

Microservices-based Business Process Model Automation

Agnes Koschmider



Overview

- What are Microservices?
- Microservices vs. Monolith
- Microservices vs. SOA
- Microservices Framework + BPM
- Challenges of Microservices-BP Automation

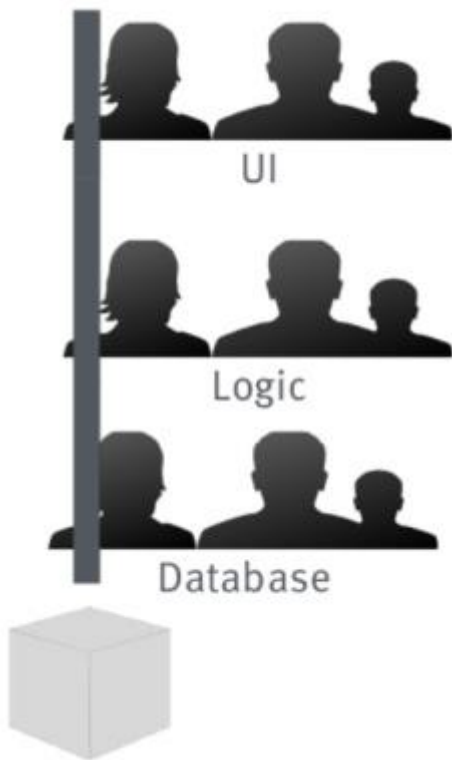
What is a Microservice?

The term "Microservice Architecture" has sprung up over the last few years to describe a particular way of designing software applications as suites of independently deployable services. While there is no precise definition of this architectural style, there are certain common characteristics around organization around business capability, automated deployment, intelligence in the endpoints, and decentralized control of languages and data.

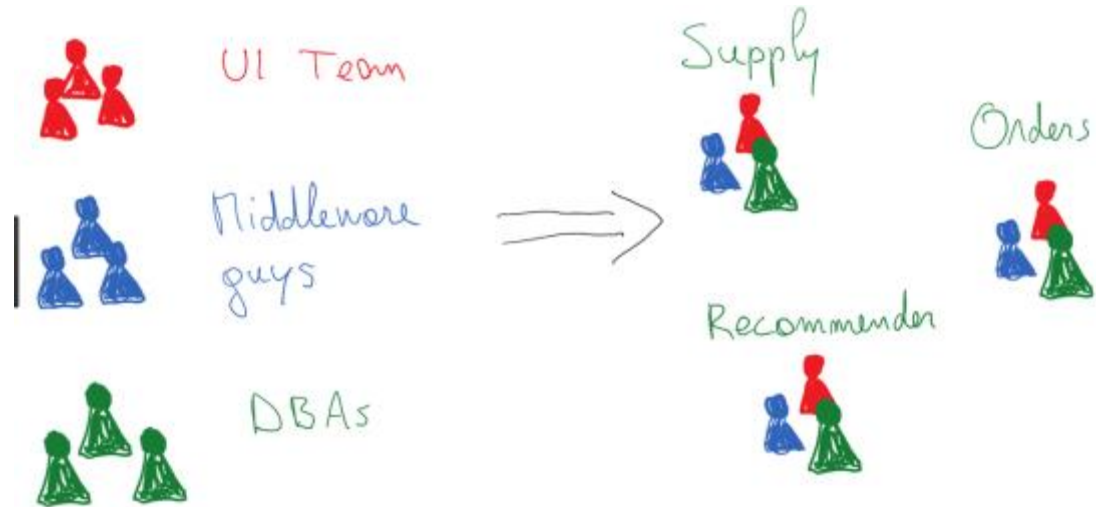
Martin Fowler
(<https://martinfowler.com/>)

Characteristics of Microservices

■ Organizational Dimension



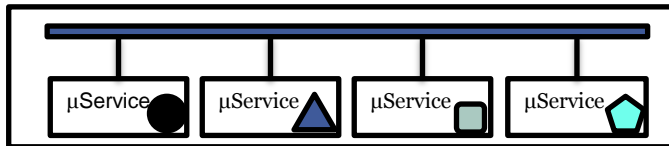
Classical Organization



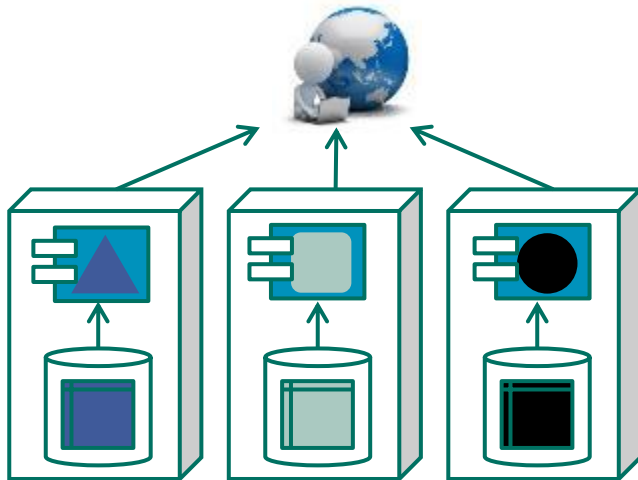
Team setup by business functions

Characteristics of Microservices

■ Technical Dimension

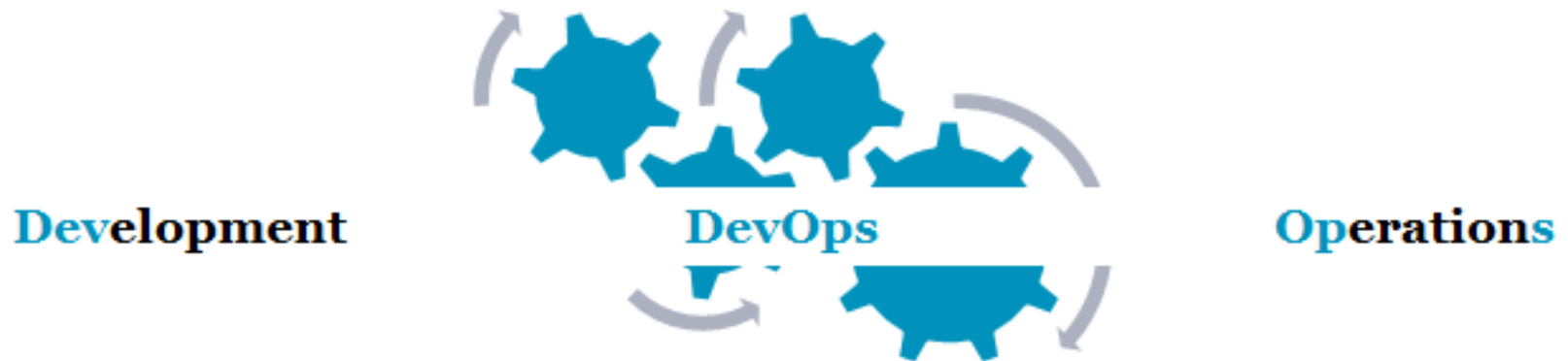


Smart endpoints, dumb pipes:
(i.e., only the endpoints are smart and pipes only forward information)

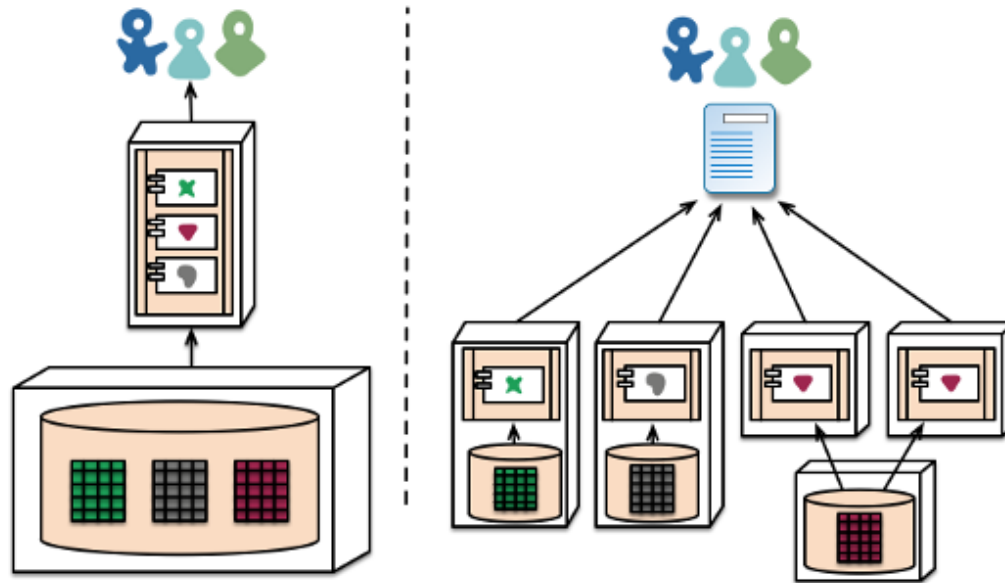


Decentralized Governance/
Decentralized Data Management

Microservices target Products and not Projects







Microservices vs. Monolith



- Several modules in one process
- comprise the whole technology stack
- technology independent

Microservices vs. SOA

-  ■ A microservice comprise the **whole technology stack**
 - By packaging the whole technology stack (data, business, UI) a microservice is easily deployable since all inter-service dependencies are shipped and deployed at the same time.
-  ■ **Technology independent to other microservices**
 - Each team responsible for developing and maintaining a microservice can choose the technology stack suitable for their teams. The team is independent of choices made by other teams. Only the CI infrastructure has influences on the technologies.
-  ■ **No heavyweight messaging (no ESB), simple protocols (REST)**
 - The microservice architecture follows the principle smart endpoint and dumb point. Meaning, that a microservice knows exactly with which other microservice it needs to communicate with.
-  ■ **Deployable individually, testable (better support of DevOps)**
 - Due to the independency to other services and loose coupling, each microservice can be deployed on its own. Same holds for testing the functionality because a microservice is self contained.

 No statement in SOA

 Oppositional Statement

Microservices Framework

Container



Docker as most commonly used virtualization mechanism for finegrained Microservices (one container per microservice)

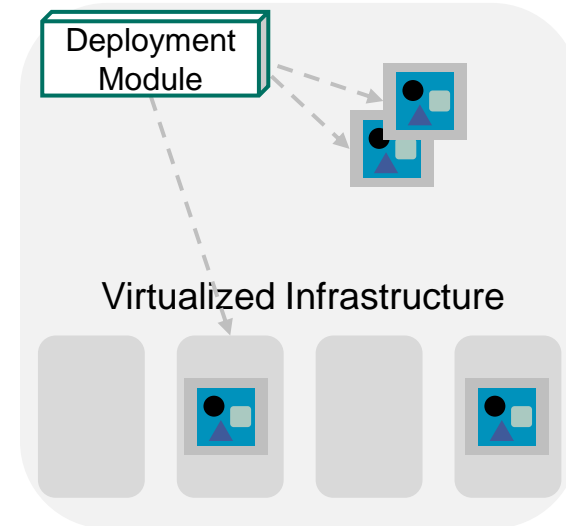
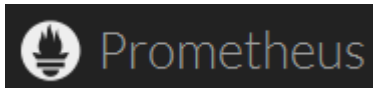
Orchestrator



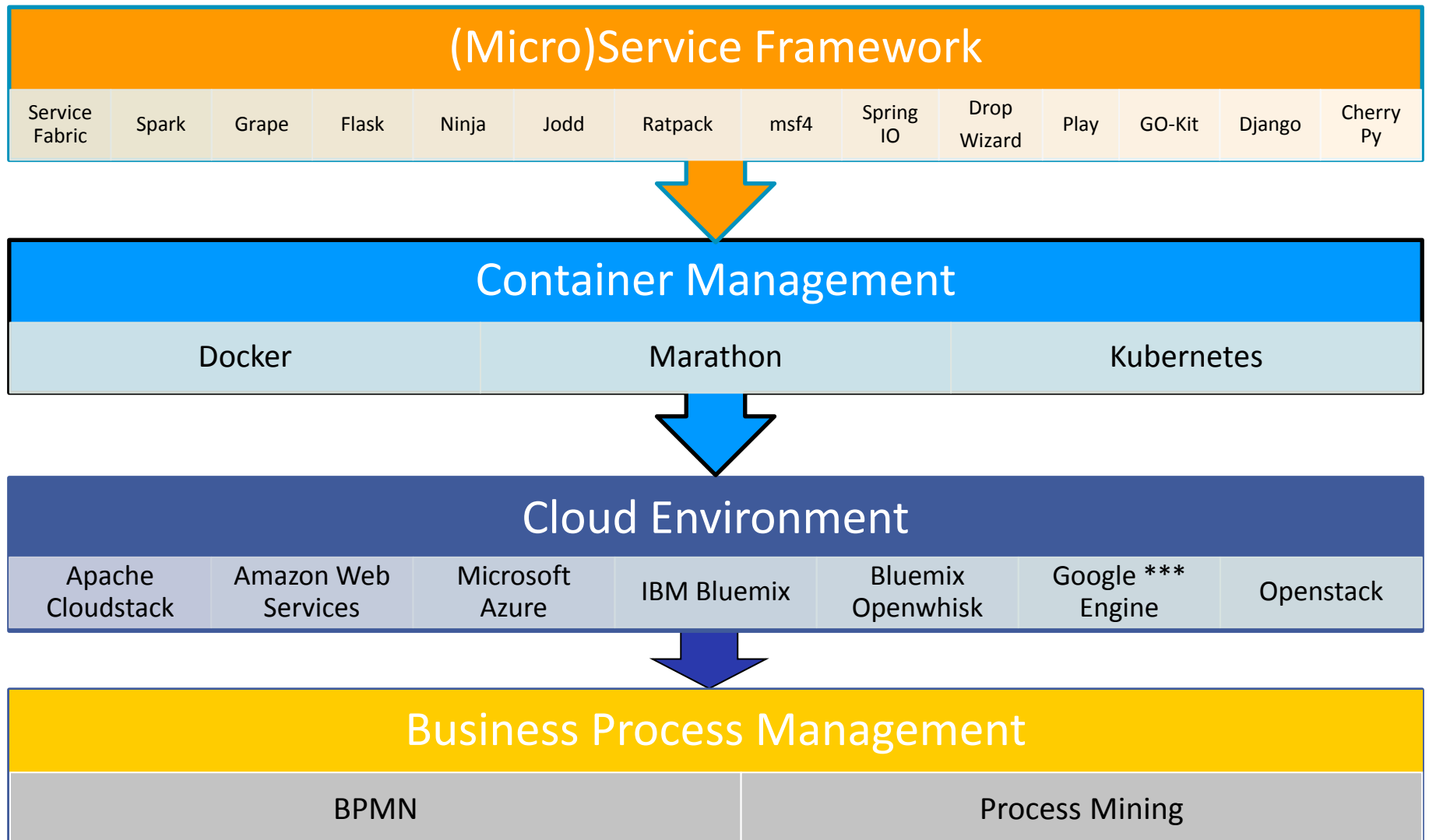
Orchestration and Replication tools available such as Kubernetes

Monitoring

Performance Monitoring

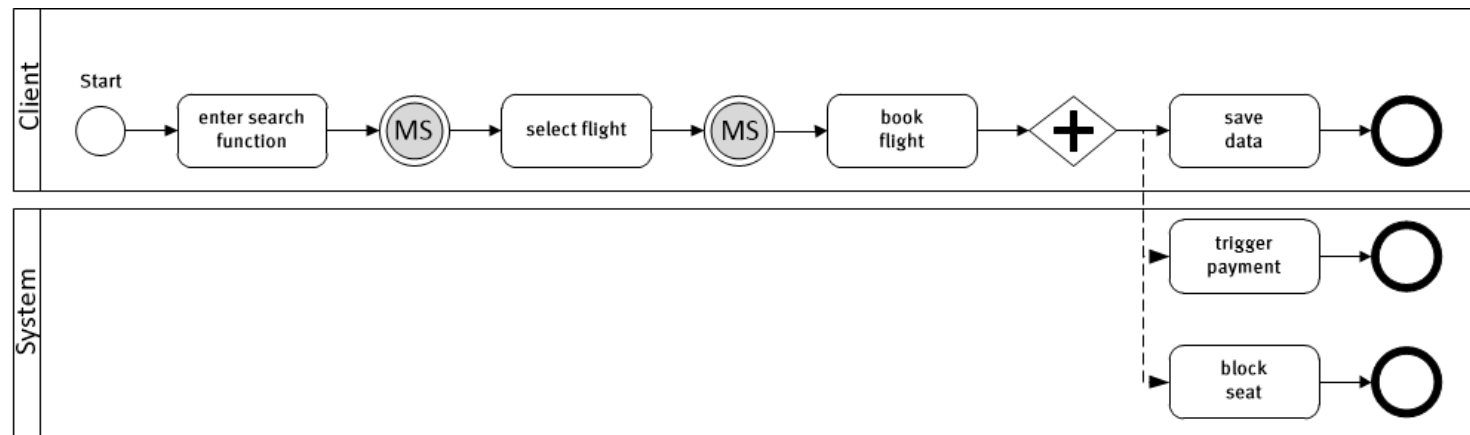


Microservices Technologies

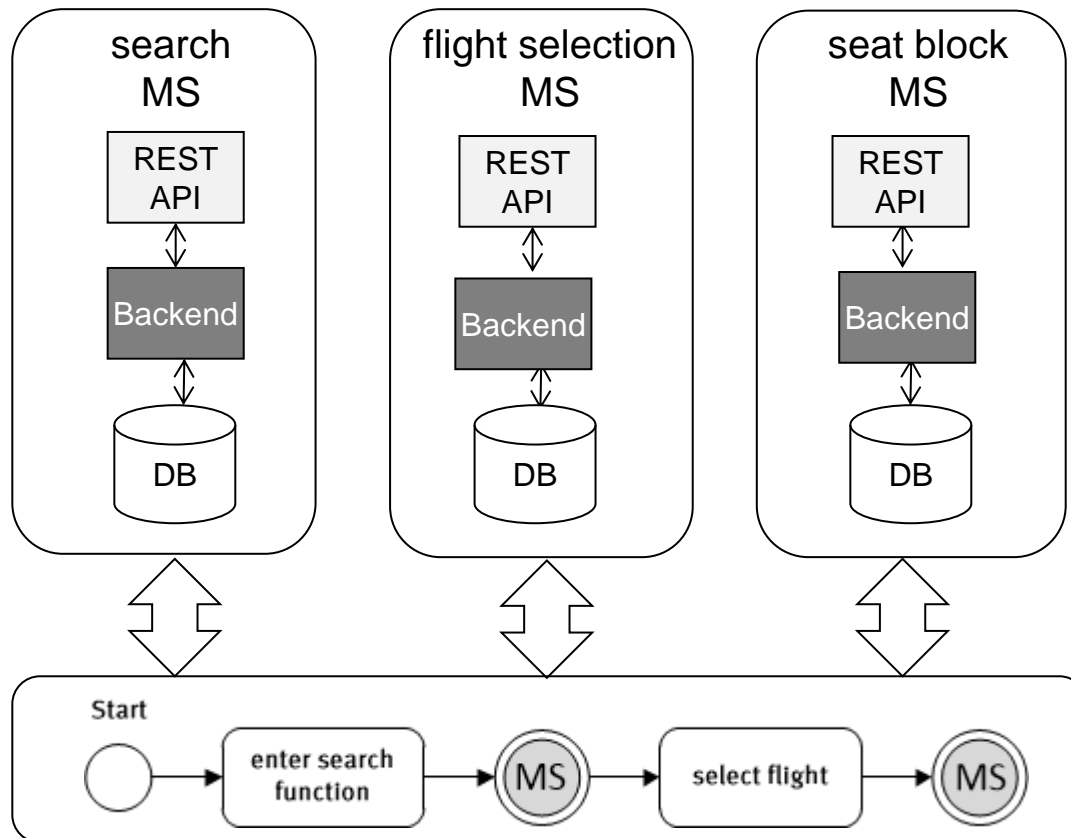


Microservices-based BP Automation

- **Challenge:** *What is an appropriate visualization and assignment of microservices in a business process model?* It remains to investigate if microservices are specified as events or as a new type of process activities (then an icon is attached to the activity) or if a process model fragment defines a microservice.



Microservices-based BP Automation



Microservices-based BP Automation

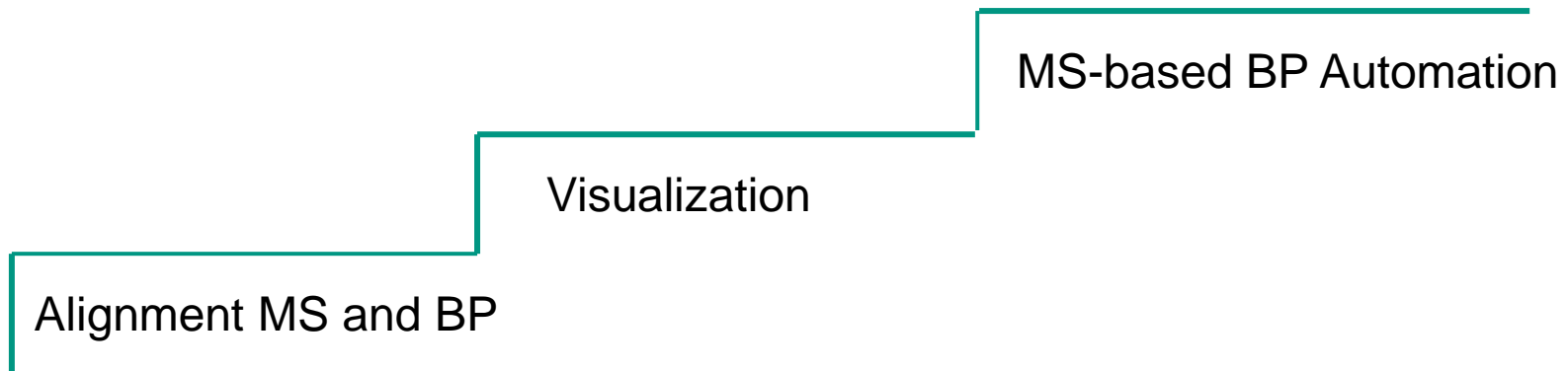
Challenge: *What are appropriate strategies for business process model orchestration (e.g., routing of business process activities). The control-flow of process models could make the unpredictable flow of microservices instances more transparent.*

Challenge: *How to deal with processes containing “automatic tasks” and “semi-automatic tasks” and thus how to combine process-based engines and containerized application orchestrator?*

Microservices-based BP Automation

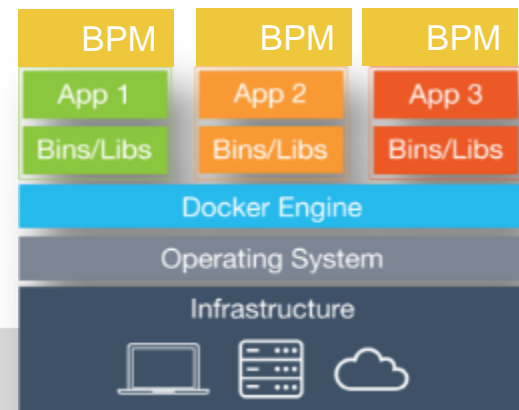
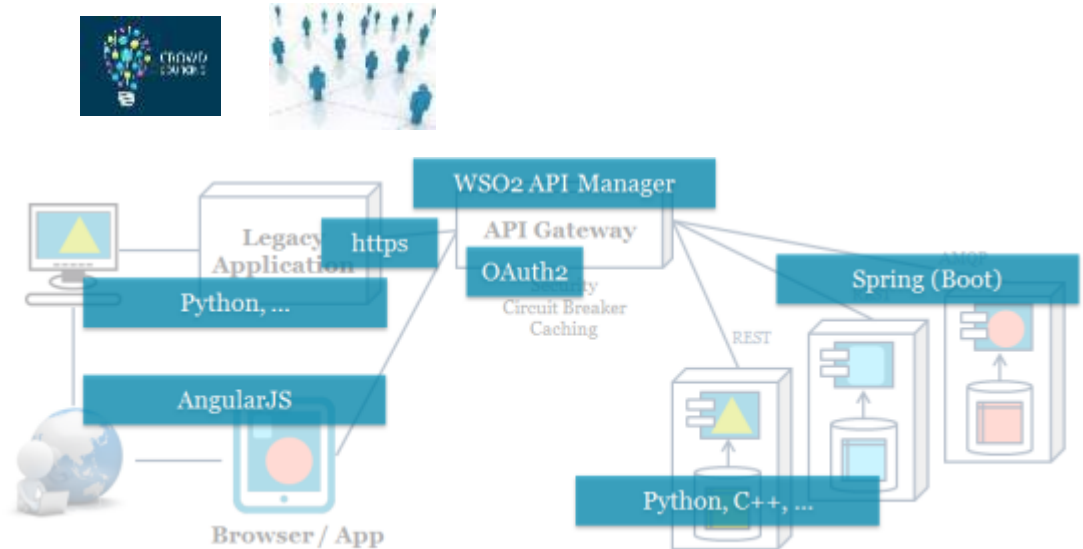
- **Challenge:** *How to specify the dependencies of microservices executed within distributed business process models?* Microservices require only a minimum of central control, which also raises the challenge of
- **Challenge:** *Who takes care of the process?* Microservices communicate with lightweight mechanisms but have so called “dumb pipes“ raising the challenge of:
- **Challenge:** *How to implement processes according to „smart endpoints and dumb pipes“ characteristic? Does „smart endpoints and dumb pipes“ allow business process improvement?*

Roadmap



- Do you want to contribute
→ Contact me 😊

Szenario: On-demand Services for Autonomous Car Sharing



*“If you can’t build a monolith,
what makes you think microservices are the
answer?”*