DIGIVATION VERANSTALTUNG

MultiCloud basierte Dienstleistungen für die Produktion

Stuttgart, den 13. Juni 2018

Universität Stuttgart
ISW

Timur Tasci, M.Sc.

Wissenschaftlicher Mitarbeiter

Institut für Steuerungstechnik der Werkzeugmaschinen

und Fertigungseinrichtungen ISW - Universität Stuttgart

Seidenstraße 36

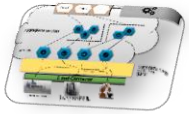
70174 Stuttgart

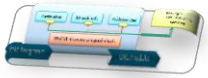
Telefon +49 711 685 - 82433| Fax +49 711 685 – 82808

Timur.Tasci@isw.uni-stuttgart.de

Universität Stuttgart
ISW

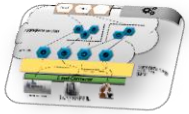
Inhalt

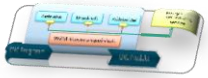




- Ausgangssituation
- Zielvorstellung
- Erste Ergebnisse
- Ausblick

Inhalt

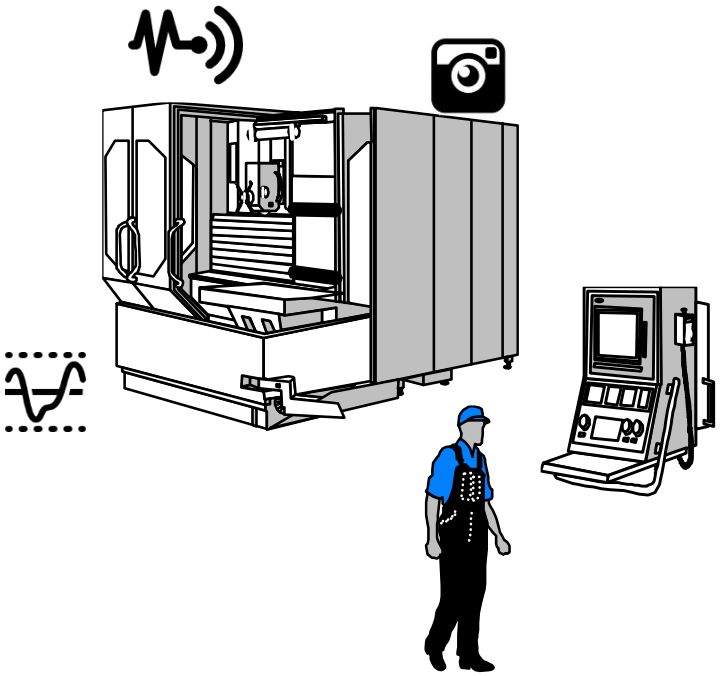




- Ausgangssituation
- Zielvorstellung
- Erste Ergebnisse
- Ausblick

Ausgangssituation

Potential vorhandener Datenquellen wird nicht genutzt!



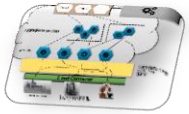
- Zahlreiche Datenquellen im Produktionsumfeld vorhanden
  - Steuerungen
  - Sensoren
  - Qualitätsdaten
  - MES
  - ERP
  - …
- Keine stand. Nutzung der Daten

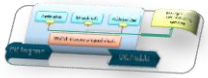
→ Zugriff auf Datenquellen ist möglich

Ausgangssituation
Services und Plattformen

- Aufwendiger Entwicklungs- und Integrationsprozess der Services
  - Schnittstellen zu den Datenquellen
  - Serviceentwicklung
  - Integration in die Produktionsumgebung
- Einzellösungen von Services
  - Sharing-Modelle können nicht genutzt werden
- Bestehende Lösungen bieten meist nur eine Plattform an
  - Es werden Faktoren wie Leistungsfähigkeit, Verlässlichkeit, Kosten und Serverstandort nicht berücksichtigt

Inhalt

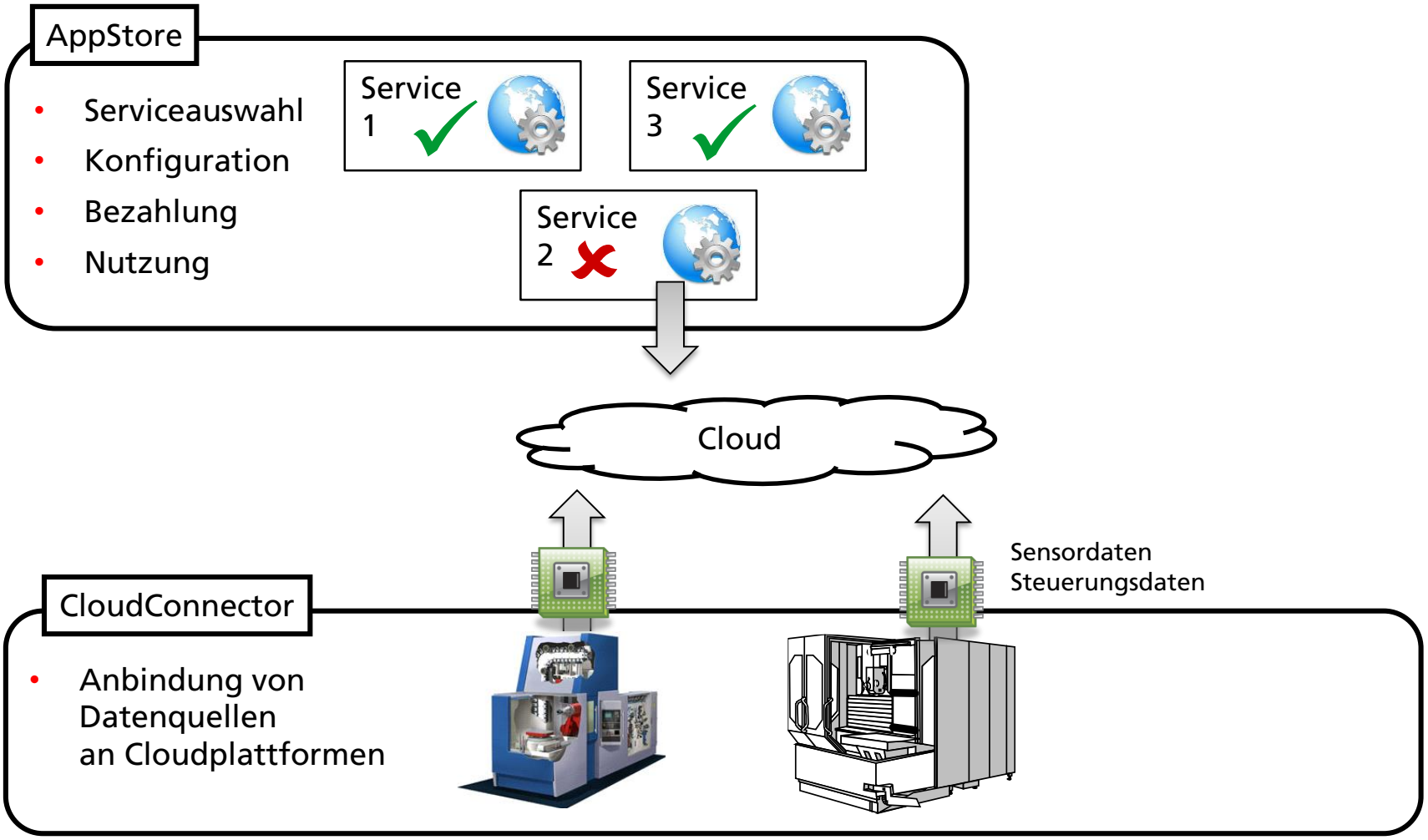




- Ausgangssituation
- Zielvorstellung
- Erste Ergebnisse
- Ausblick

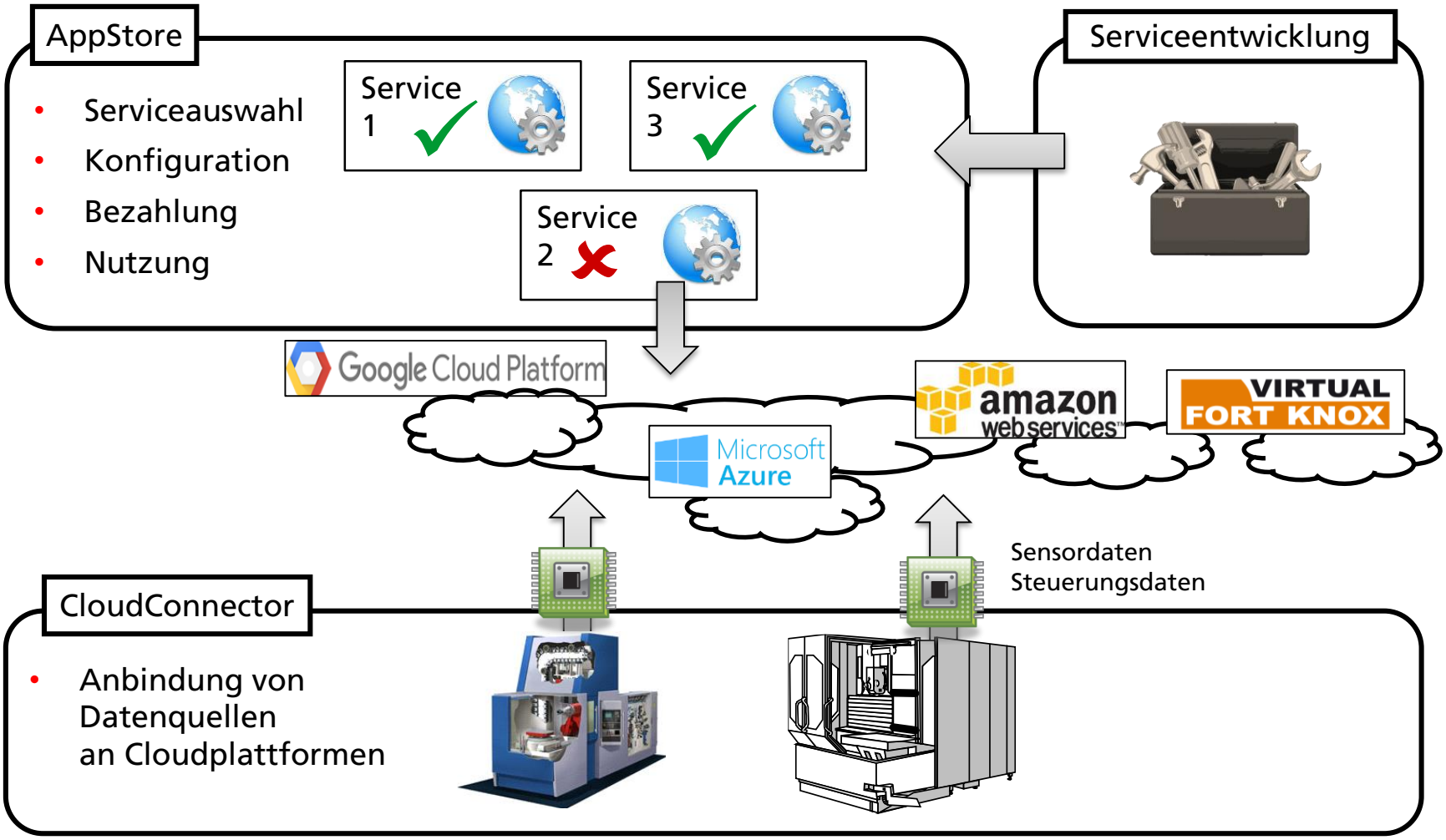
Ziel MultiCloud:

Minimierung des Entwicklungs- und Integrationsaufwand



Ziel MultiCloud:

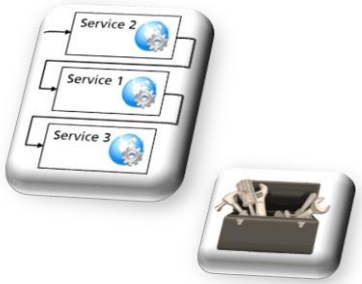
Minimierung des Entwicklungs- und Integrationsaufwand



Erwartete Ergebnisse

- Minimaler Zeitaufwand von Serviceauswahl bis zur Servicenutzung
  - Nutzung von Datenquellen
  - Konfiguration Services

- Reduktion des Entwicklungsaufwands von Services
  - durch Wiederverwendbarkeit und Orchestrierung
  - durch Entwicklungswerkzeug (Libs,…)



- Minimale Kosten für Kunden
  - Sharing Modelle: Kunden bezahlen nur für genutzte Leistung
  - Keine eigene Infrastruktur notwendig

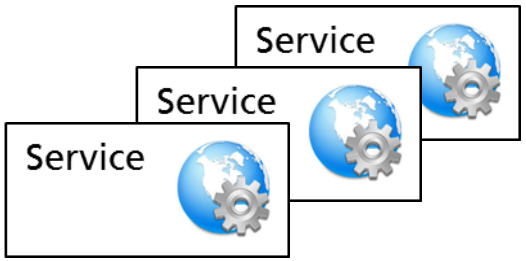
- Wandlungsfähig, kundenorientiert und technologie-unabhängig durch jederzeit möglichen Wechsel der Cloudplattform

Konsortium

Serviceentwicklung

- Data Mining
- Predictive Maintenance
- …

MAX-CON DATA SCIENCE
Rexroth Bosch Group



Infrastruktur

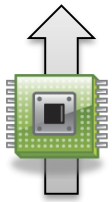
- AppStore
- Cloudplattformen
- Entwicklungswerkzeuge

Fraunhofer IPA
ISW
SOTEC
Rexroth Bosch Group

Datenintegration

- Steuerungen
- Aktoren, Sensoren
- MES, BDE

Rexroth Bosch Group
ISW
SOTEC

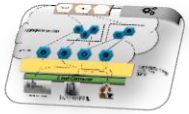


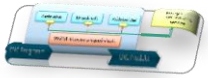
Anwender

- Datenlieferant
- Ideen für Services
- Nutzung der Services

KHS Filling and Packaging — Worldwide
wepa

Inhalt

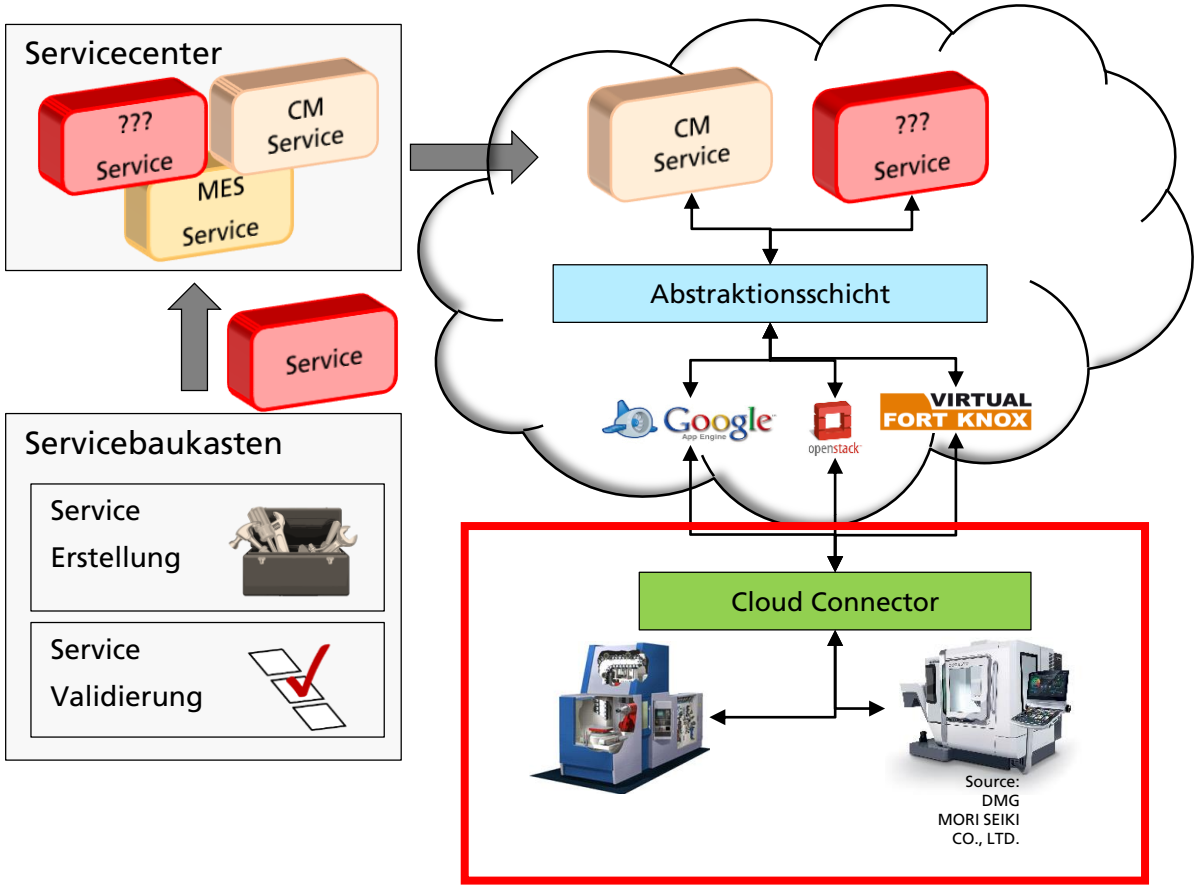




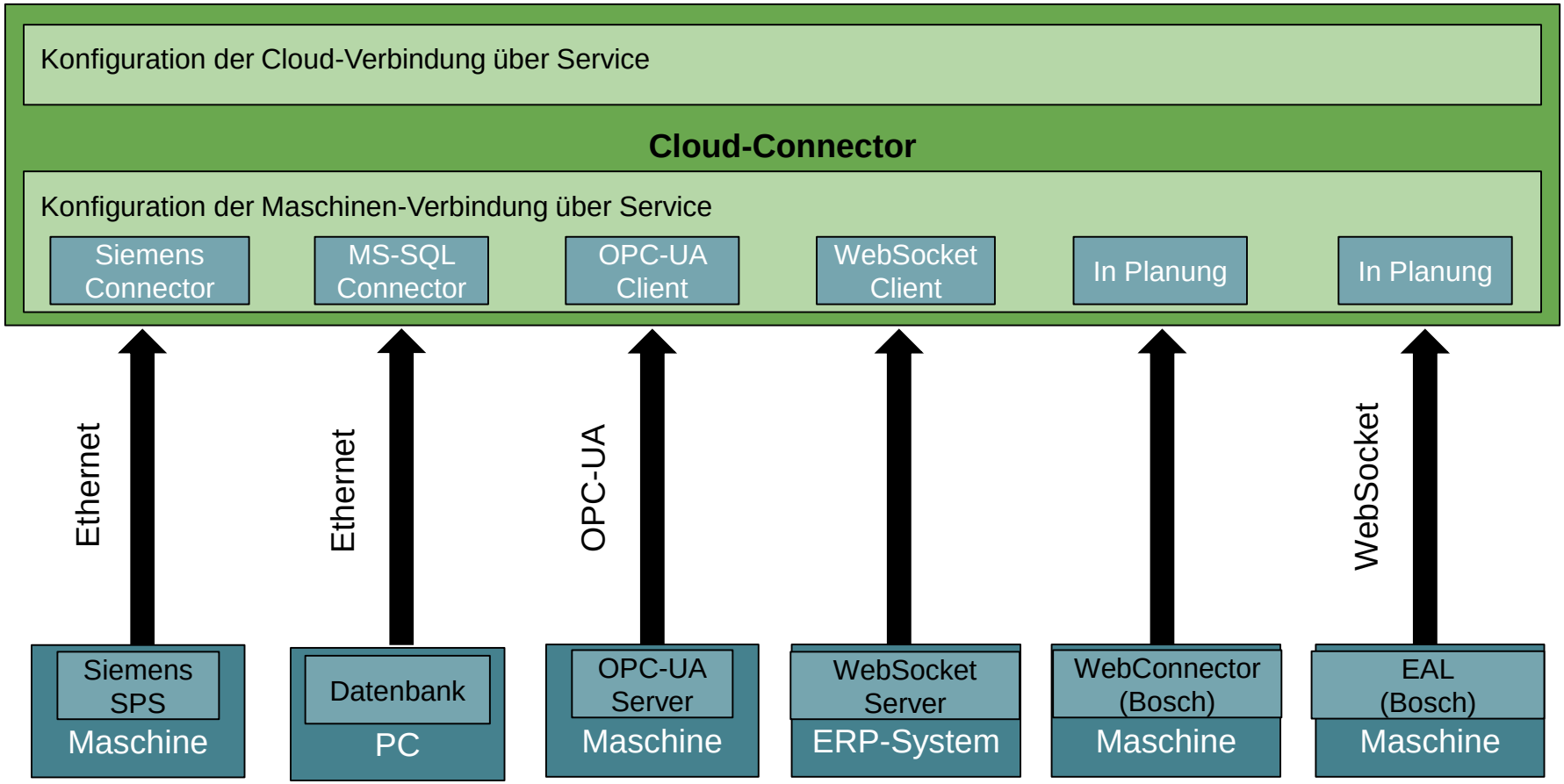
- Ausgangssituation
- Zielvorstellung
- Erste Ergebnisse
- Ausblick

Zielvorstellung

Erste Ergebnisse anhand der Gesamtarchitektur

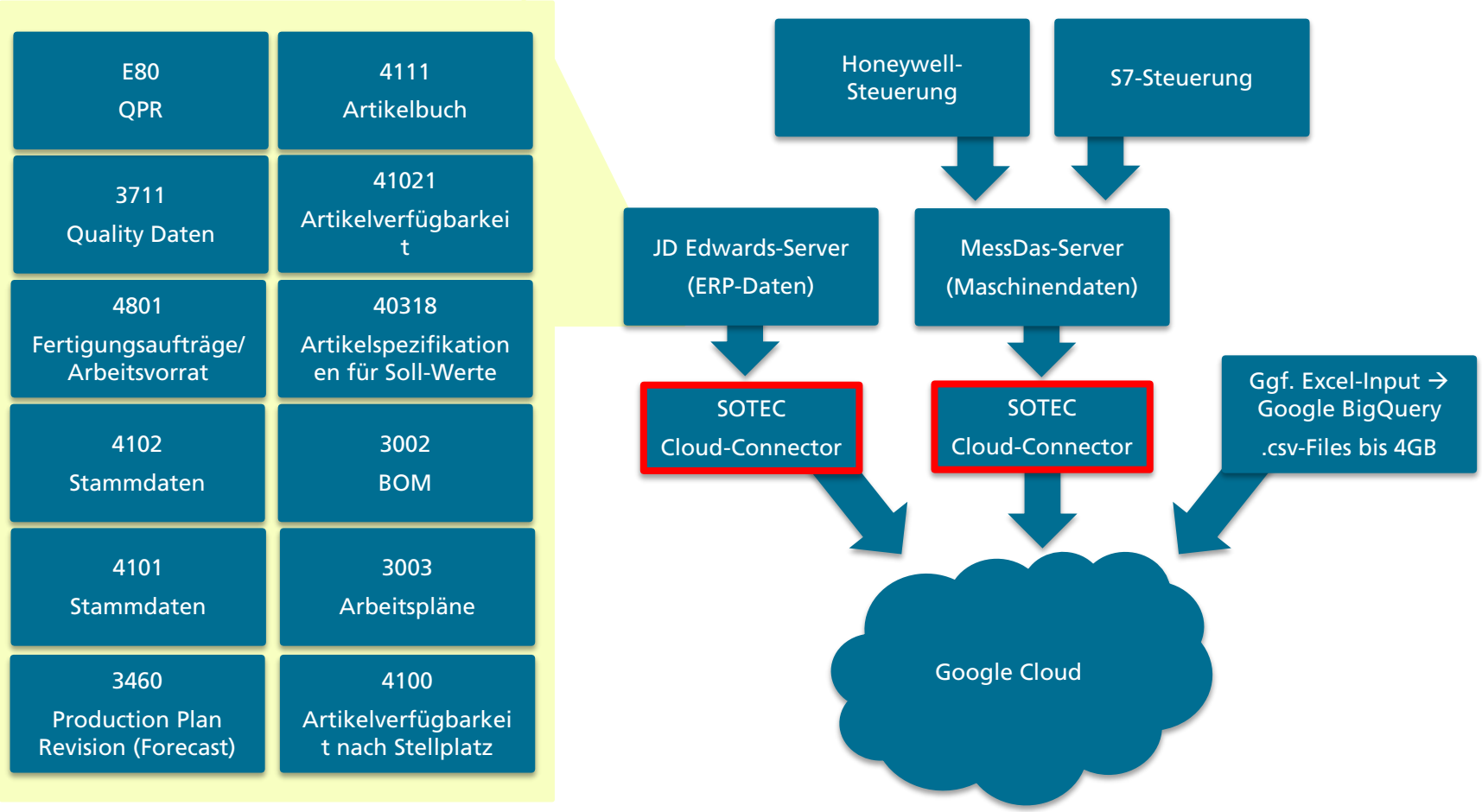


Erste Ergebnisse
Cloud-Connector



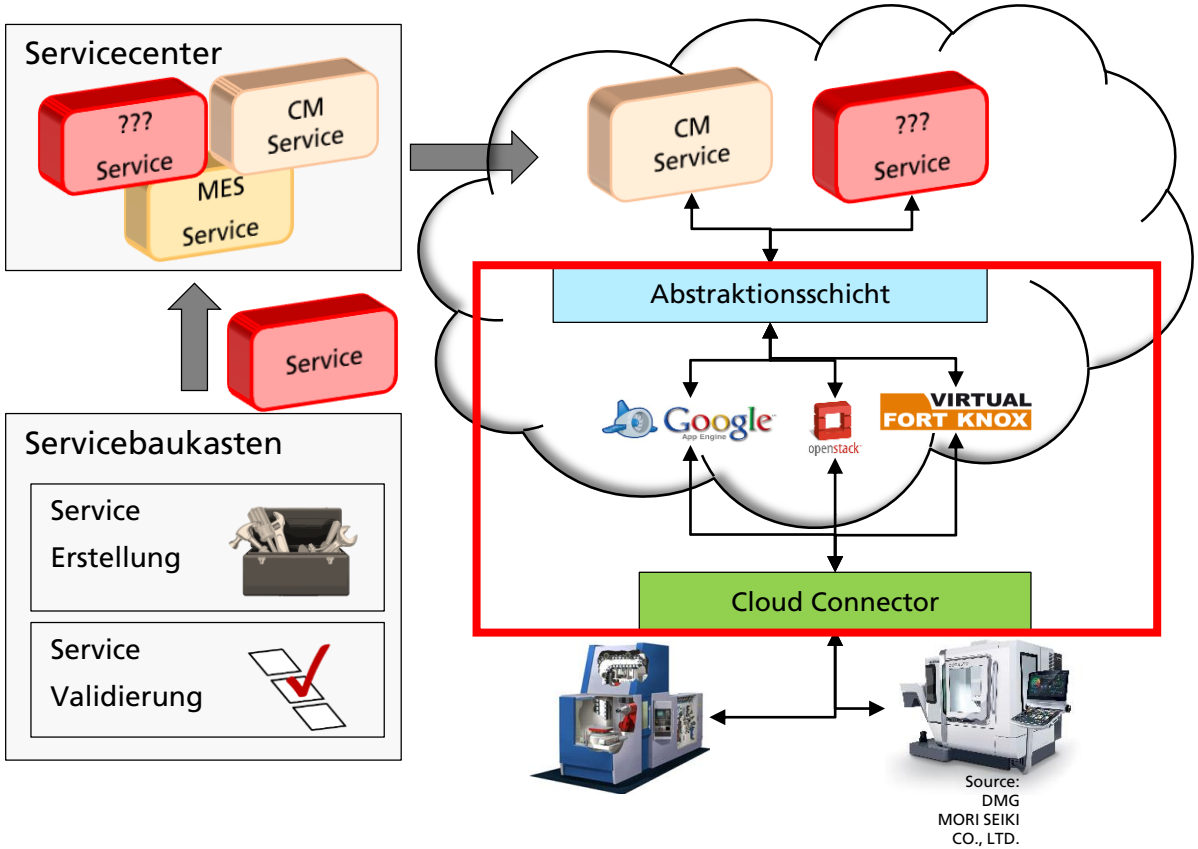
Erste Ergebnisse
Datenintegration und Datenbereitstellung

- Cloud-Anbindung beim Anwender WEPA

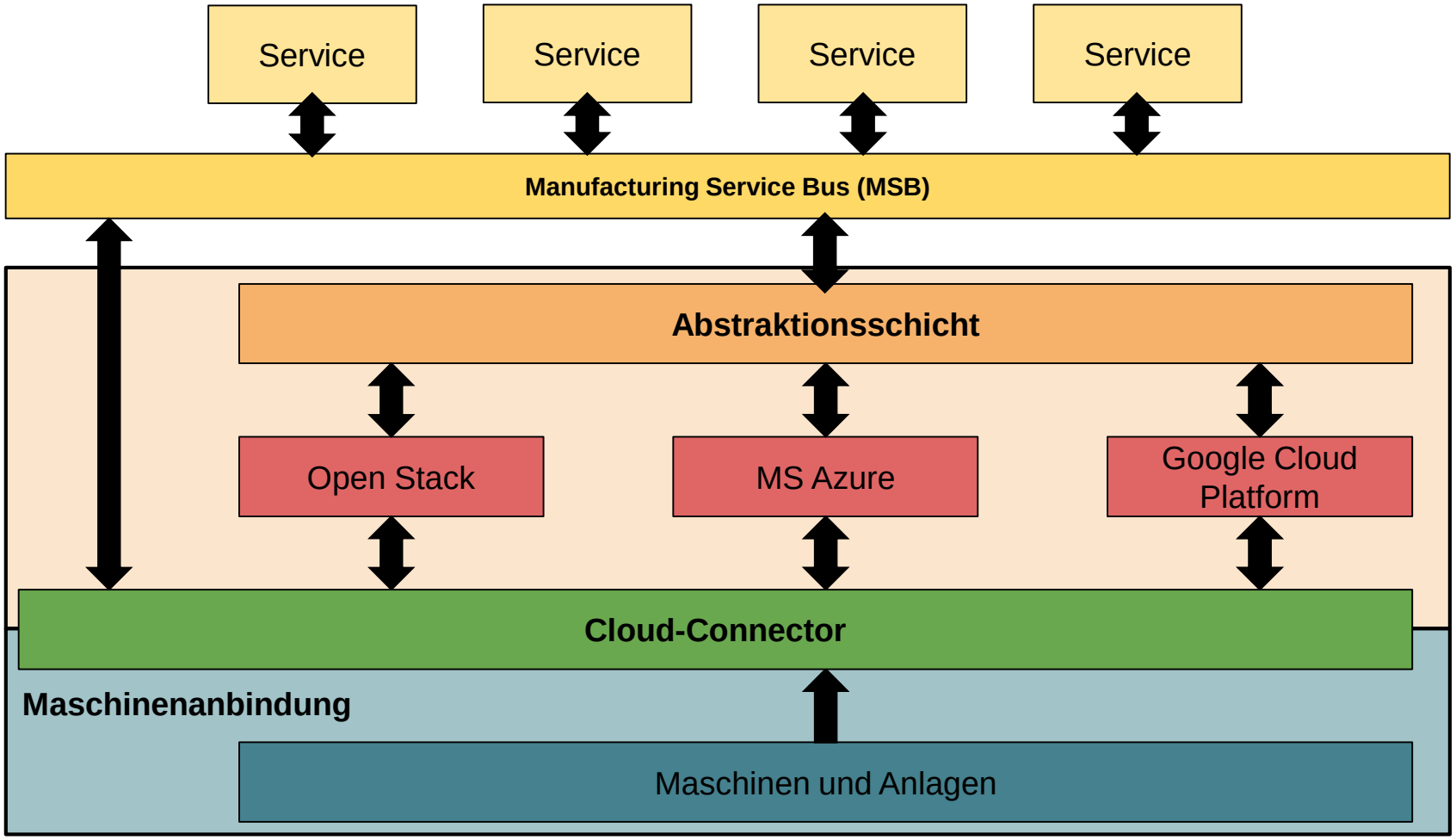


Zielvorstellung

Erste Ergebnisse anhand der Gesamtarchitektur

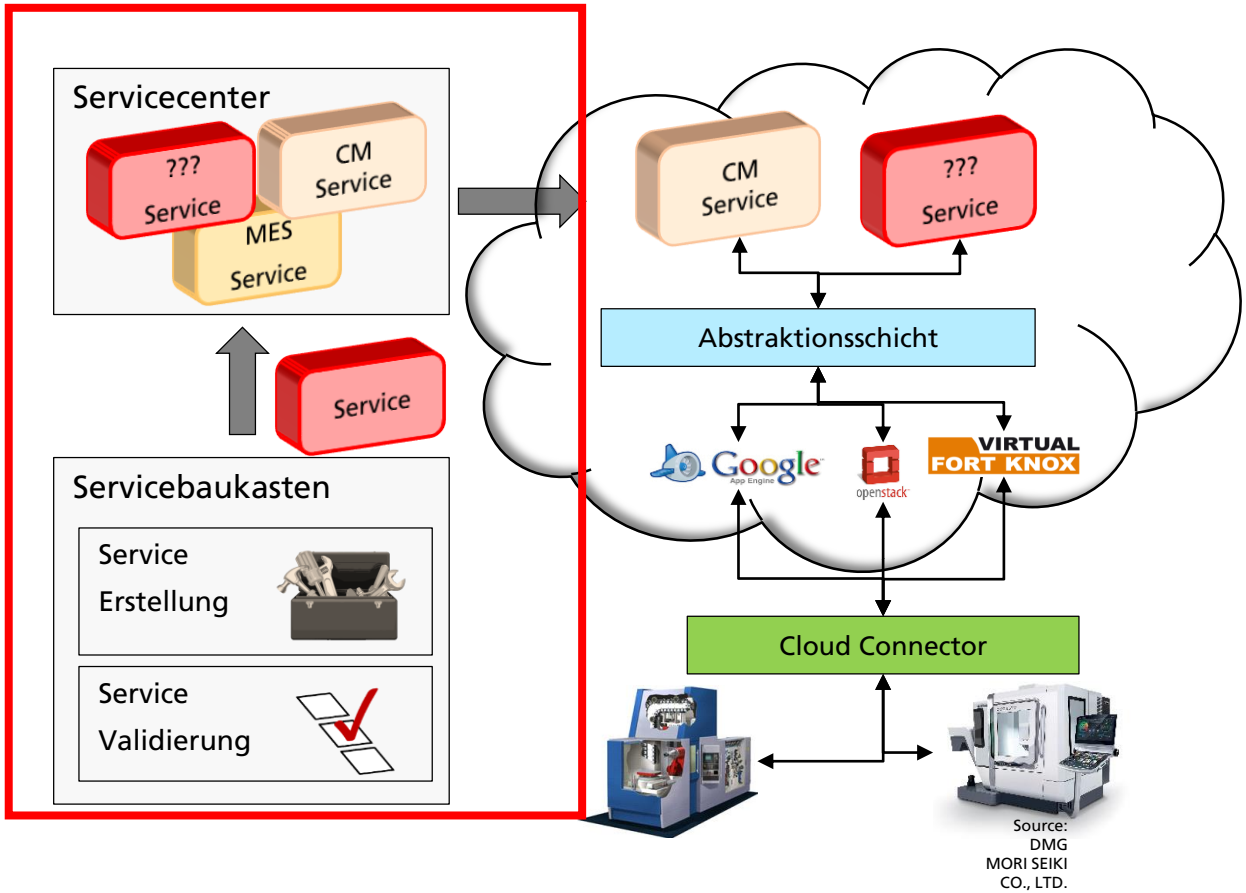


Erste Ergebnisse
Abstraktionsschicht



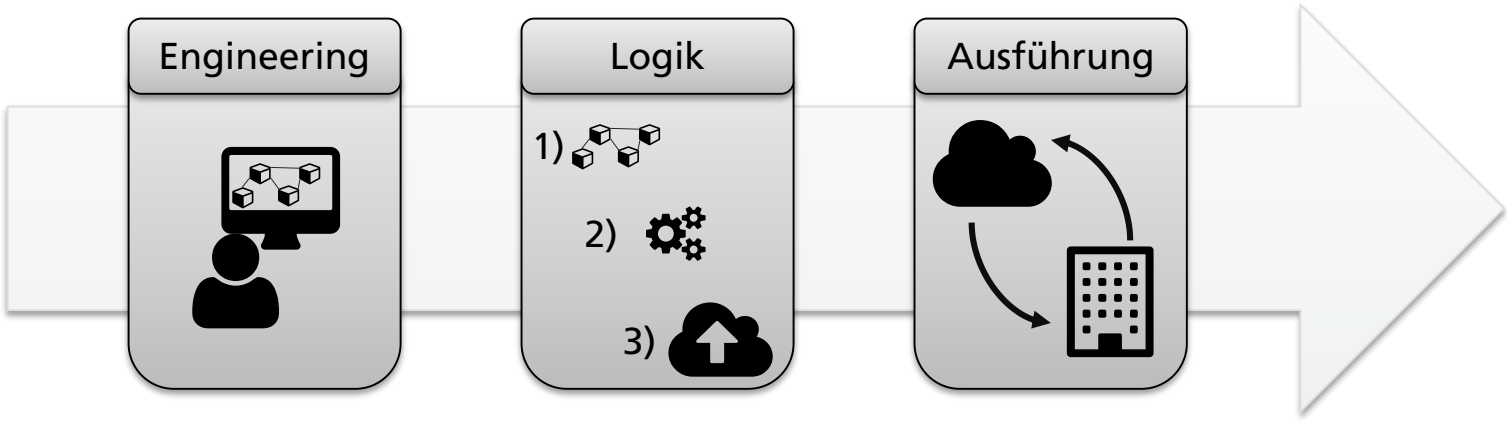
Zielvorstellung

Erste Ergebnisse anhand der Gesamtarchitektur



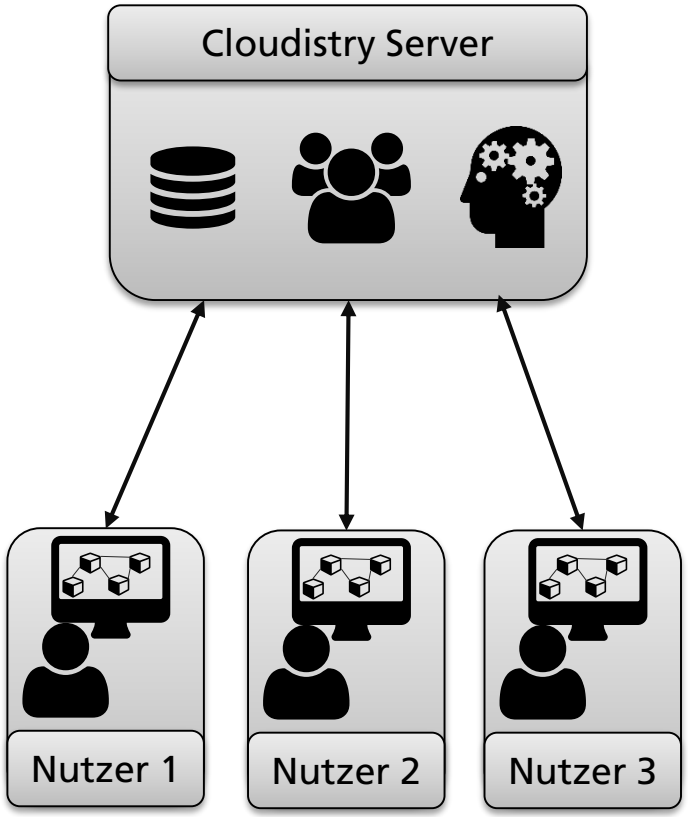
Konzept des Servicesystems
Grundlagen

- Unterteilung in drei Komponenten:
  - Grafische Benutzeroberfläche für das **Engineering**
  - Engineering Backend mit der **Logik** zum Einrichten des Servicesystems
  - Plattform zum **Ausführen** der Service-Komposition



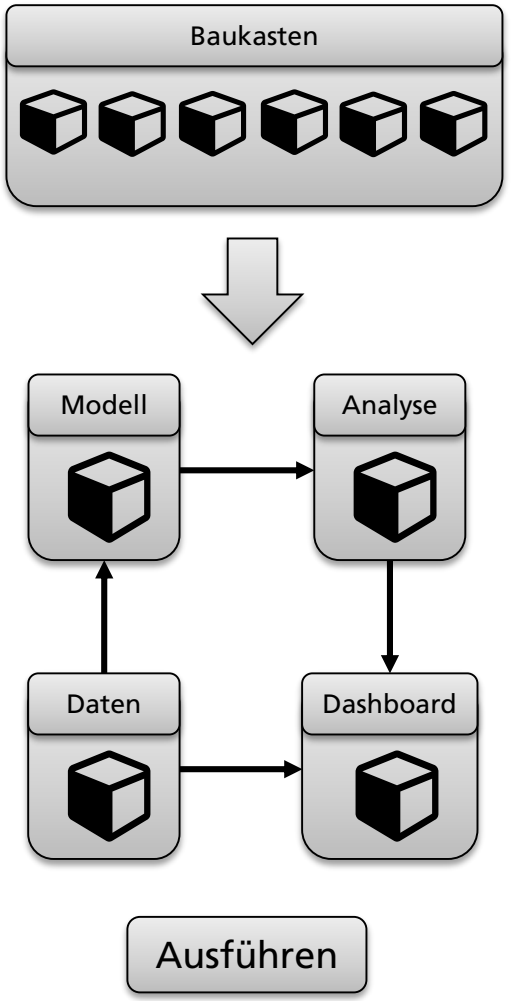
Konzept des Servicesystems
Engineering Komponente

- **Grafische Oberfläche** für die Nutzer des Servicesystems
- **Web-basiert** für jeden Nutzer ohne Installation erreichbar
- **Client**
  - Engineering
  - Serververwaltung
- **Server**
  - Speicher für Bausteine
  - Nutzerverwaltung
  - Logik-Komponente



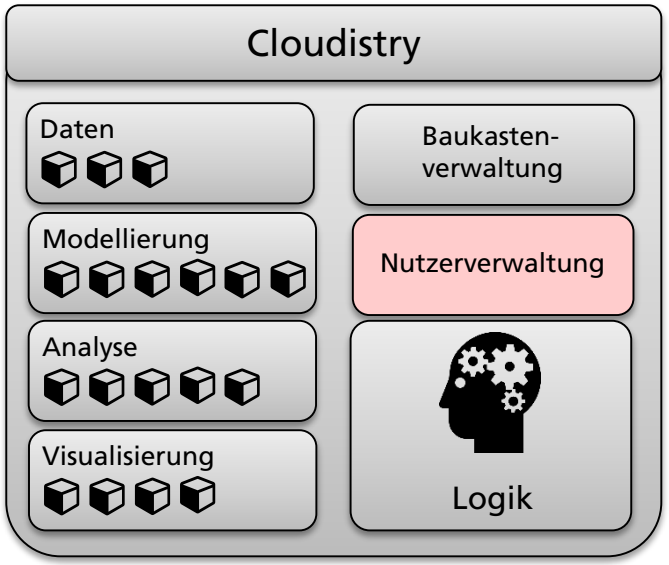
Konzept des Servicesystems
Engineering Komponente - Client

- **Bausteine** (Services) können aus einem **Baukasten** gewählt werden.
  - Auswahl von Datenquellen
  - Wahl eines Modellierungsverfahrens und eine anschließenden Analyse
  - Wahl einer Rückführungsmöglichkeit (Dashboard, andere Visualisierungen)
- Bausteine **grafisch verbinden**, um **Informationsfluss** zwischen den Services festzulegen
- Festlegen, wo ein Baustein ausgeführt werden soll
- Per Knopfdruck alle **Bausteine** gleichzeitig **starten**



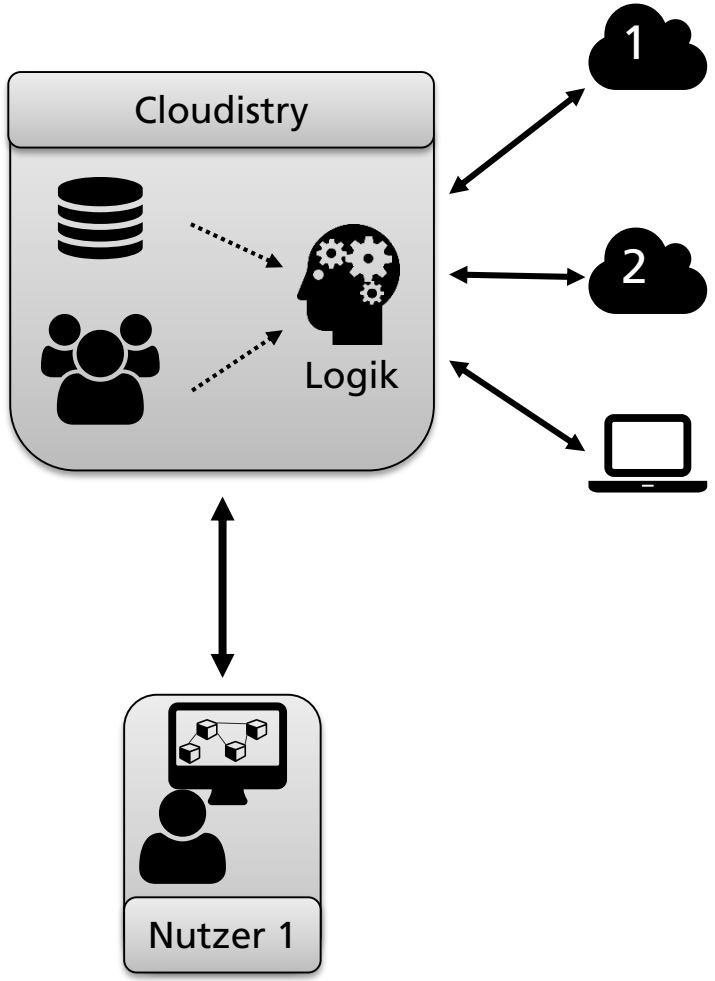
Konzept des Servicesystems
Engineering Komponente - Server

- Server enthält folgende Teile
  - **Baukästen** inklusive der Services, verpackt in einzelne Bausteine
  - **Verwaltungskomponente**, um Baukästen zu verwalten
    - Bausteine hinzufügen, updaten, löschen etc.
  - **Nutzerverwaltung**, um rollenbasierte Rechte zu verwalten
  - **Logik**, die Eingaben aus dem Engineering verarbeitet

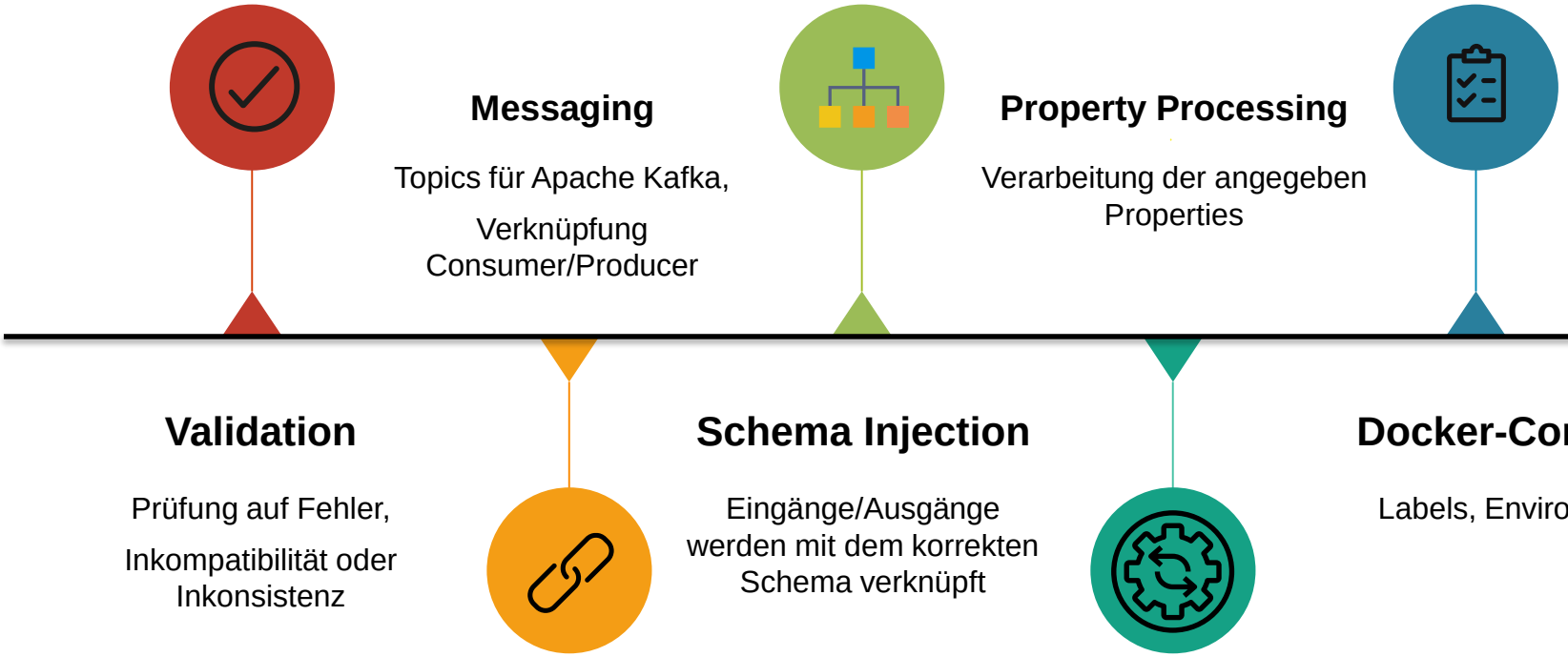


Konzept des Servicesystems
Logik Komponente

- Logik Komponente ist der **Mittelsmann**, der den im Engineering erstellten Plan umsetzt
- Logik Komponente besitzt **Zugriff** auf Datenspeicher und Nutzerverwaltung
  - Hierdurch sind sowohl **Rechte**, als auch **Services** eines bestimmten Anwenders bekannt
- Services können auf **unterschiedlicher Infrastruktur** gestartet werden
- Nach dem Start muss die Logik den **Status** der einzelnen Services überwachen und dem Engineering bereit stellen

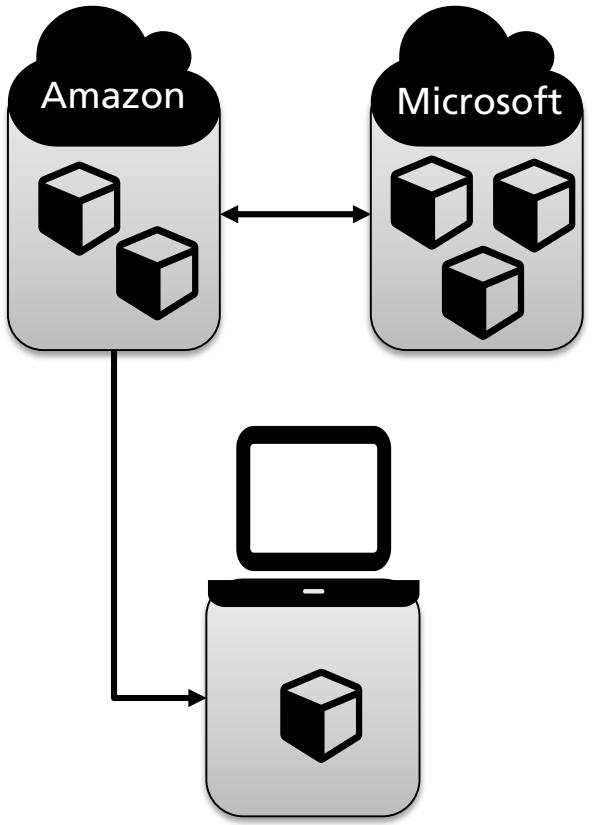


Pipeline



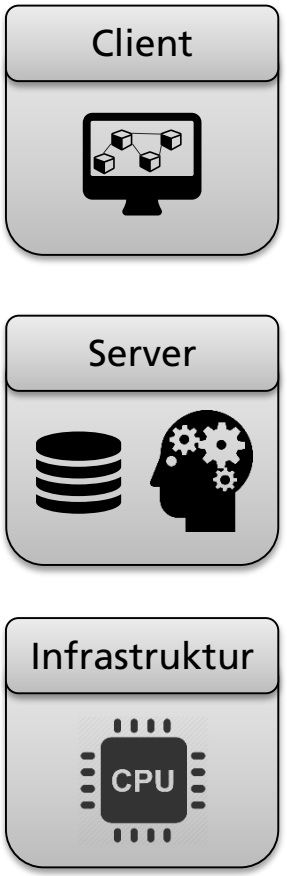
Konzept des Servicesystems
Ausführung Komponente

- Diese Komponente führt den gewünschten Baustein (Service) aus
- Ausführende Komponente soll **frei wählbar** sein
  - Public / Private Cloud
  - On Premise Cloud
  - Edge Device
  - Normaler Arbeits-PC
- Services, die auf unterschiedlicher Infrastruktur ausgeführt werden, sollen Informationen austauschen



Umsetzung
Aufbau

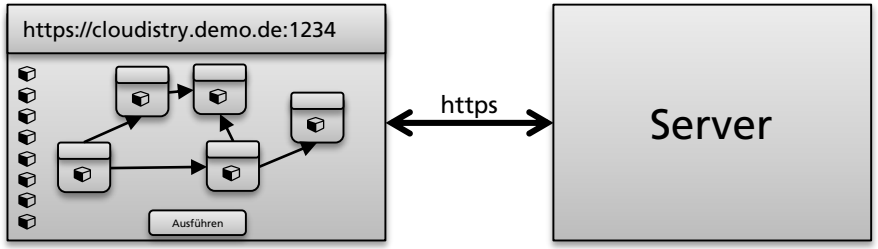
- **Client**
  - Visualisiert lediglich die Engineering-Komponente
  - Keine eigene Datenbank
- **Server**
  - Muss von einem der Nutzer aufgesetzt und betrieben werden
  - Enthält Nutzerdaten, Baukästen, Bausteine und Logik
- **Infrastruktur**
  - Führt Bausteine aus
  - Kann intern oder extern sein



Umsetzung
Client

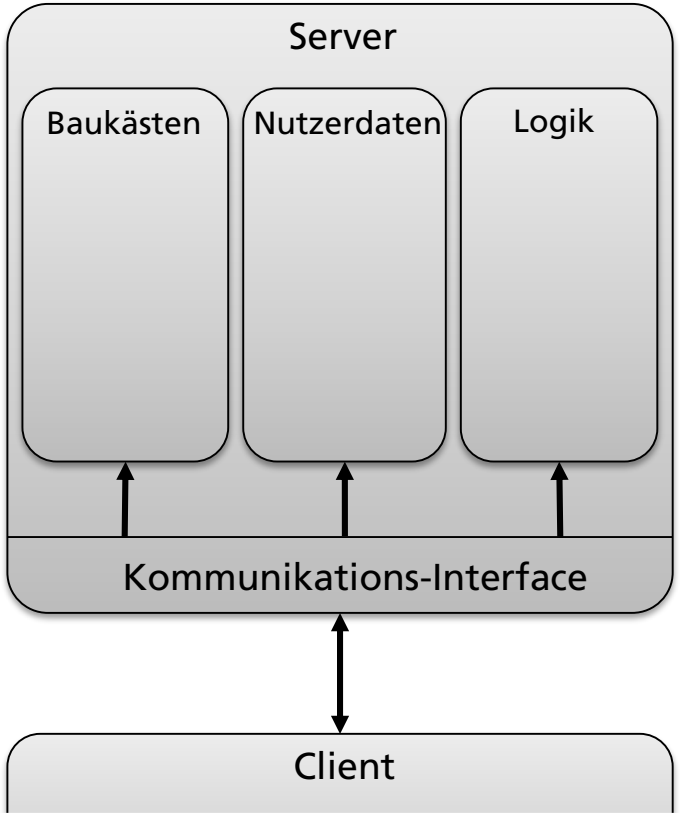
- Klassischer Web-Client mit HTML 5 / Java Script
- Ausführbar auf den meisten PCs unabhängig von deren Betriebssystem und installierter Software
- Anforderung: Internetbrowser mit Java Script Unterstützung
- Systemarchitektur
  - Visualisierungskomponente wird gefüllt mit Informationen vom Server
  - Kommunikation über gesicherte Verbindung

Quelle: www.w3.org



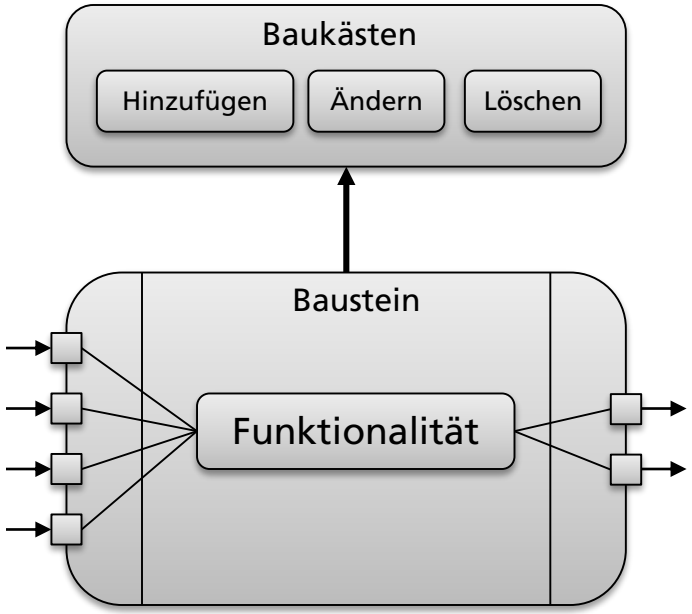
Umsetzung
Server

- Schnittstelle für Kommunikation mit dem Client leitet Informationen an drei verschiedene Komponenten weiter
  - Baukästen
    - Enthält alle Bausteine
    - Bietet Informationen zu einzelnen Bausteinen an
  - Nutzerdaten
    - Login-Daten für alle Nutzer
    - Account-Information für externe Infrastrukturanbieter
  - Logik
    - Verarbeitet erstellten Graph aus dem Engineering
    - Zugriff auf andere Komponenten



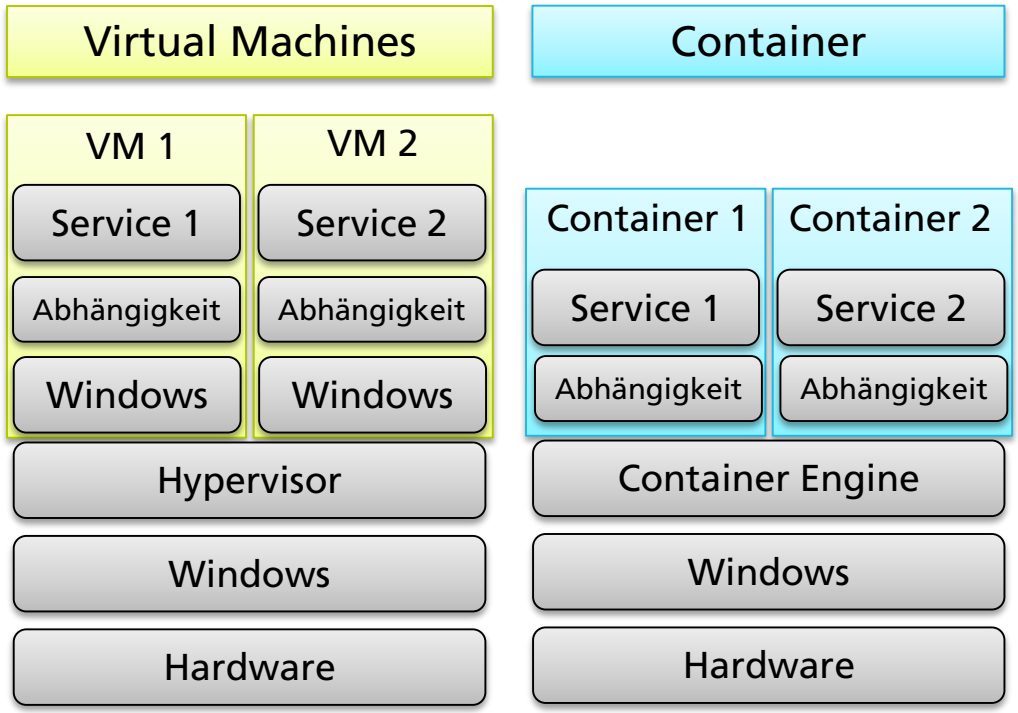
Umsetzung
Server - Baukästen

- Bietet Funktionalität zum Löschen, Ändern und Hinzufügen von Bausteinen
- Definiertes Schema für Bausteine nötig
  - **Funktionsbeschreibende** Informationen
    - Welche Funktionalität bietet der Baustein?
  - **Schnittstellenbeschreibende** Informationen
    - Welche Inputs und Outputs liefert der Baustein?
  - **Anforderungsbeschreibende** Informationen
    - Welche Hardware ist für den Baustein nötig?



Umsetzung
Einführung: Container - Abstraktionsschicht

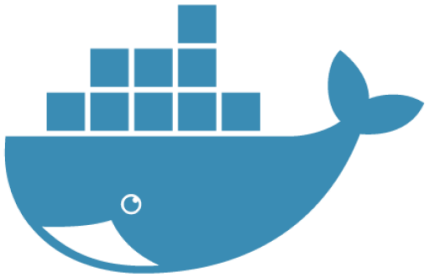
- Container sind **effizientere** Virtuelle Maschinen
- Sie nutzen das **native Betriebssystem** statt ein komplett isoliertes Betriebssystem darüber zu bauen
- Resultierende Vorteile
  - Abhängigkeiten können geteilt werden
  - Kein Startvorgang eines zweiten Betriebssystems nötig
  - Benötigt weniger Ressourcen



Umsetzung

Was muss ich als Bausteinlieferant machen?

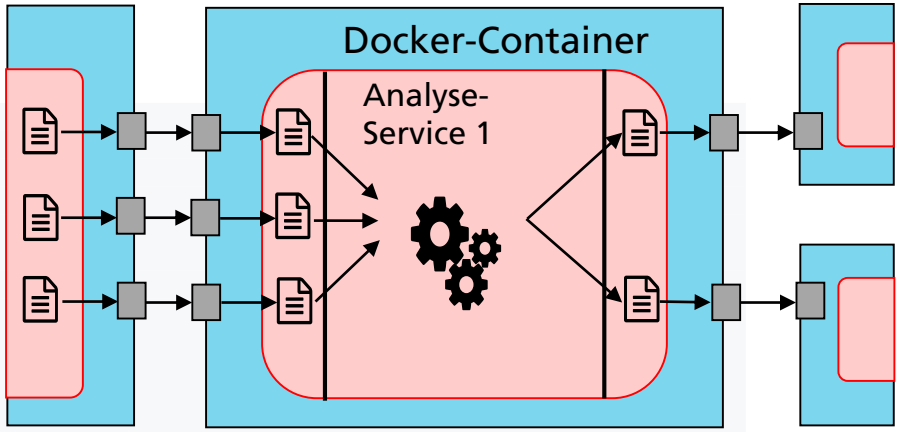
- Ein Docker-Container bündelt einen Baustein inklusive seiner Abhängigkeiten
- Hierfür müssen die Abhängigkeiten sowie die ausführbare Datei des Services definiert werden
- Um mit anderen Bausteinen kommunizieren zu können, müssen zusätzlich noch Ports freigegeben werden
- Alle genannten Informationen werden in einer „Dockerfile“ definiert
- Danach kann der Container mit Docker erstellt werden
- Der erstellte Container kann anschließend per Docker gestartet werden



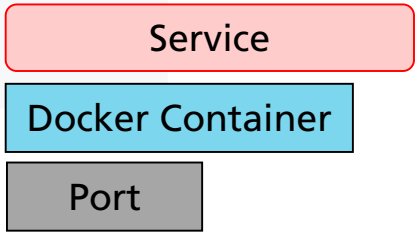
https://www.docker.com

Umsetzung
Services

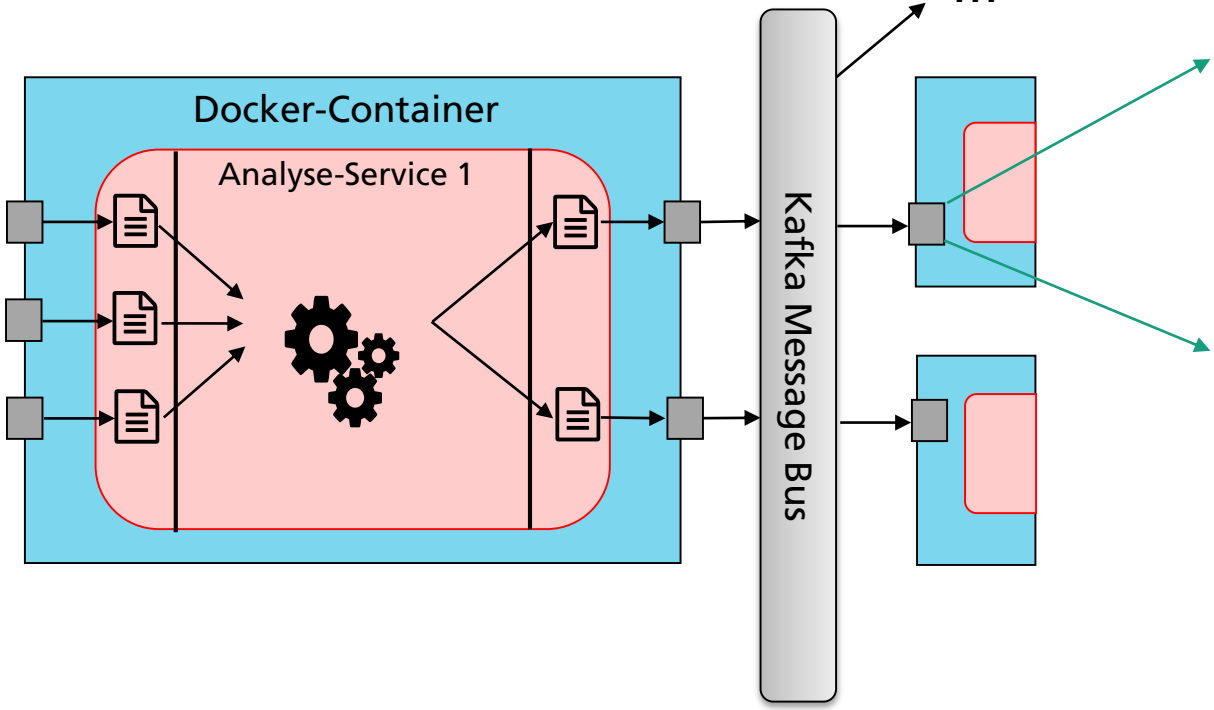
```
Service:
    Metadata:
        Title:
        Description:
        Author:
    Image:
        Registry: (optional)
        Name:
        Tag: (optional)
    Properties:
        - Property:
            Name: (eindeutig)
            Description: (optional)
            DefaultValue
            Required
            ReadOnly
            Type: (optional, [int(64), float(64), string (default), boolean, port])
    Input:
        - Port:
    Output:
        - Port:

Port:
    Id:
    Schema-Id:
    Schema-Version: (optional)
```



Legende:



Umsetzung
Services Kommunikation



Avro Schema

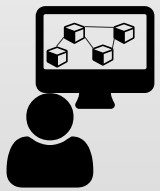
```
{
  "type": "record",
  "name": "temp",
  "doc": "This record represents a temp",
  "namespace": "de.isw.unistuttgart",
  "fields": [
    {
      "name": "tmp",
      "type": "float"
    },
    {
      "name": "timestamp",
      "type": "float"
    }
  ]
}
```

- Services legen ihr Schema in einem Registry ab

➔ Typsicherheit gegeben. Generisch und Flexibel

Umsetzung
Zusammenfassung

Engineering



- Planung und Start von Baustein-landschaft
- Browserbasiert mit Backend zur Datenhaltung
- Client überall ausführbar

Logik

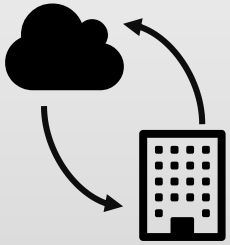
1) 

2) 

3) 

- Interpretation von Baustein-landschaft
- Zugriff auf Nutzerdaten und Bausteine
- Übergibt Bausteine an die verschiedenen Infrastrukturen

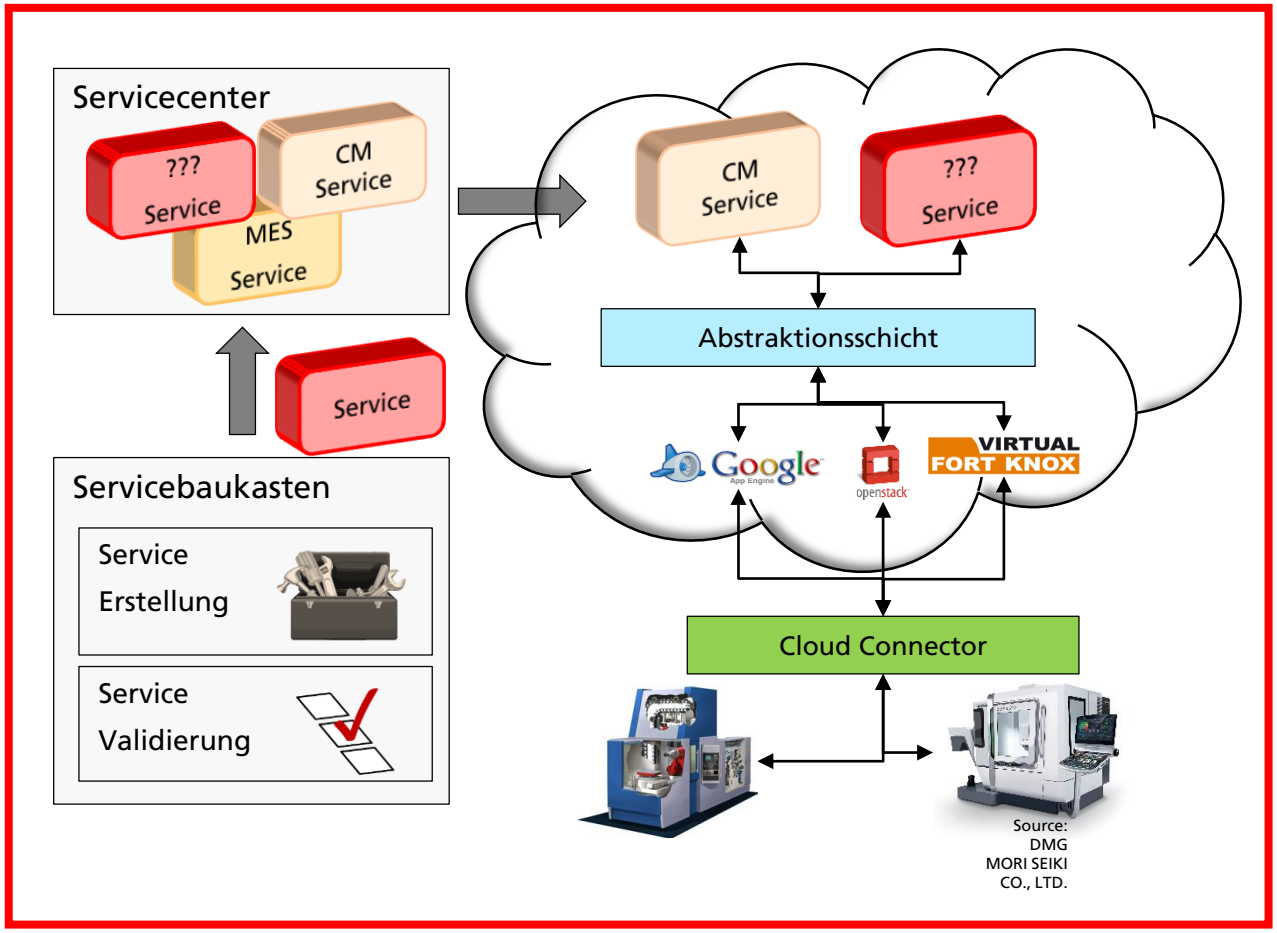
Ausführung



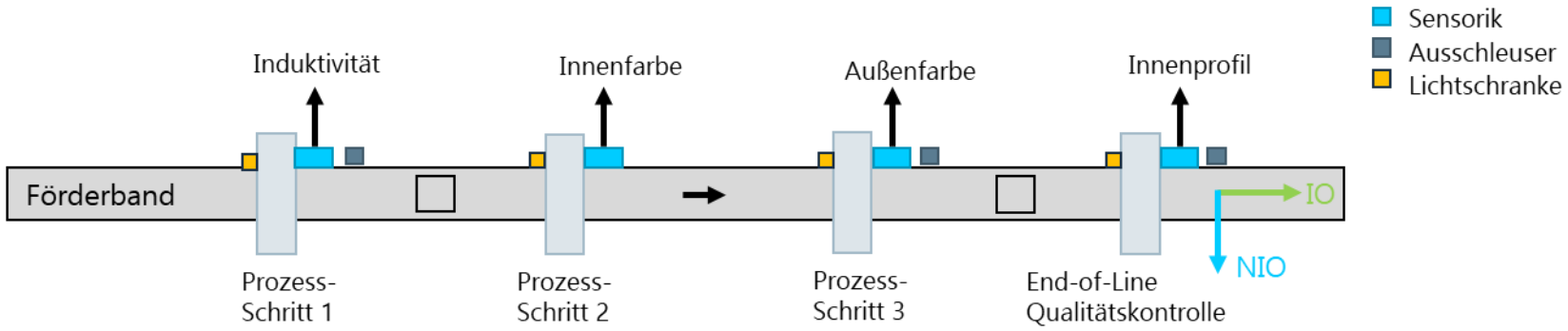
- Ausführung auf interner oder externer Infrastruktur
- Bausteine in Container zu verpacken, ermöglicht Plattformunab-hängigkeit

Zielvorstellung

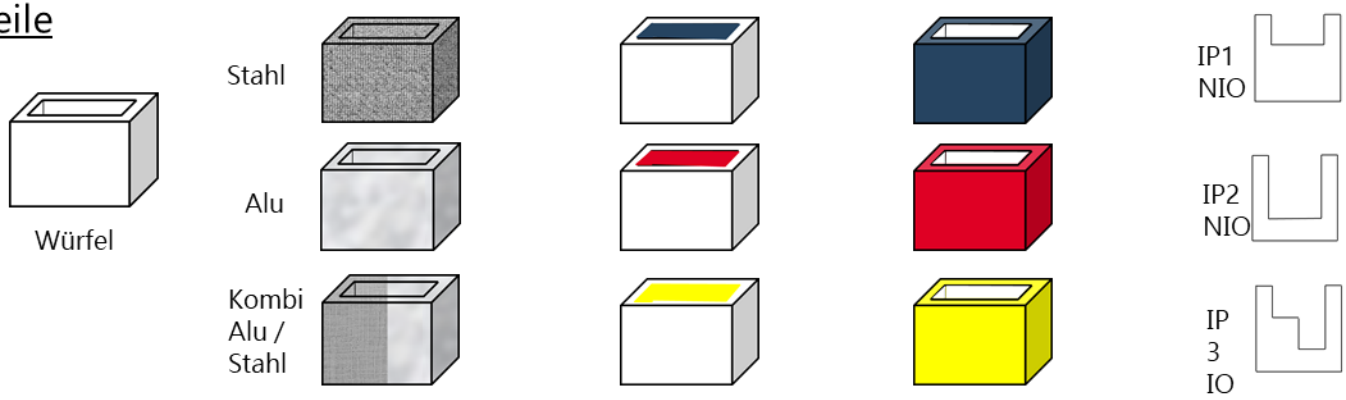
Erste Ergebnisse anhand der Gesamtarchitektur



Demonstrator
Aufbau



Bauteile

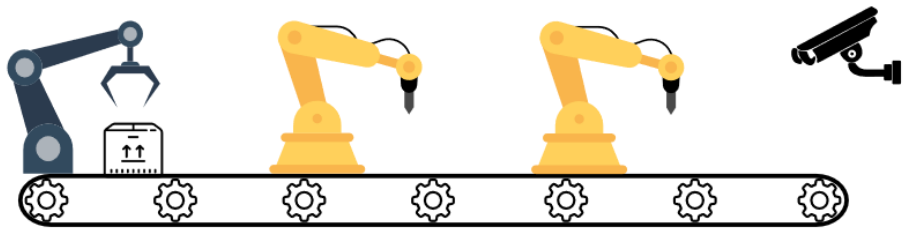


Demonstrator
Web-Ui

Live Analysis Dashboard

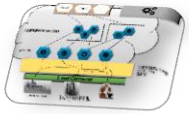
ISW Rexroth Bosch Group

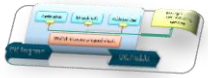
Event	Evidence	Action
P==IP1	FA==R&FI==R&M==StSt	Discharge workpiece
P==IP1	FA==R&FI==R&M==AlAl	Alarm



Demonstrator

Inhalt





- Ausgangssituation
- Zielvorstellung
- Erste Ergebnisse
- Ausblick

Ausblick

- Erweiterung des Cloud-Connectors um weitere Konnektoren
- Konzeption von Services
  - Energieverbrauchs-Service
  - Energiemanagement-Service
  - Optimierung der Produktion auf Basis von Produktions- und Logistikdaten
- Integration ins Produktionsumfeld

Ihr Ansprechpartner am ISW

Wir steuern Zukunft

Innovativ. Interdisziplinär. Wissenschaftlich.

Vielen Dank!

Zeit für Fragen

Timur Tasci, M.Sc.
Wissenschaftlicher Mitarbeiter
Software- und Engineeringmethoden

Telefon: +49 711 685-82433
E-Mail: timur.tasci@isw.uni-stuttgart.de
Web: www.isw.uni-stuttgart.de