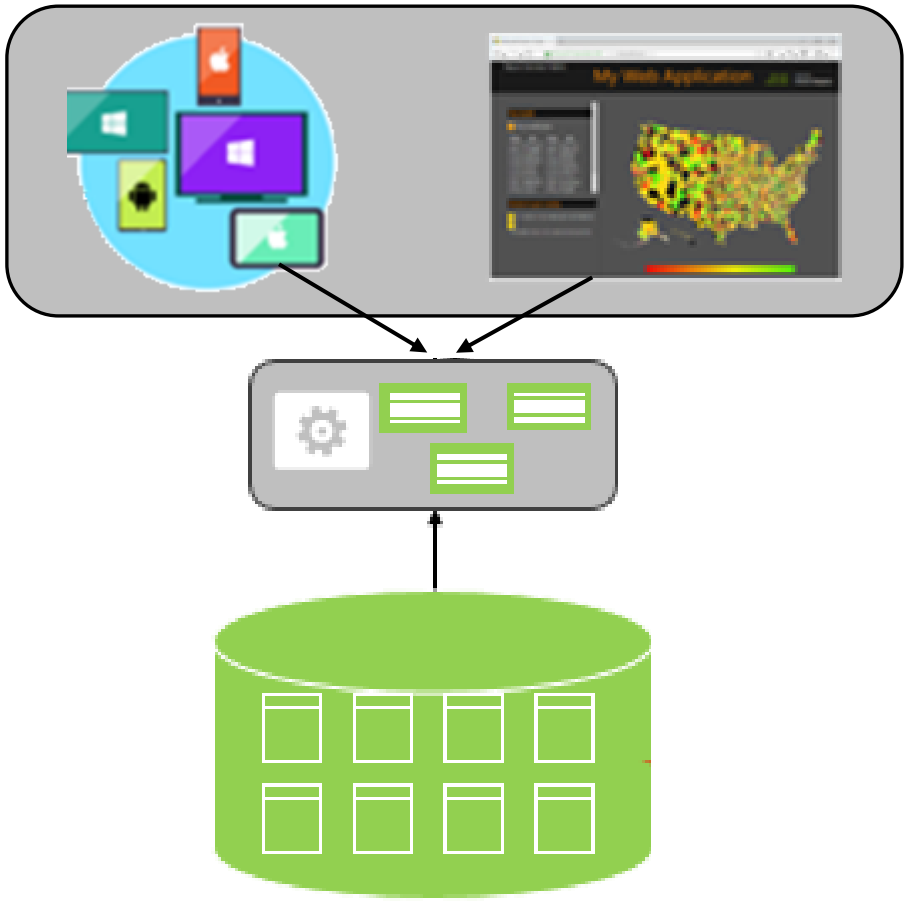
Fraunhofer Verlag


I
ERP
IV
MES
III
SCADA Network
PLC PC PID
Sensors Actuators Hardware

Software Architektur

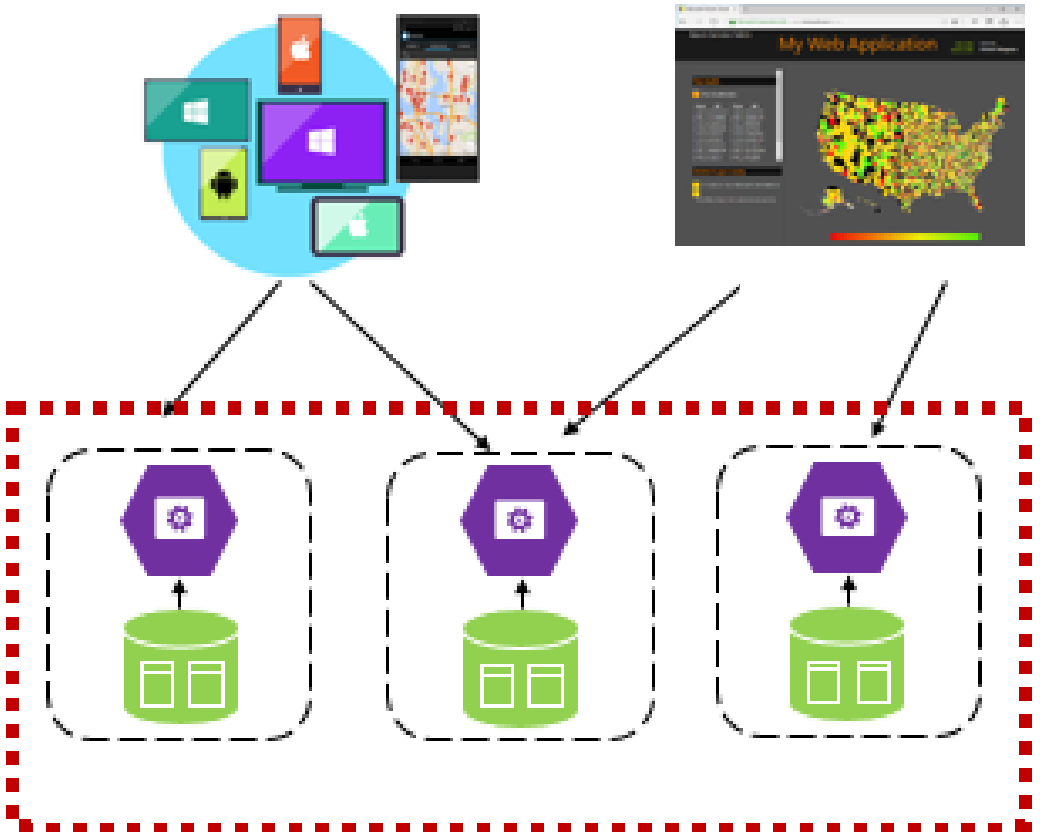
Monolith vs. Microservices

Monolith vs. Microservices



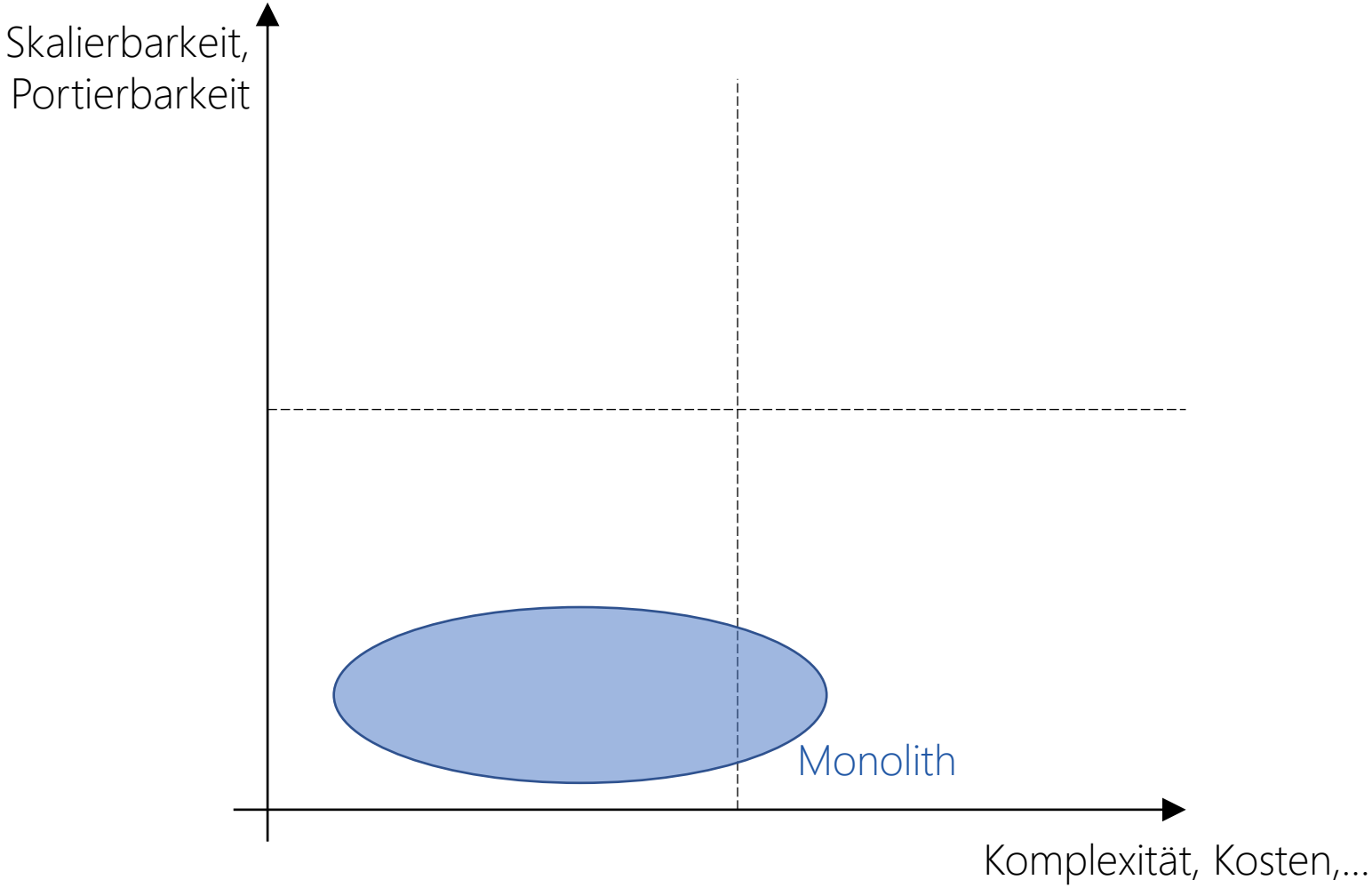
[*] Abbildung 1-4, https://learn.microsoft.com/de-de/dotnet/architecture/cloud-native/definition

Monolith vs. Microservices



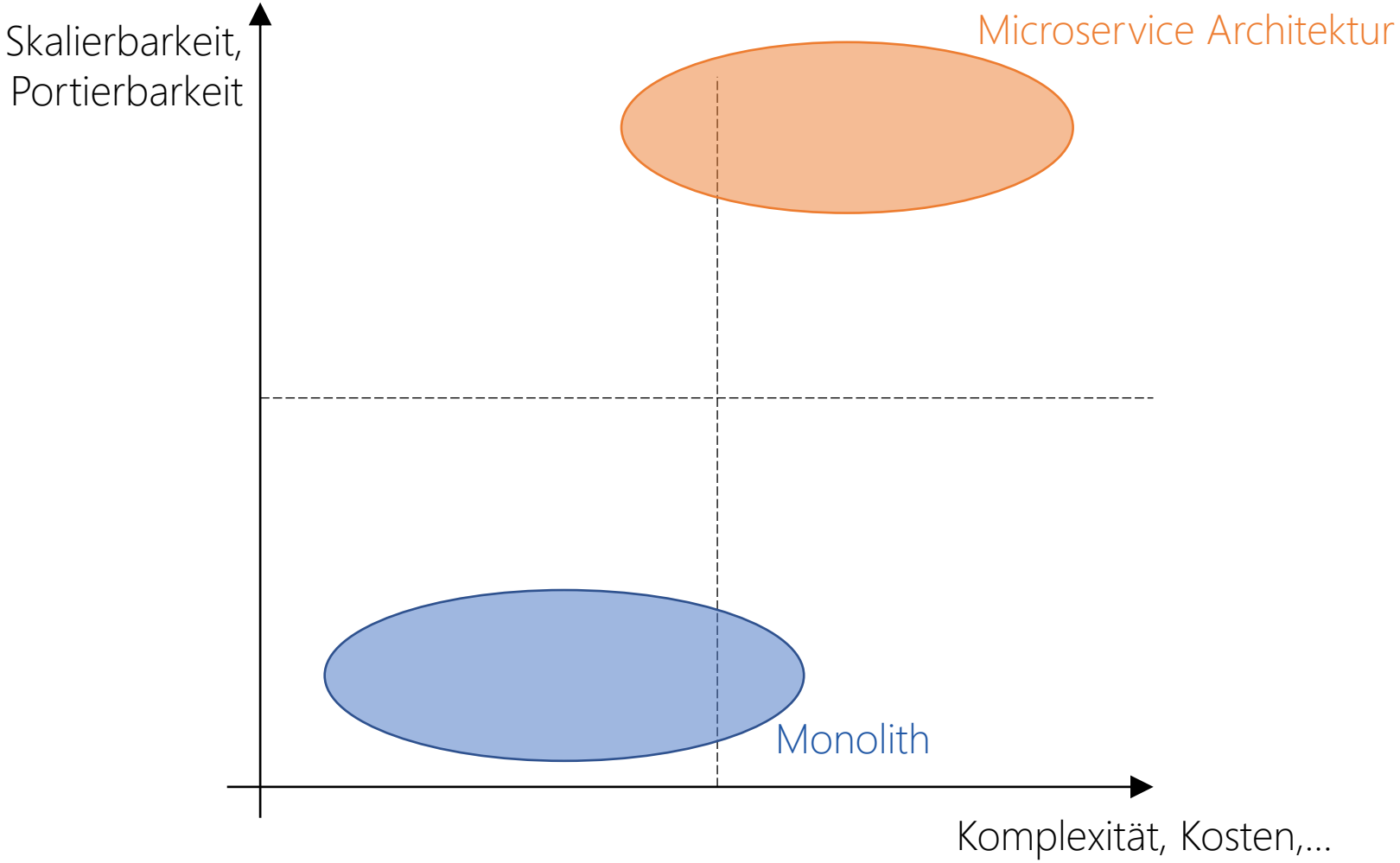
[*] Abbildung 1-4, https://learn.microsoft.com/de-de/dotnet/architecture/cloud-native/definition

Architektur – Vergleich



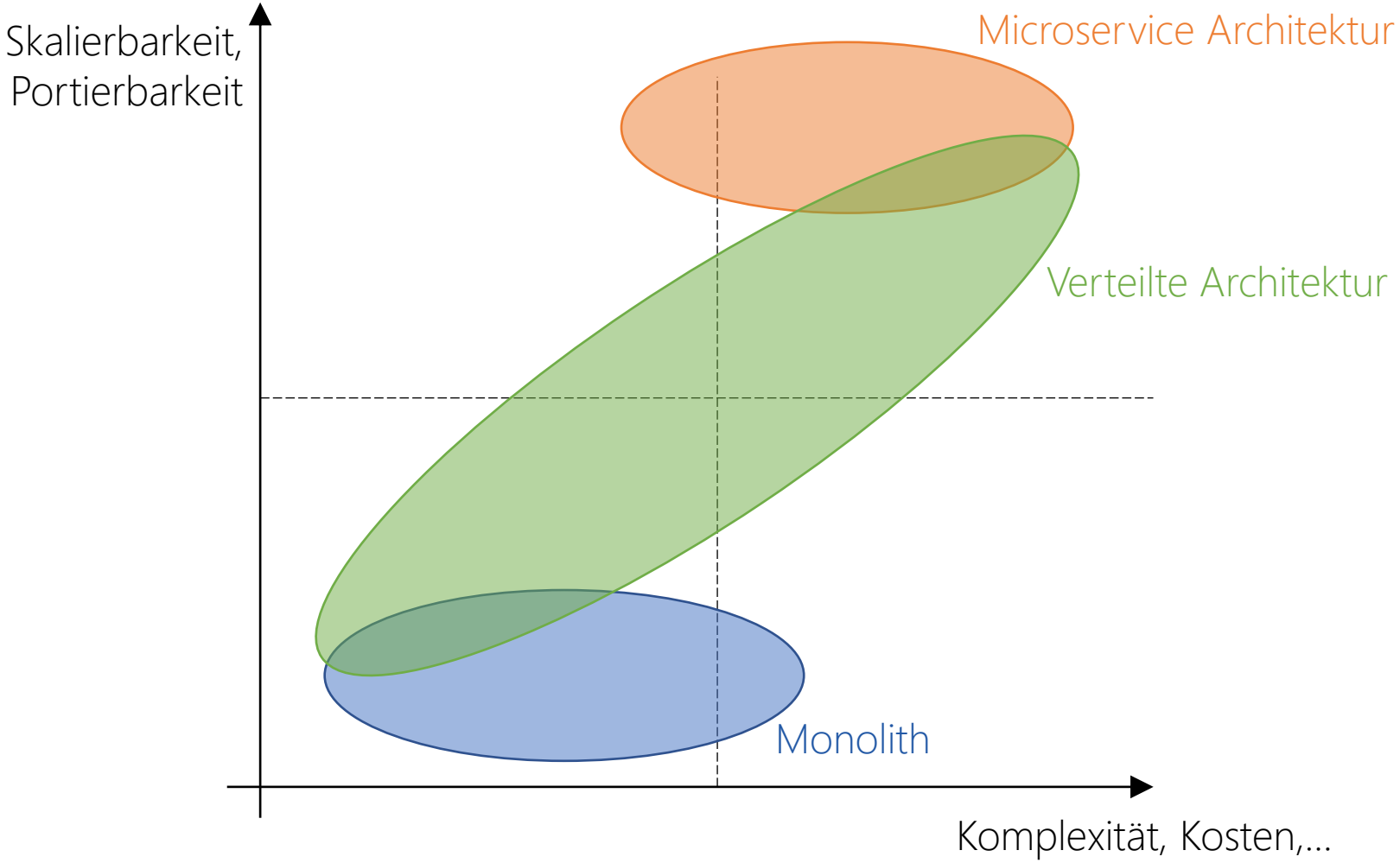
[*] https://entwickler.de/software-architektur/microservices-oder-monolithen-beides

Architektur – Vergleich



[*] https://entwickler.de/software-architektur/microservices-oder-monolithen-beides

Architektur – Vergleich



[*] https://entwickler.de/software-architektur/microservices-oder-monolithen-beides

Microservices – Take Aways

- Entkopplung
  - Konfigurationen
  - Daten
  - Schnittstellen
  - Organisationen
- Kohäsion (Single-Responsibility)
- Kapselung (Informationsausblendung)
- Skalierbarkeit

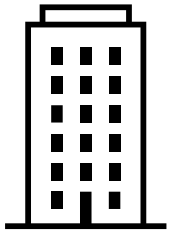
Architektur – Beispiel

- Im Industrieumfeld eines größeren Unternehmens werden Anlagen im Feld überwacht und deren Daten (Zeitreihen) aus verschiedenen Gründen langzeitarchiviert
- Diese Daten sollen einer Vielzahl an Applikationen und Unternehmensbereichen zugänglich gemacht werden
- Die Hersteller solcher Datenarchive stellen versionsgebundene SDKs / APIs zur Verfügung

Wie könnte eine (Microservice) Architektur für solch ein System aufgebaut sein?

Laufzeitumgebungen

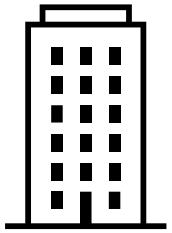
On-Premise

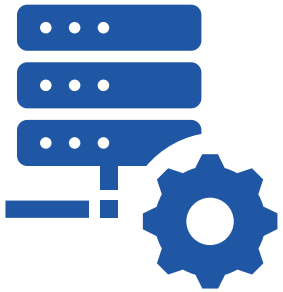


Laufzeitumgebungen

Virtuelle Umgebungen – IaaS

On-Premise



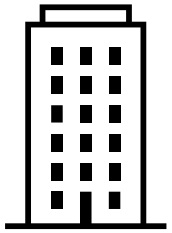


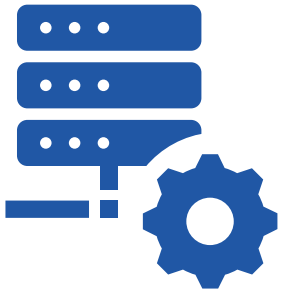
Laufzeitumgebungen

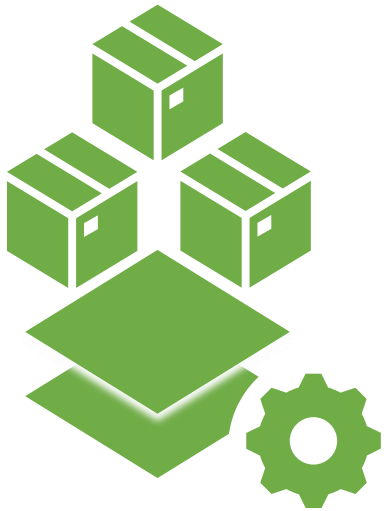
Container Umgebung – PaaS

Virtuelle Umgebungen – IaaS

On-Premise





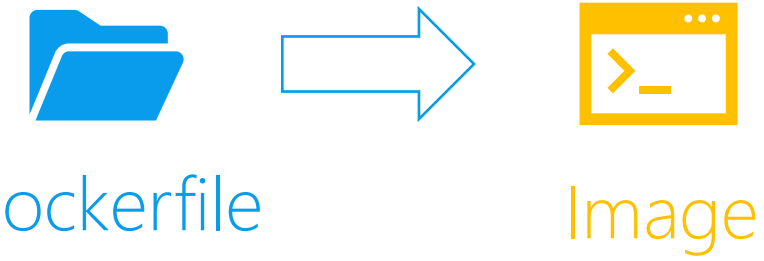


Cloud-native – Design [*]

- Microservices
- Container
- Backing Services
- Automatisierung (CI/CD)

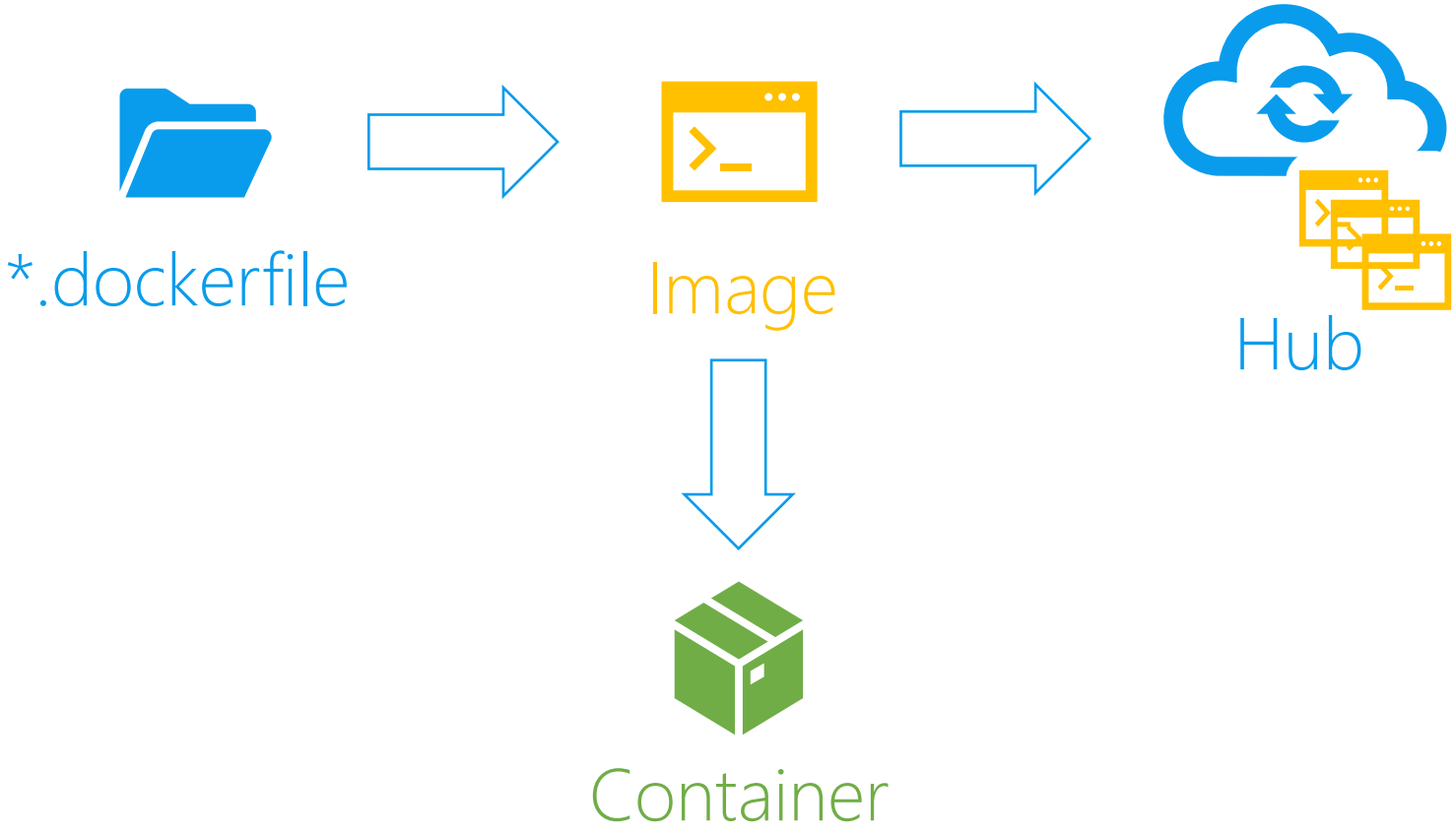
[*] https://learn.microsoft.com/de-de/dotnet/architecture/cloud-native/definition

Container – Docker

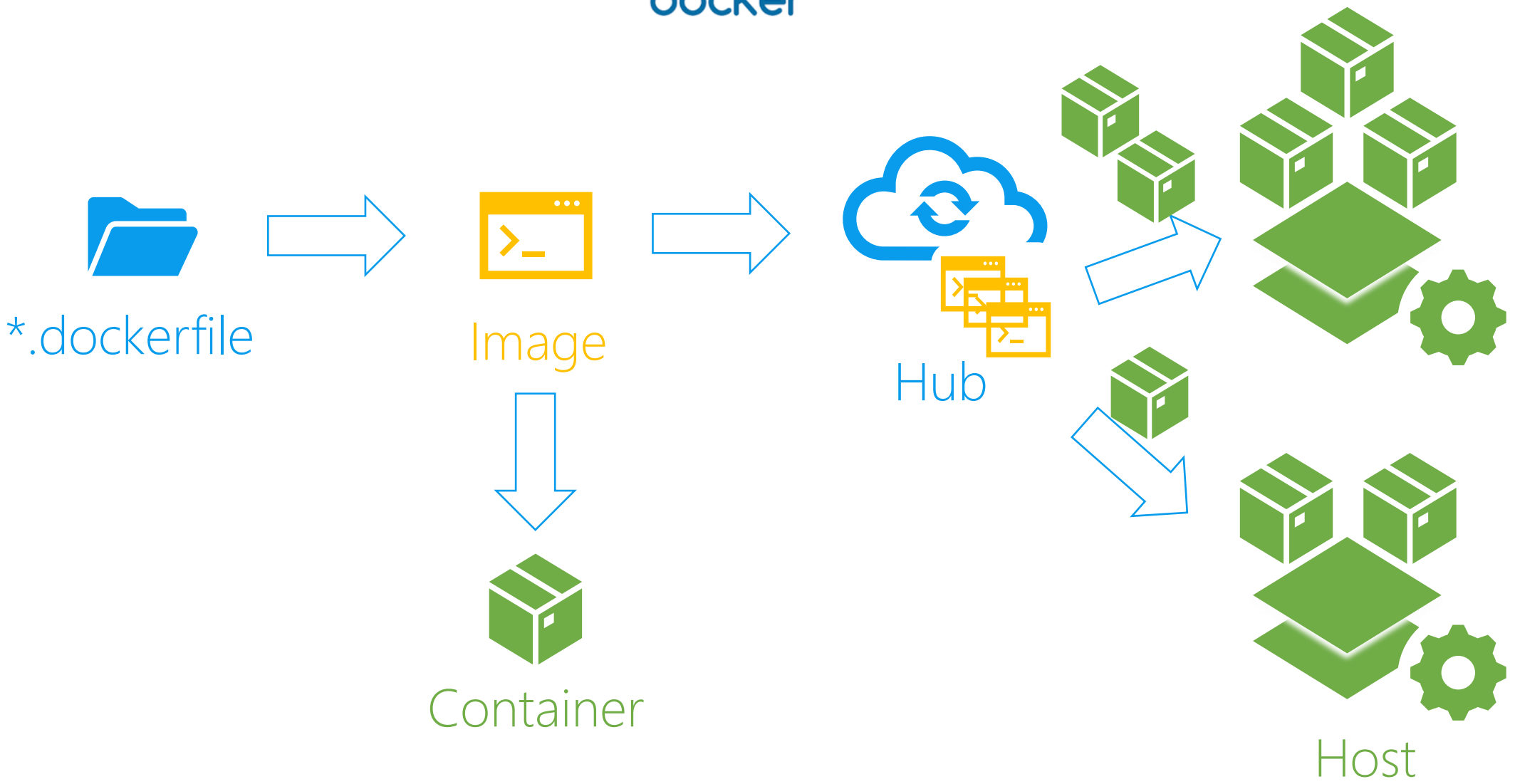



```
FROM mcr.microsoft.com/dotnet/aspnet:6.0 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build
WORKDIR /src
COPY ["WebApplication1.csproj", "."]
RUN dotnet restore "./WebApplication1.csproj"
COPY . .
WORKDIR "/src/."
RUN dotnet build "WebApplication1.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "WebApplication1.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "WebApplication1.dll"]
```

Container – Docker



Container – Docker



Cloud-native – Take Aways

- Unabhängige Bereitstellung
- Bereitstellung ohne Downtime
- Bessere Möglichkeiten der Skalierung
- Portierbarkeit
- Bessere Testbarkeit
- Isolation
- Automatisierung

Cloud-native – Take Aways

- Unabhängige Bereitstellung
- Bereitstellung ohne Downtime
- Bessere Möglichkeiten der Skalierung
- Portierbarkeit (Concurrency)
- Bessere Testbarkeit
- Isolation
- Automatisierung

- Kosten („Microservice Architectures buy you options“ [*])

[*] James Lewis, Software Architect and Director, ThoughtWorks

noch Fragen?