

FIRST STEPS TOWARDS A MICROSERVICE ARCHITECTURE FOR VIRTUAL POWER PLANTS

Manuel Wickert, Sven Liebehentze, Prof Dr. Albert Zündorf

Microservices 2019 - Dortmund



AGENDA

- WHY ?
- HOW ?
- LEARNINGS ?

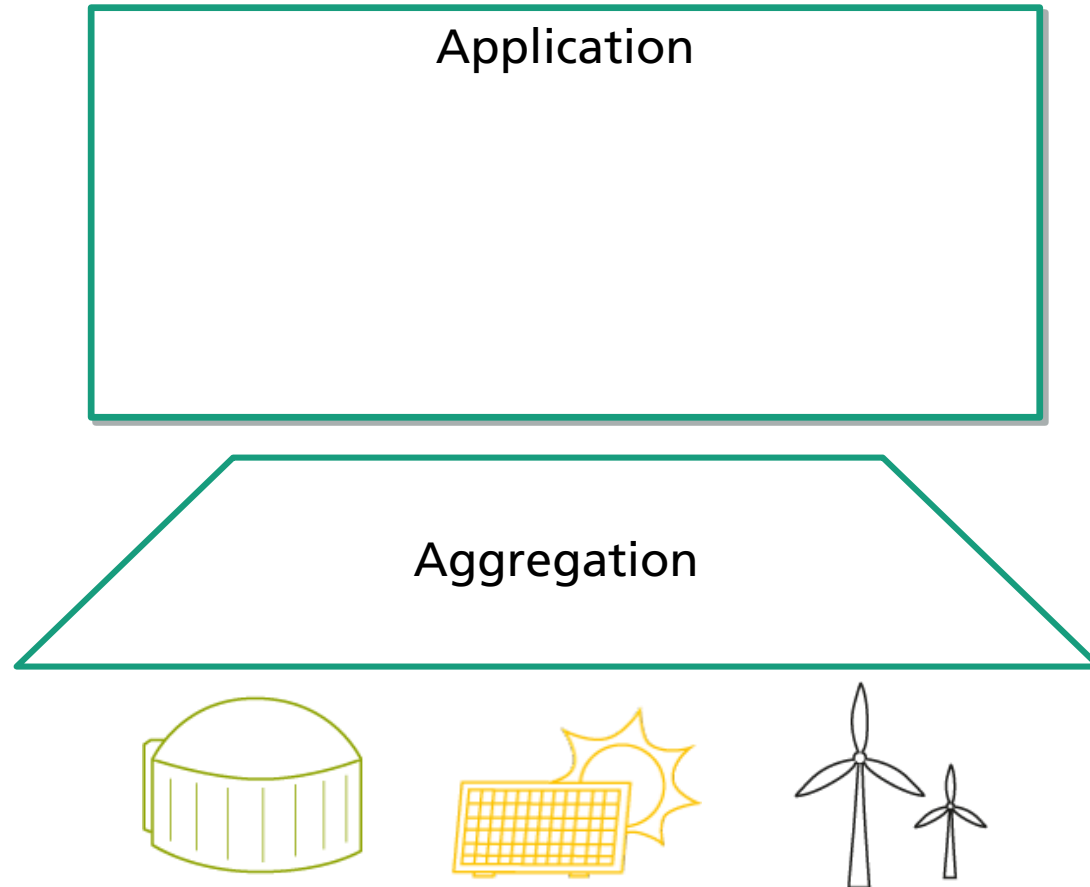
WHY?

WHY? - DOMAIN

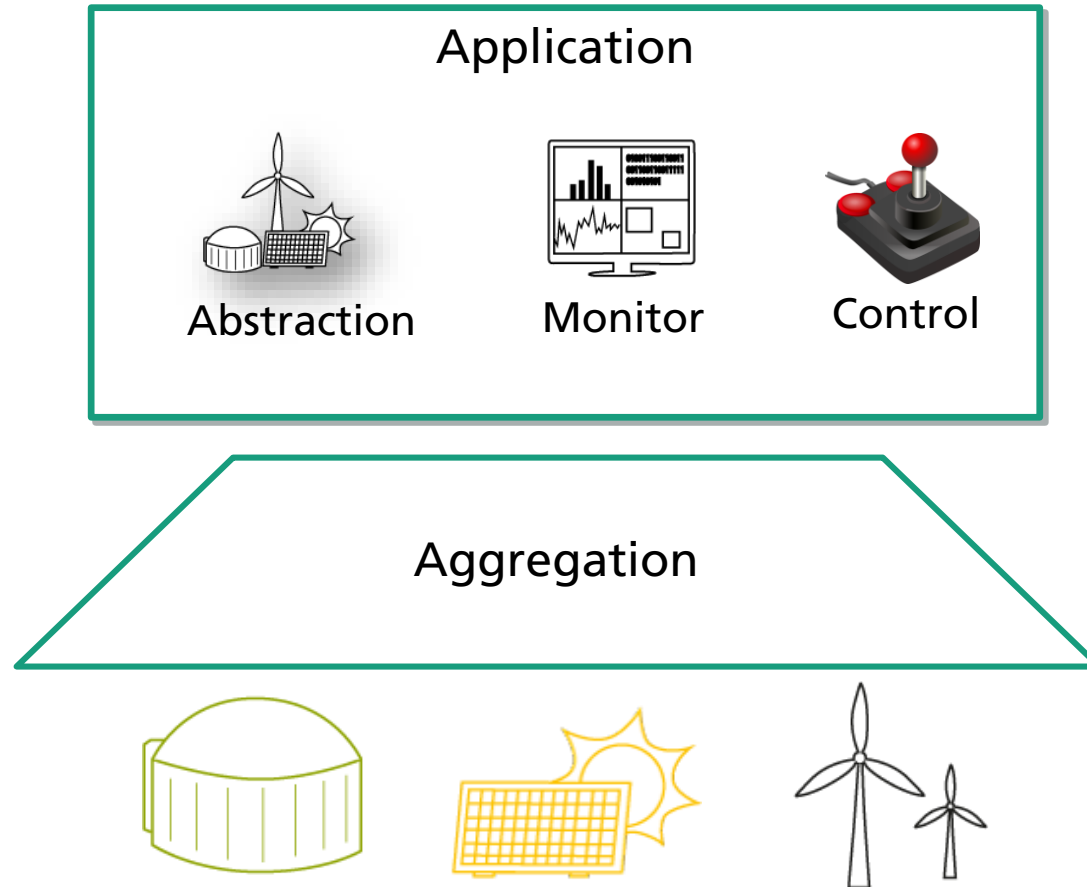


Application

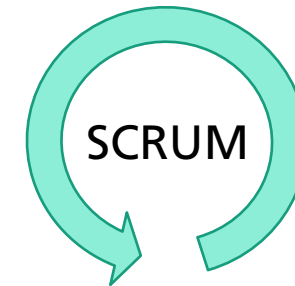
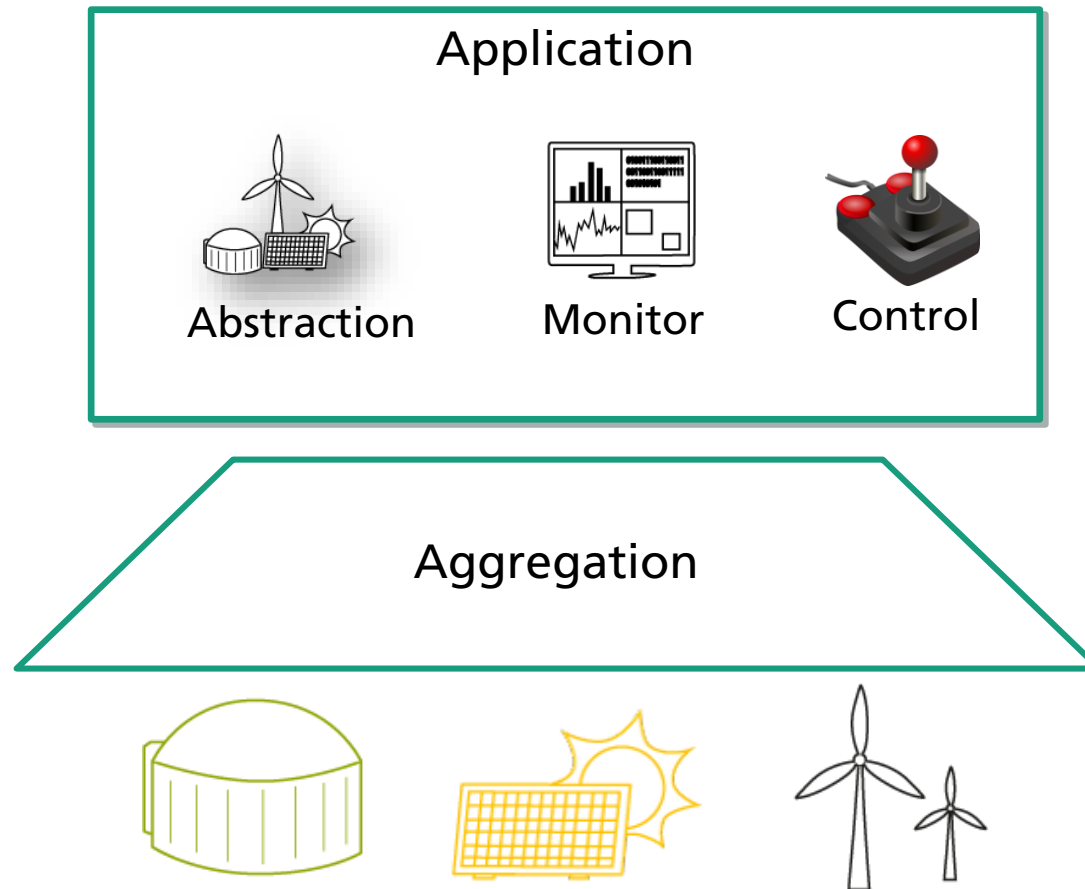
WHY? - DOMAIN



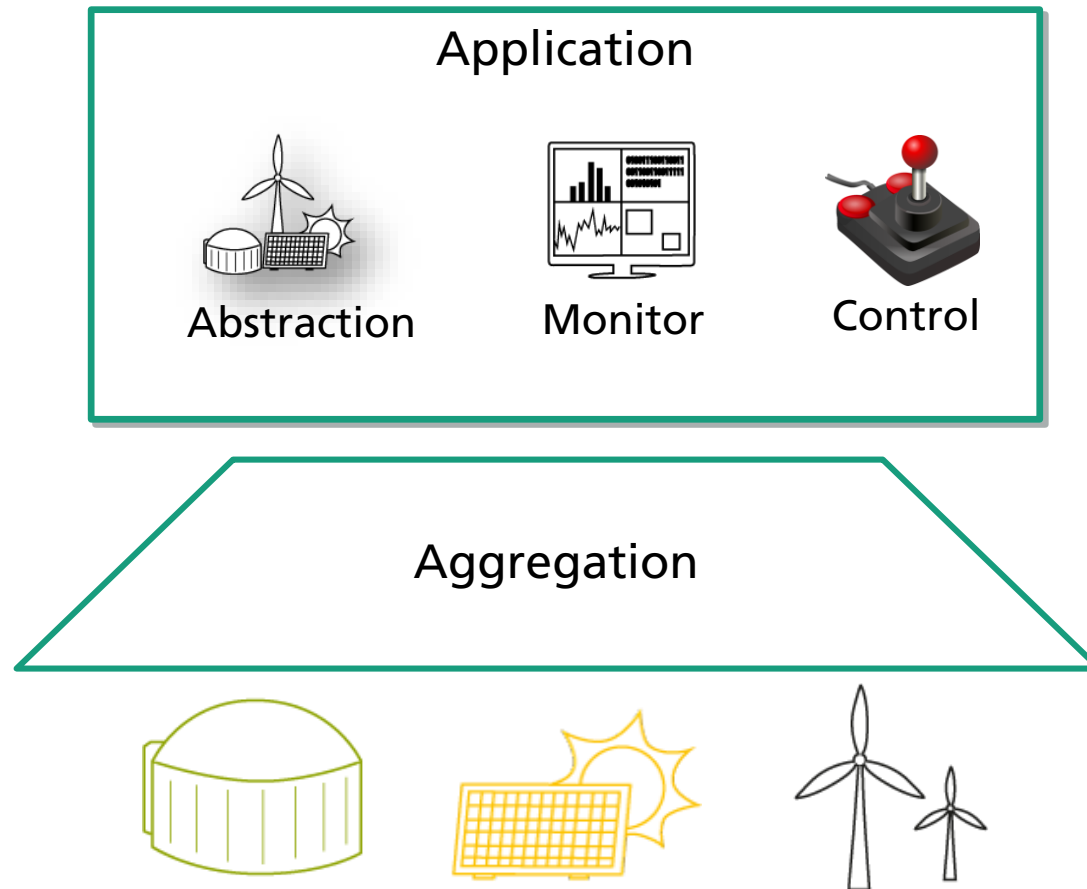
WHY? - DOMAIN



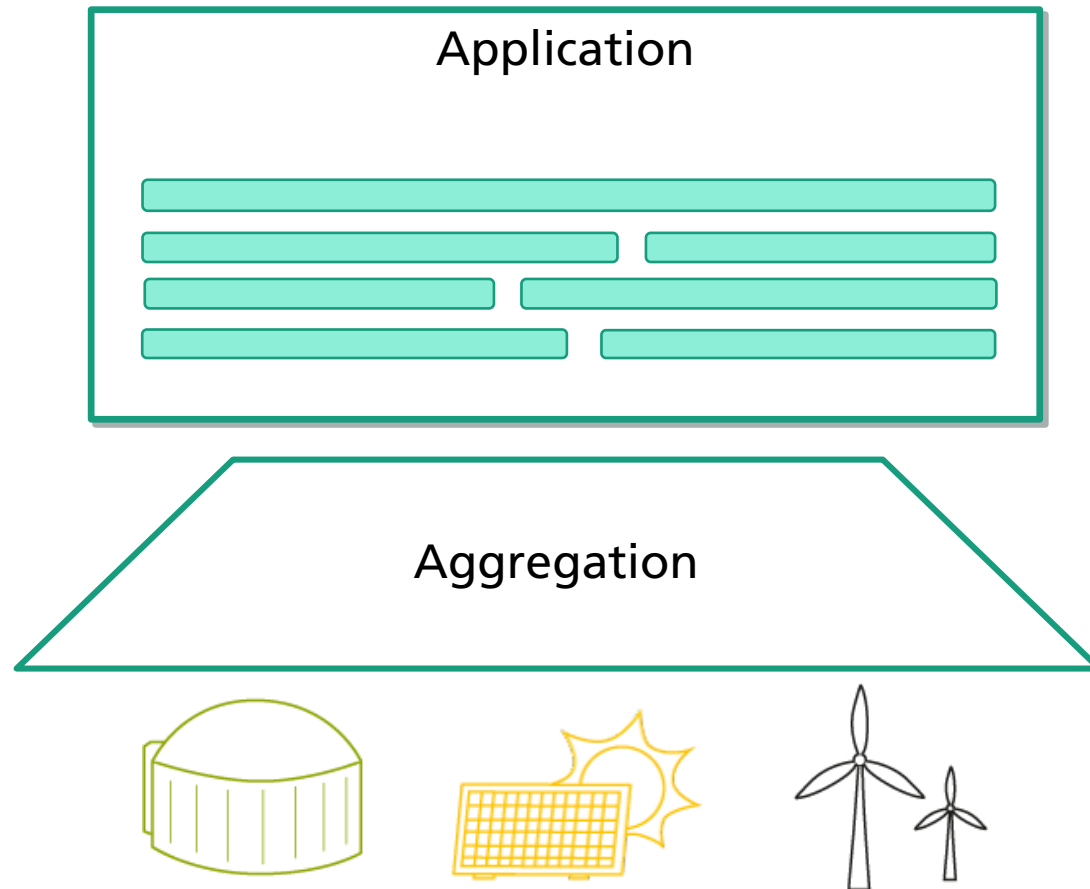
WHY? - TEAM



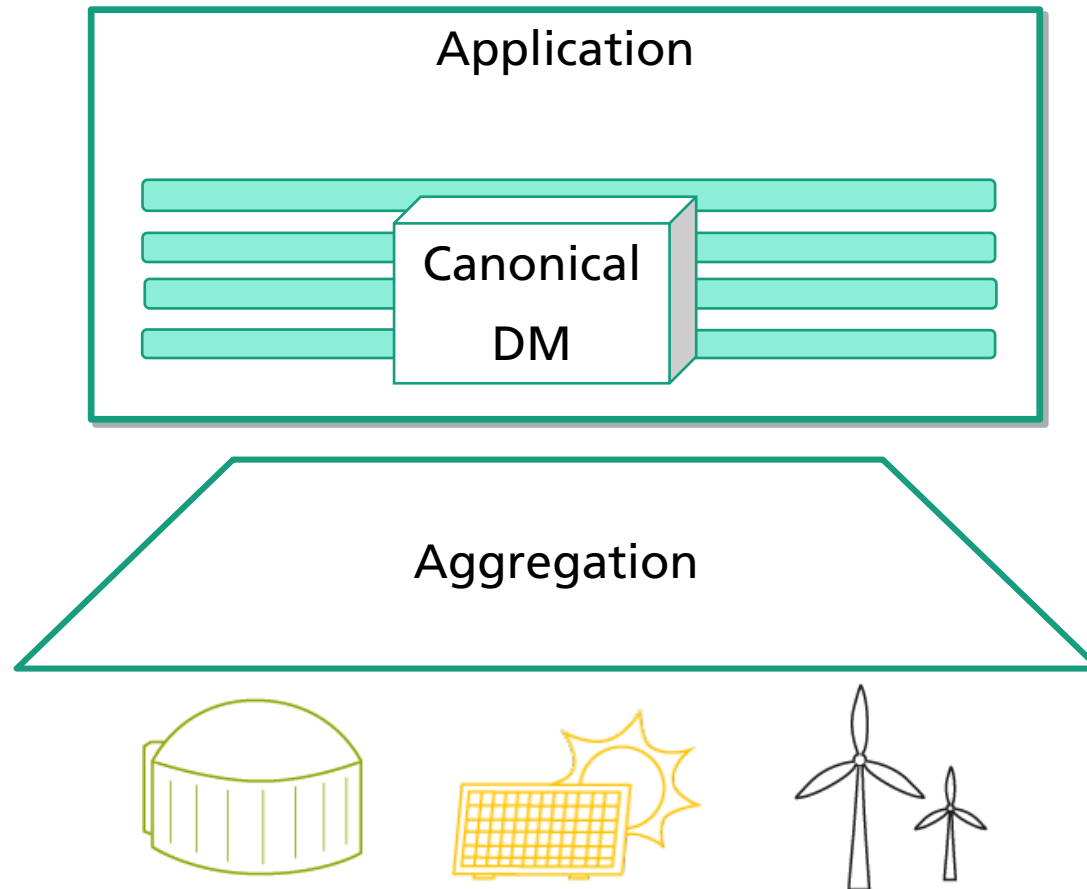
WHY? - TEAM



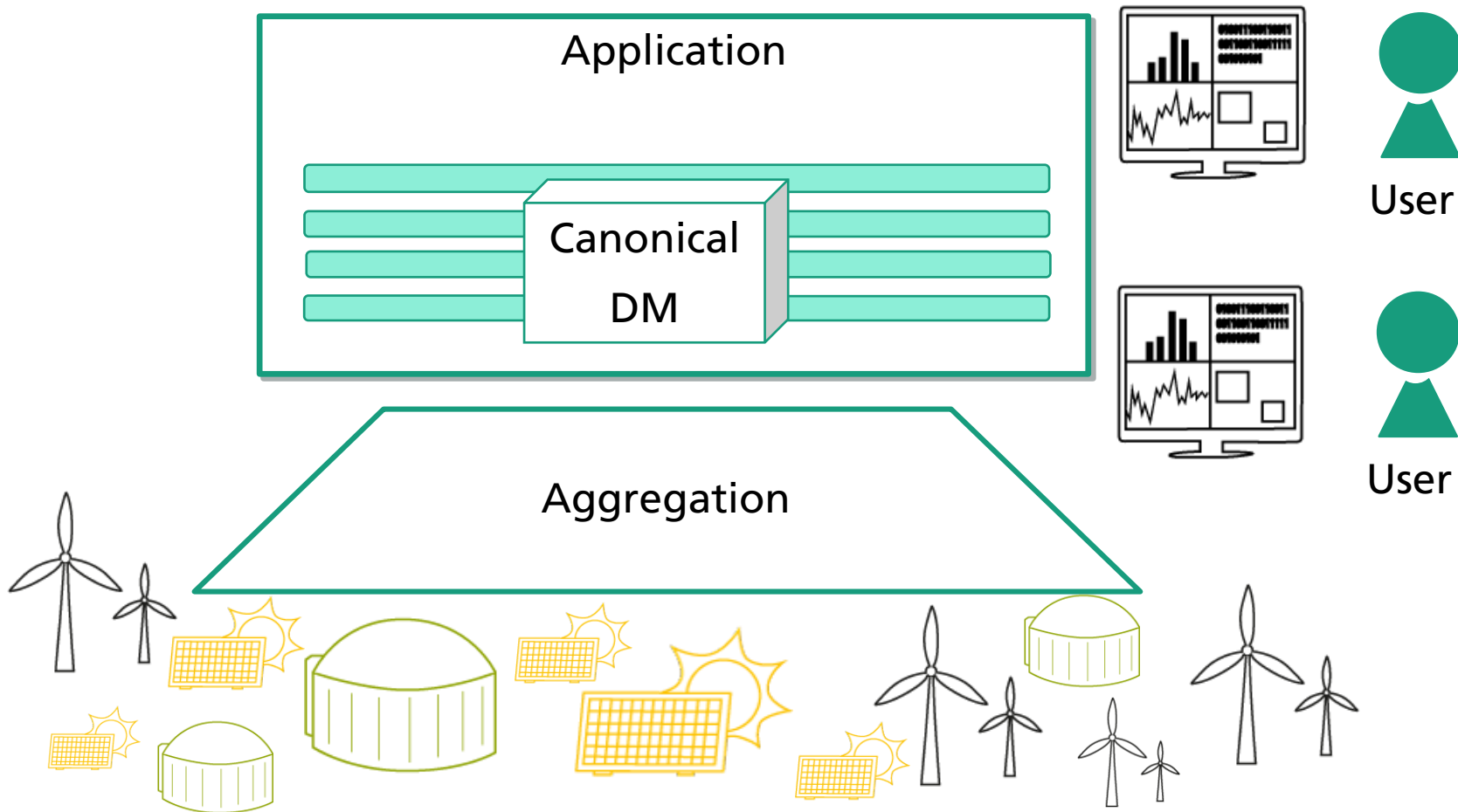
WHY? – ARCHITECTURE



WHY? – ARCHITECTURE



WHY? – SCALABILITY



REGULATORIC ENVIROMENT

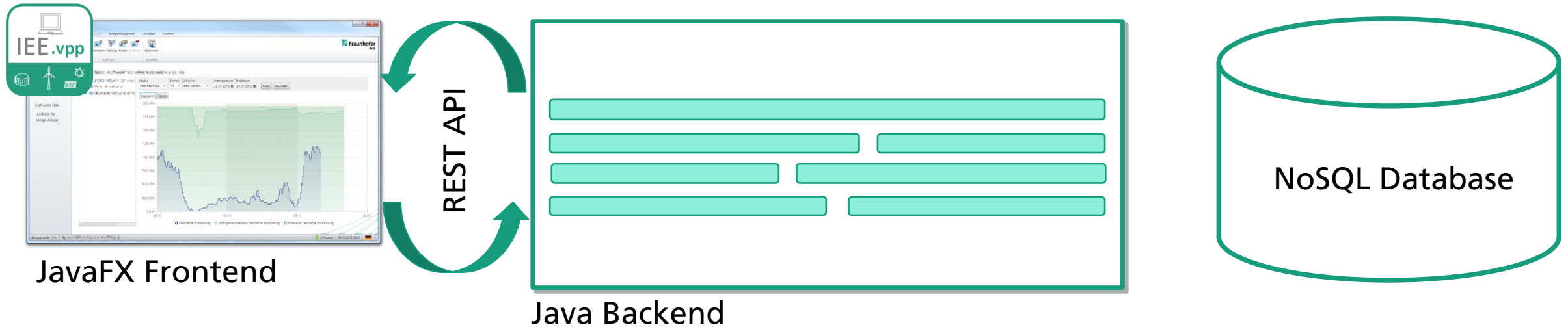
- Laws and ordinances
- Technical guidelines
- Contracts / general Agreements with grid operators
- Contracts / general Agreements with energy markets

SO WHY ?

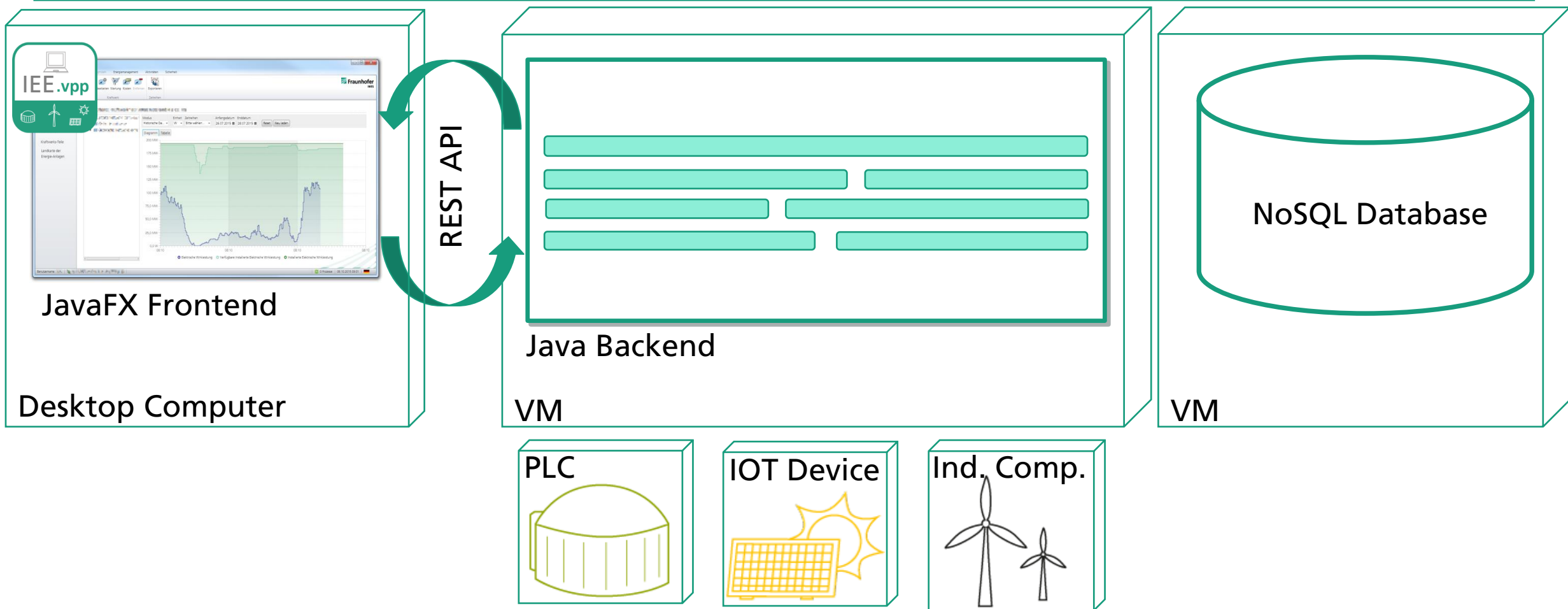
- Teams with different programming language background → Implement parts in Python
- Canonical data model inflexible in some situations → Use DDD
- Scalability becomes more important → Build scalable architecture
- Fast changing Regulatoric environment → Avoid hard changing software
- Migrating to a microservices architecture

HOW?

SOFTWARE SYSTEM



SOFTWARE SYSTEM



START WITH MICROSERVICE

Please Install:

- Docker, Kubernetes
- API Gateway (e.g. Kong)
- Authentication (e.g. Keycloak)
- Logging Server (Elastic Search)
- Tracing Server
-



Product Owner



Customer

START WITH MICROSERVICE

Please Install:

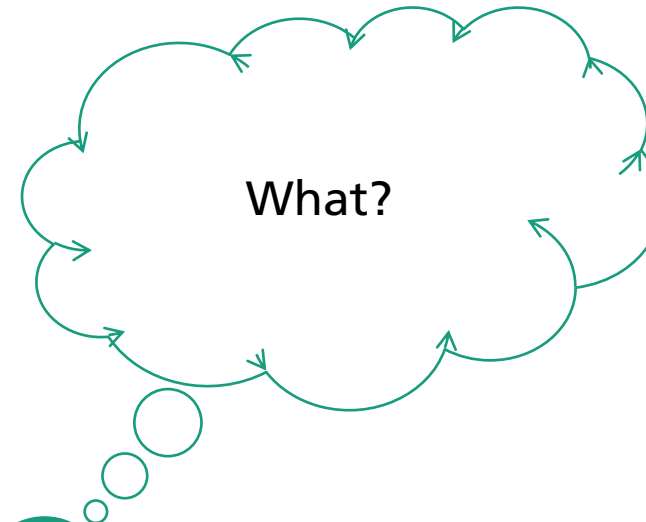
- Docker, Kubernetes
- API Gateway (e.g. Kong)
- Authentication (e.g. Keycloak)
- Logging Server (Elastic Search)
- Tracing Server
-



Product Owner



Customer



START WITH MICROSERVICE

Please

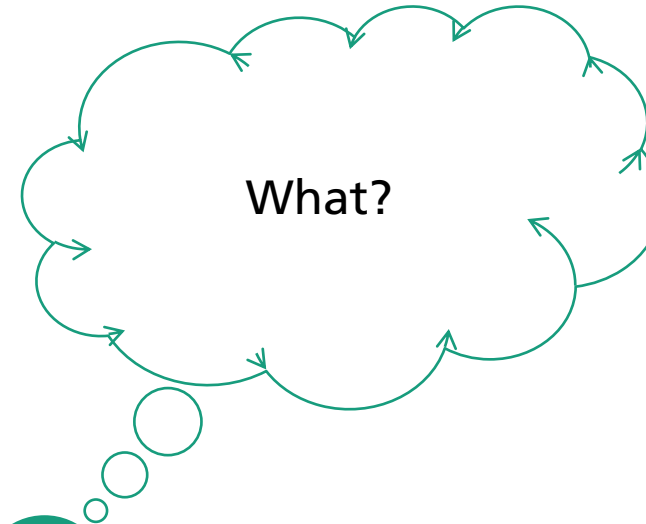
- D
- API Gateway (e.g. Kong)
- Authentication (e.g. Keycloak)
- Logging Service (Elastic Search)
- Tracing Server
-



Product Owner



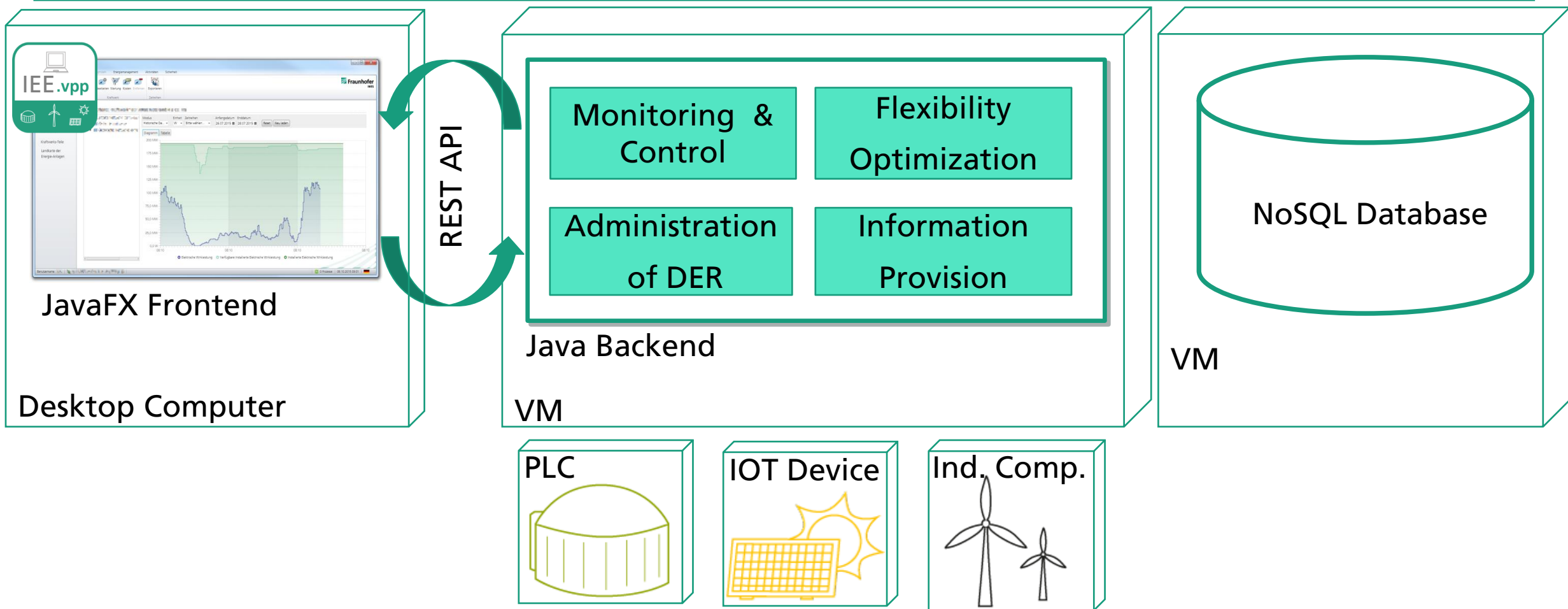
Customer



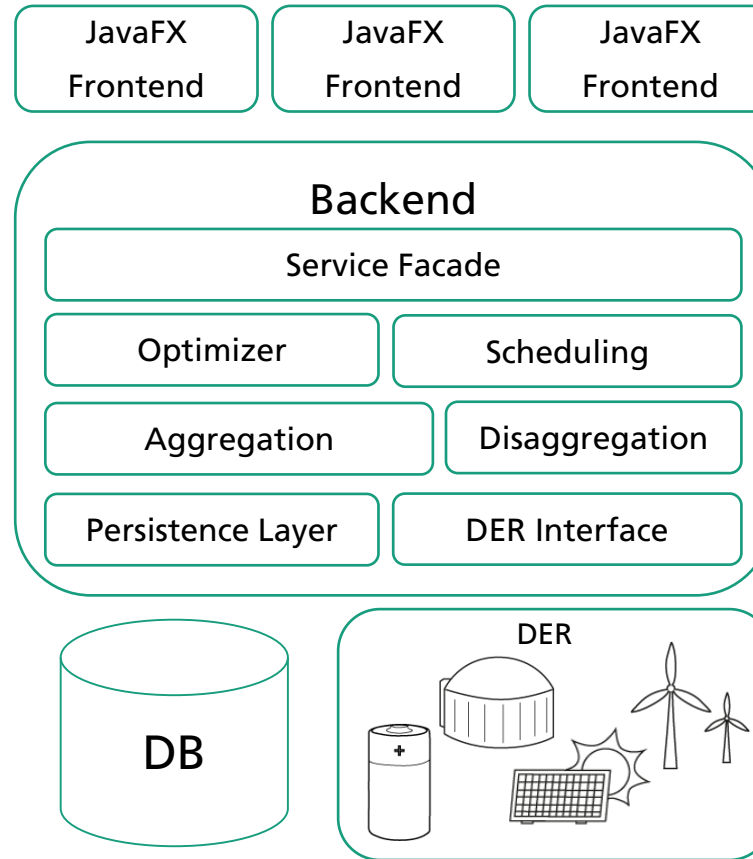
GENERAL STEPWISE APPROACH

1. Identify main use cases → choosing one use case to start with
2. Analyzing the software and examine the supporting parts
3. Extracting only the business logic to a SOA like Service
4. Proceed with migration infrastructure layers and introducing persistence
5. Finally taking independent UI considerations

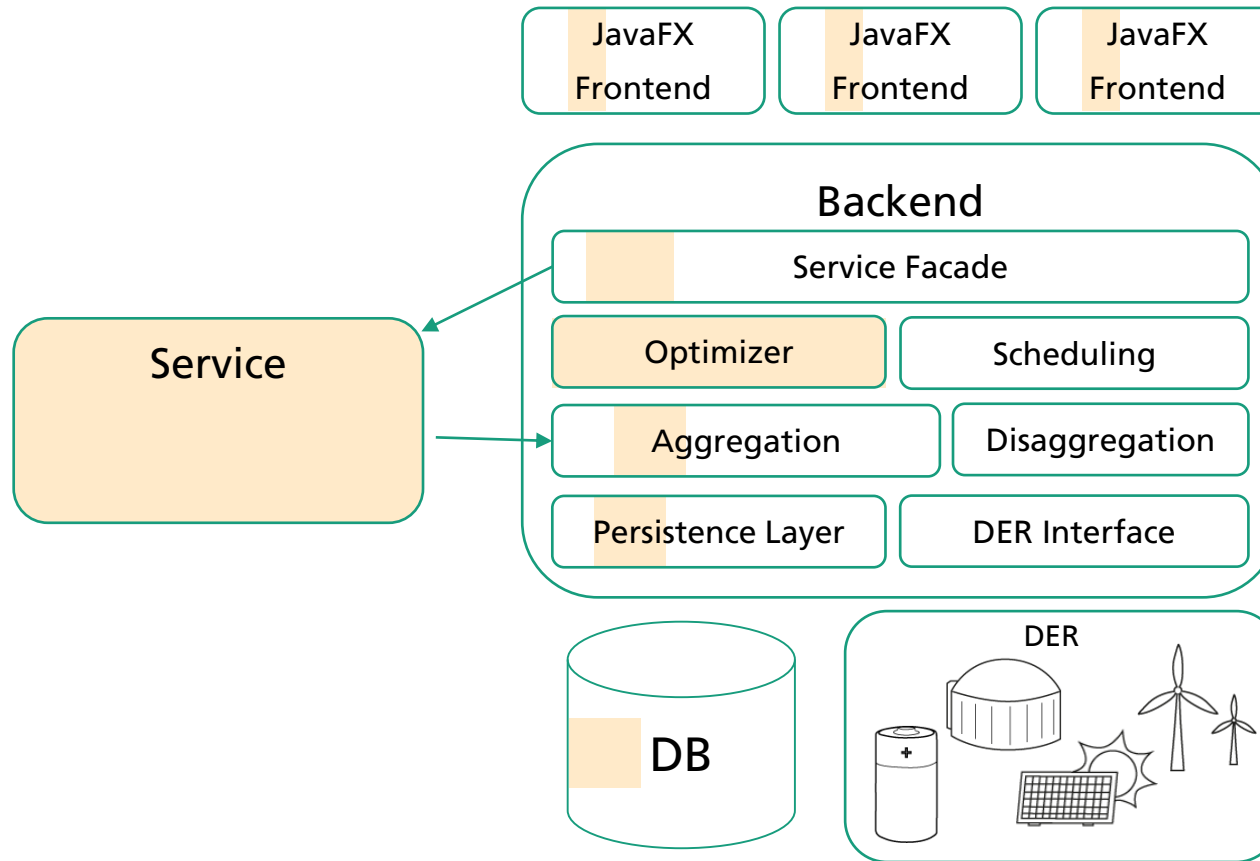
SOFTWARE SYSTEM



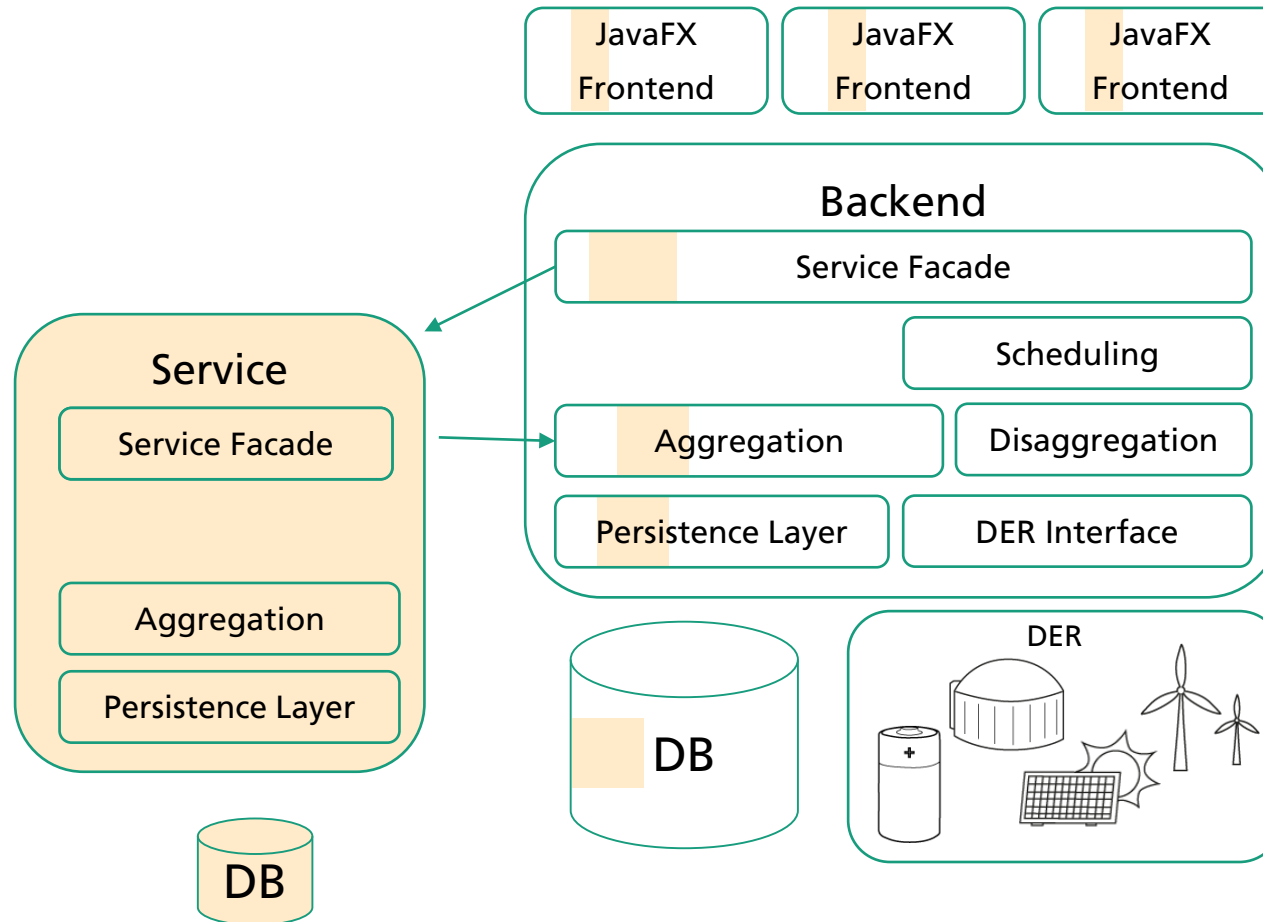
Migration



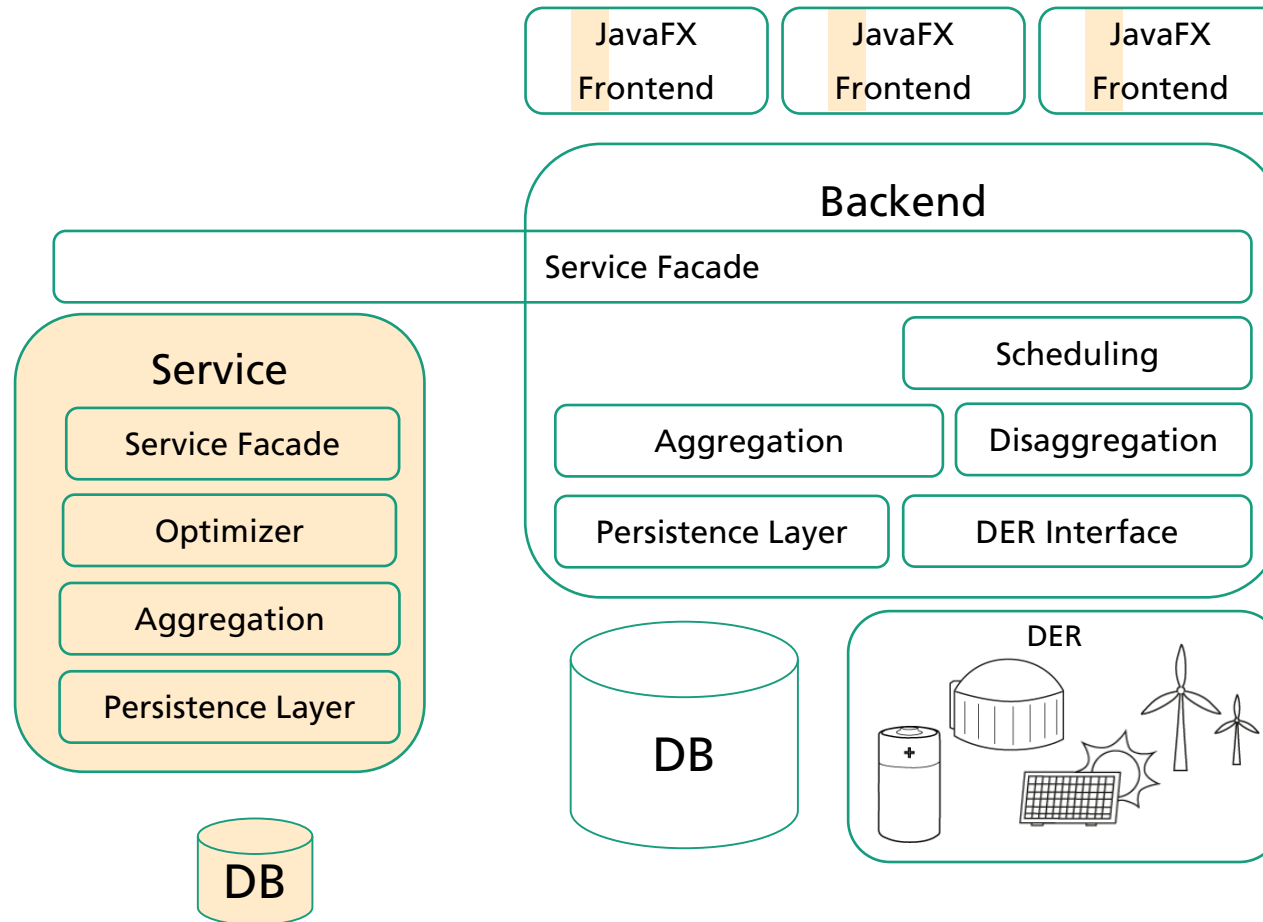
Migration



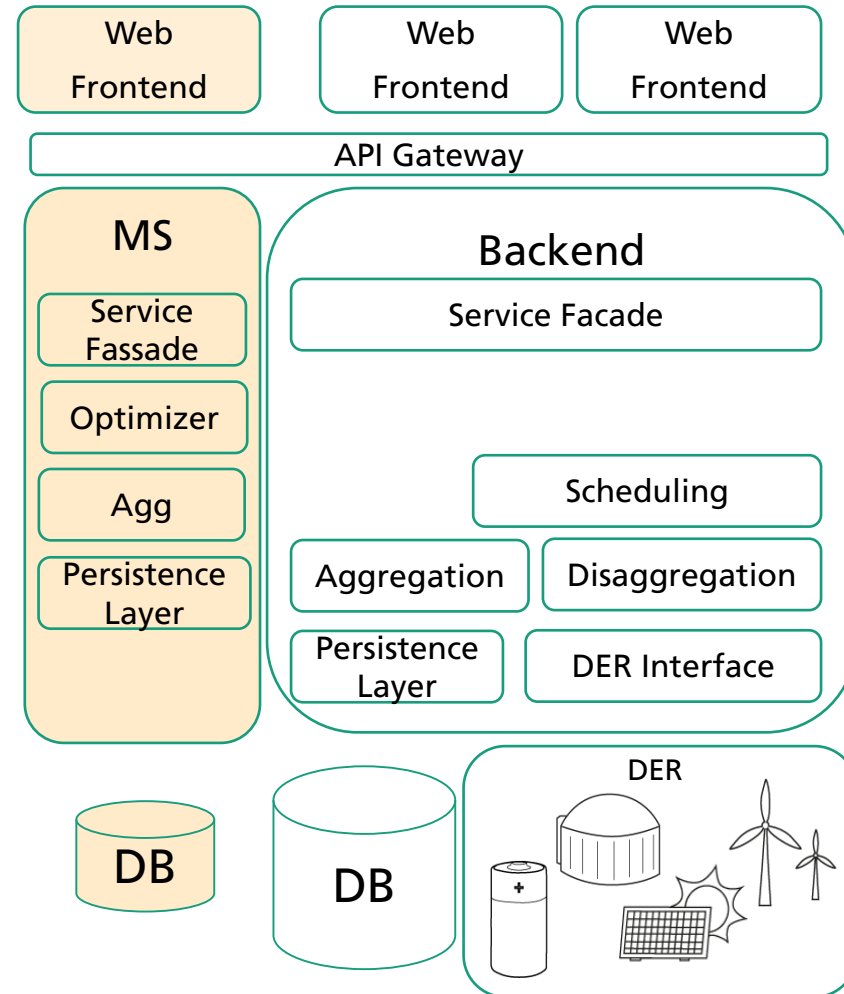
Migration



Migration



Migration



LEARNINGS?

LEARNINGS

- Provide continuously working software during migration
- **Main benefit:** Inclusion of further expert knowledge by using python
- Changing the optimization model is easier through the use bounded contexts
- Stepwise approach helped us to make smaller infrastructure changes possible

CONCLUSION

- Microservices seem to be a good architectural style for virtual power plants
- A Stepwise migration may generally be a good choice when :
 - Implementing MS in another programming language
 - Hosting on premise
- Next Steps
 - Webfrontends
 - Extend our scalability testing
 - More Microservices 😊

THANK YOU FOR YOUR ATTENTION



M. Sc. Manuel Wickert
Head of Department
Energy Informatics and Information Systems
Tel. +49 561 – 7294 – 369
manuel.wickert@iee.fraunhofer.de