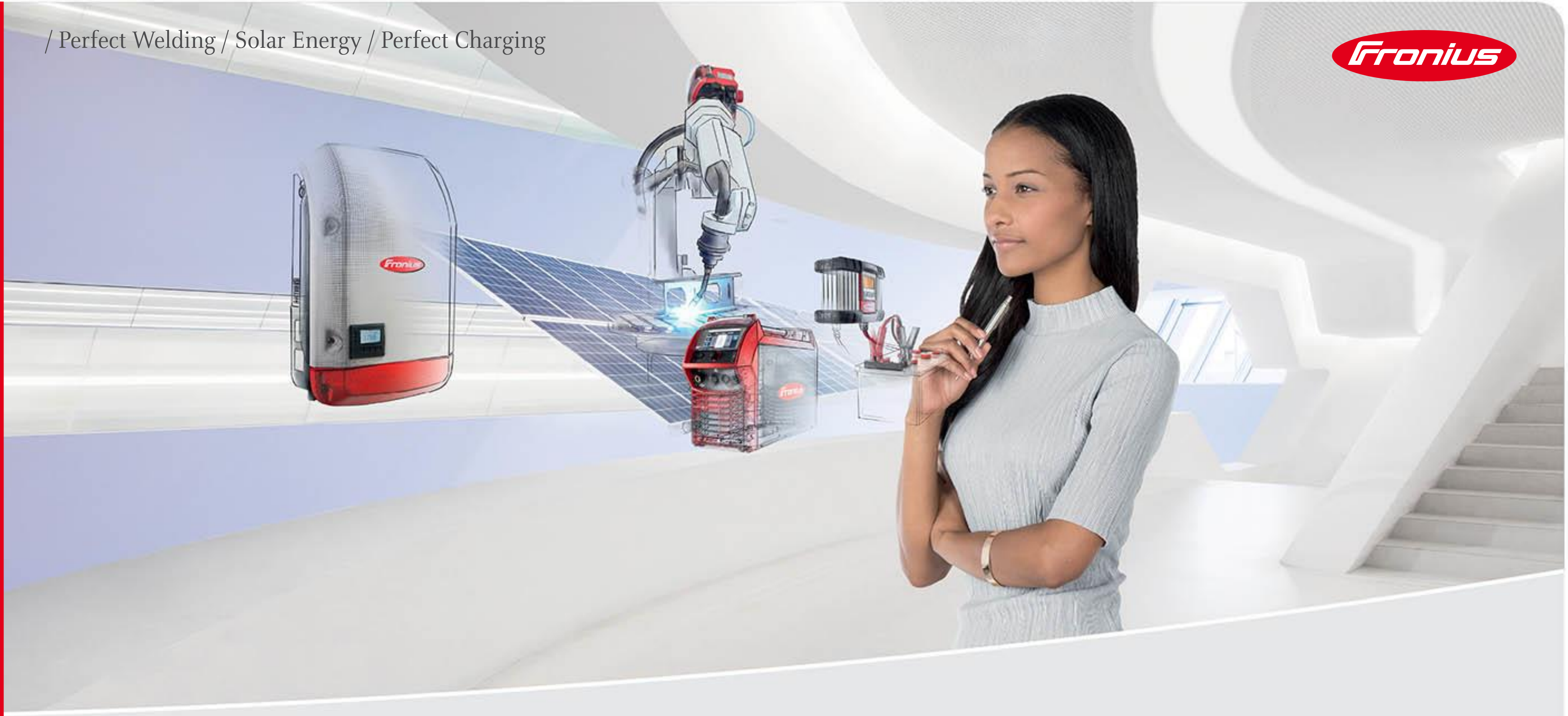


Joachim Danmayr, Fabian Neubacher
Fronius International GmbH
Froniusplatz 1, 4600 Wels

MODULAR SOFTWARE ARCHITECTURE AND SYSTEM TESTING IN THE PROCESS ENGINEERING SECTORS FOR PRODUCING GREEN HYDROGEN

Hydrogen Technology
First Solhub
2018

Welding technologies
Welding Robots
2015

Charging
Three phase charger
2013

Perfect welding
CMT technology
2005

Hydrogen Technology
Start Hydrogen research
2002

Solar energy
New inverter technologies
2001

Perfect welding
Digitalization
1998

Solar energy
First solar inverter
1995

Founded
1945

Founded new Team for hydrogen research and development
2019

Hydrogen intra-logistics at BMW
2017

Wind to hydrogen
2015

Start with hydrogen technology research
2002

HySnow research project
2020

Solhub prototype
2018

Energy self-sufficient house
2012

Hydrogen in intralogistics
2007

Joachim Danmayr

University of Hagen
Computer Science
2013 -

JKU Linz
Technical physics
2009 - 2010

HTL Steyr
Electronics
2002 - 2007

Fronius
technical lead
software development
hydrogen systems
2019 -

Fronius
software architect
embedded systems
solar inverter
2015 - 2019

Fronius
hard- and software developer
power electronics control
solar inverter
2008 - 2015

Fabian Neubacher

University of Applied Sciences
Upper Austria
Human-centered Computing (MSc)
2017 - 2019

University of Applied Sciences
Upper Austria
Medical Engineering (BSc)
2011 - 2014

HTL Vöcklabruck
Machine and Plant Engineering
2005 - 2010

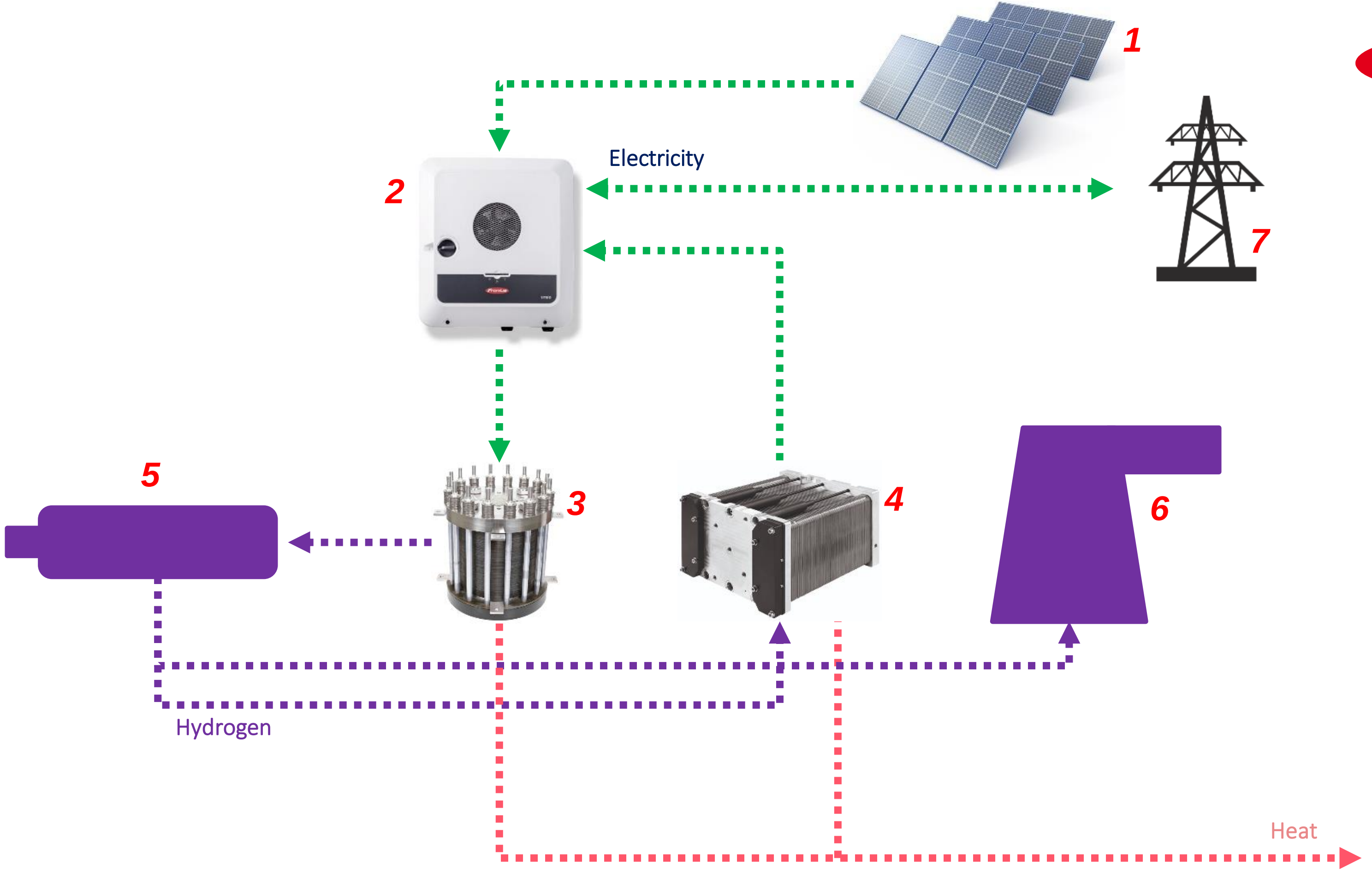
Fronius
Technical lead
testsystem development
hydrogen systems
2019 -

Novartis
Project Manager Engineering
Qualification Engineer
NTO - Aseptics
2016 - 2019

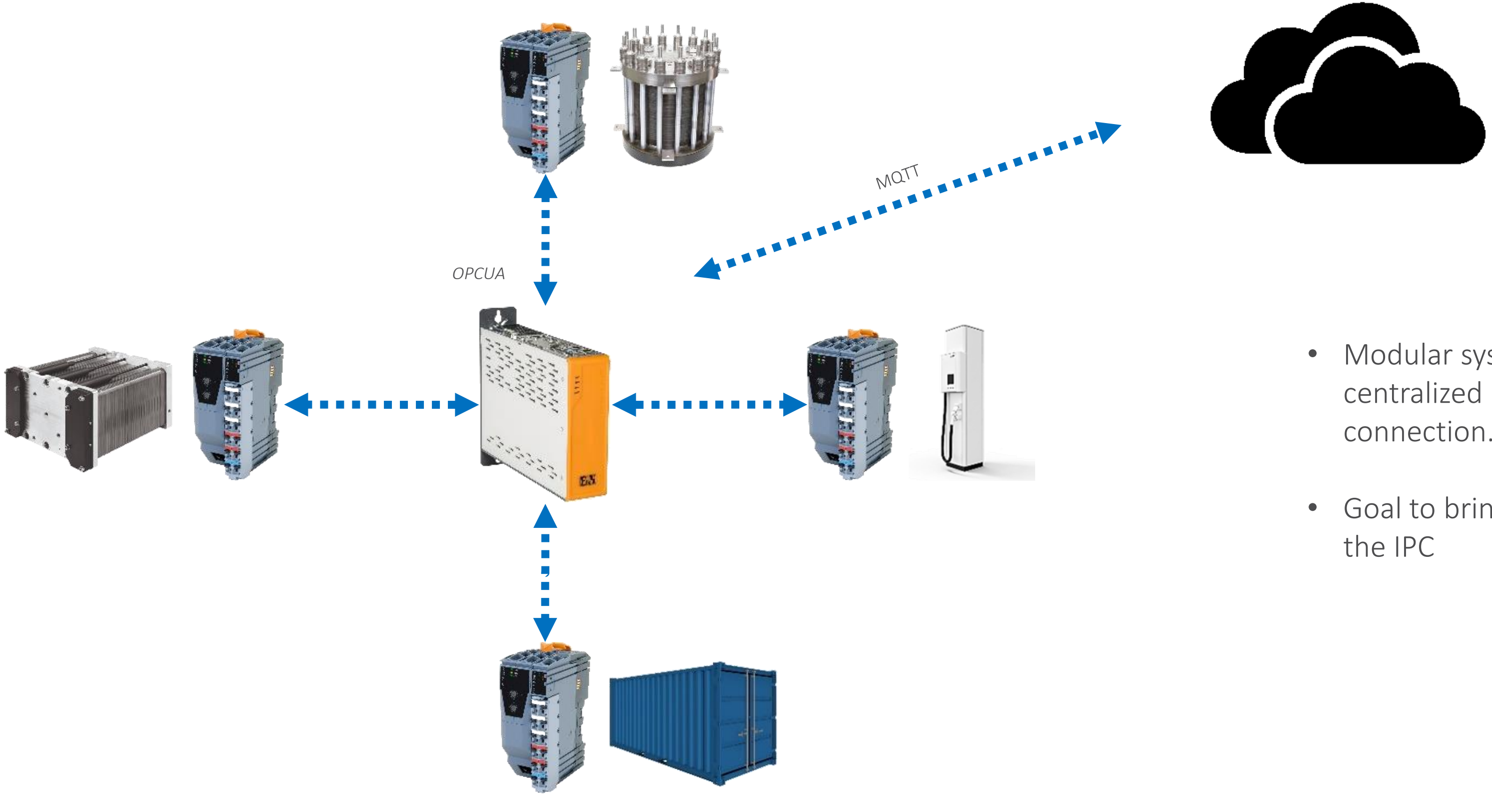
Austrian Red Cross
Emergency Paramedic
Vöcklabruck
2015 - 2016

SOLHUB


Fronius
Speicher
Storage
H_2
SOL
H_2UB
Wasserstoff
Hydrogen
Fuel Cell



1. Photovoltaics
2. Inverter
3. Electrolyzer
4. Fuel Cell
5. Hydrogen storage
6. Dispenser
7. Grid

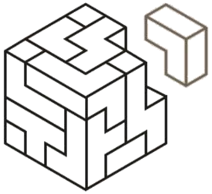


- Modular system design with centralized IPC and cloud connection.
- Goal to bring the complexity to the IPC

- Classically the main functions are implemented in PLCs.

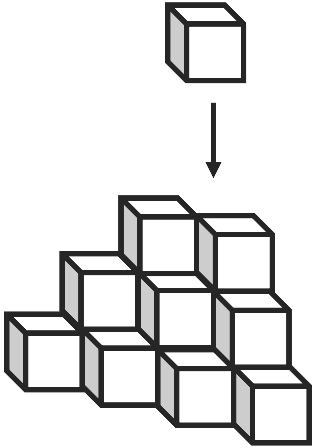
- We are focused on outsourcing most of the functionality to the IPC.

- The IPC has a microservice based architecture design which allows us to develop evolutionary.



MICROSERVICES

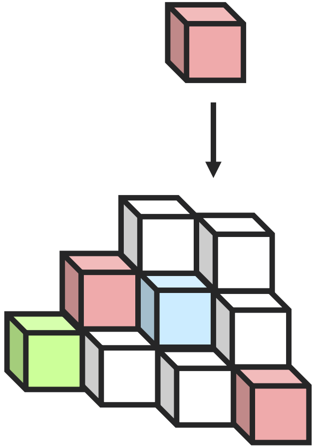
microservices| also known as the **microservice** architecture - is an architectural style that structures an application as a collection of services that are highly maintainable and testable, loosely coupled and independently deployable. [1]



[1] *What are microservices?* (2020), microservices.io, https://microservices.io/

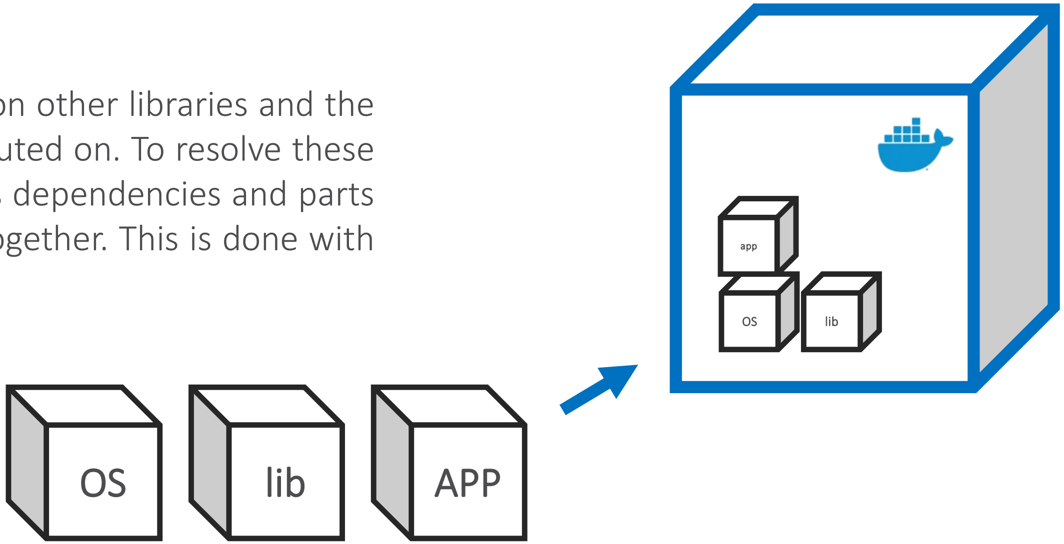
What is a microservice and why we need it?

advantages| complex systems can be divided into little, less complex components. These components can be developed independently from each other and put together later on.



How to get such independent loosely coupled software components?

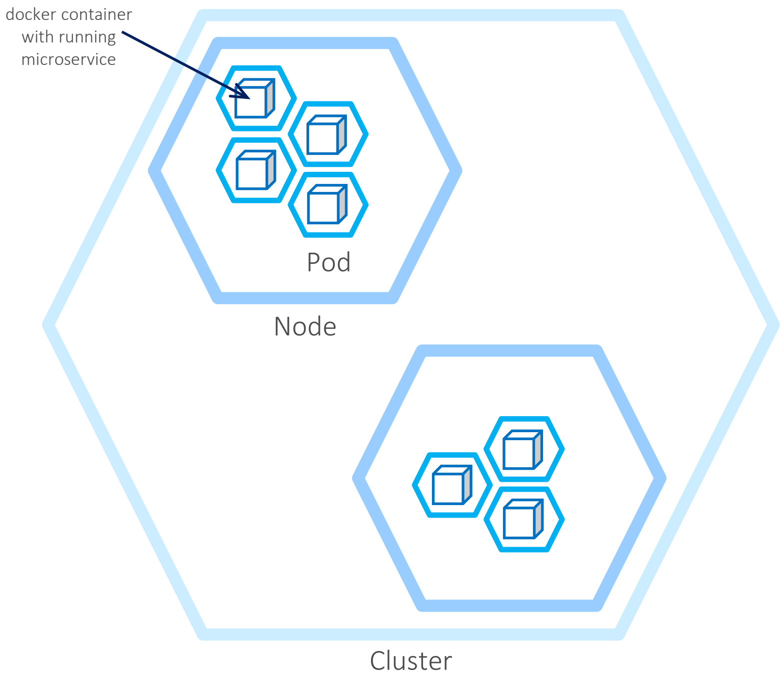
docker| software programs depends on other libraries and the operating system the program is executed on. To resolve these dependencies the program with all its dependencies and parts of the operating system are packed together. This is done with so called docker images.



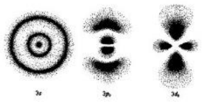

```
>> docker build
```

How to execute and manage these docker images?

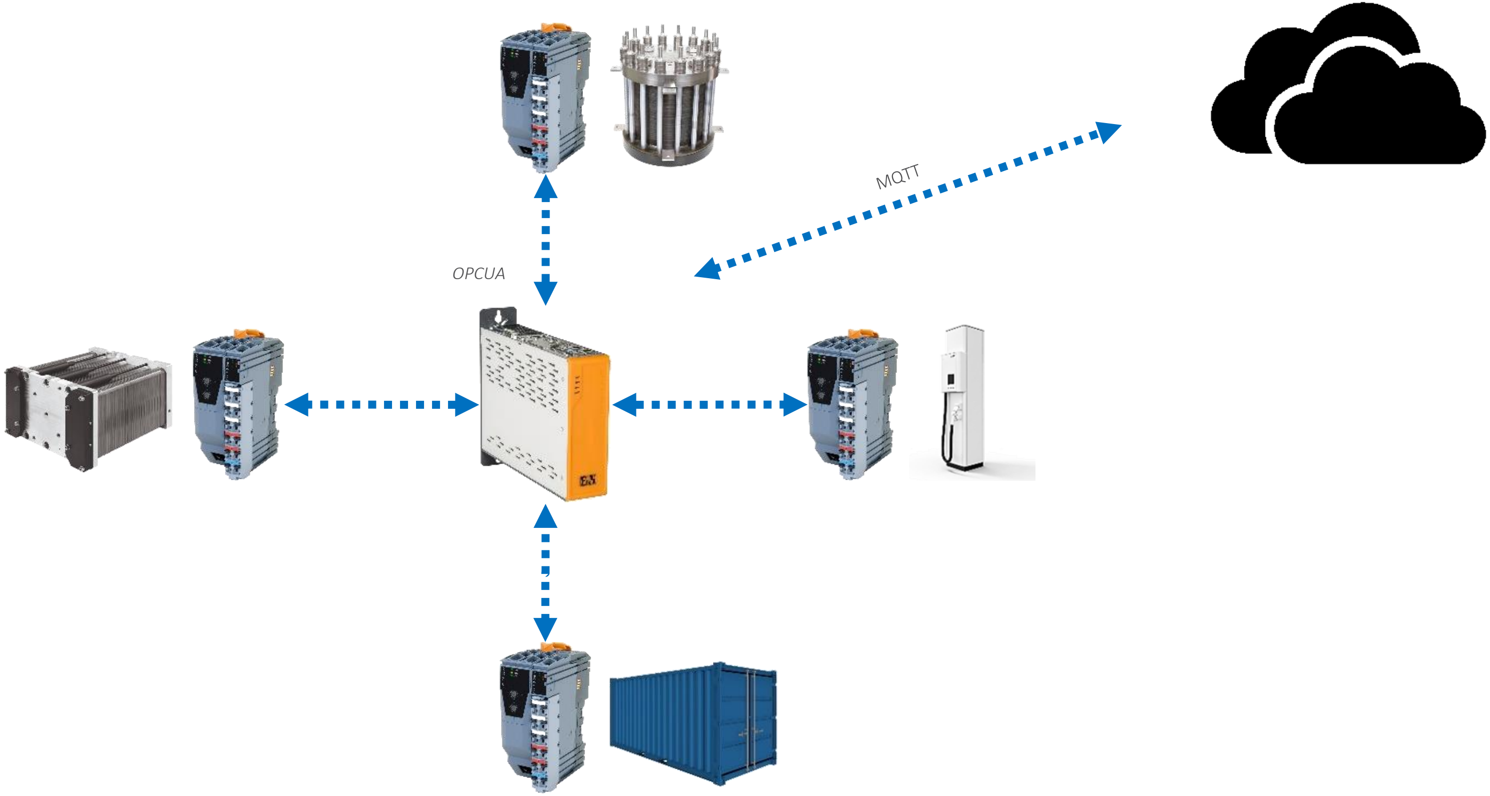
kubernetes| now the microservices are packed into images. For deployment, executing and monitoring kubernetes is used. With kubernetes docker images can be started and stopped as well as updated. Kubernetes allows to create clusters which manage load balancing.



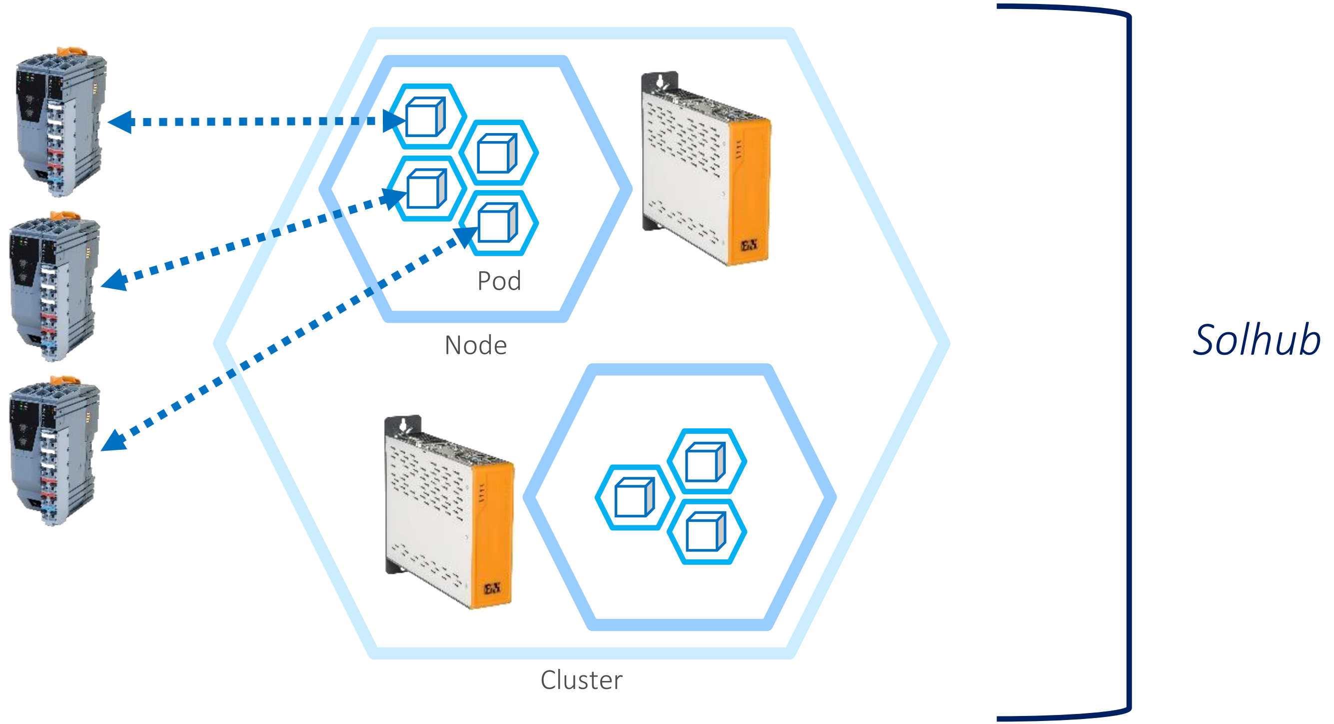

```
>> kubectl apply -f myDeployment.yaml
```



SOLHUB

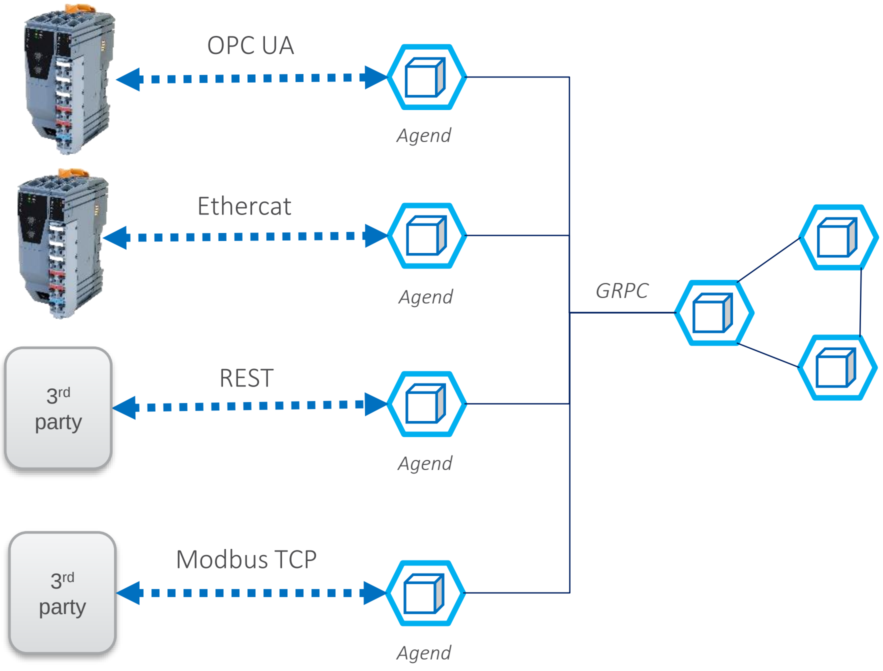

MQTT
OPCUA

Standalone hydrogen system…

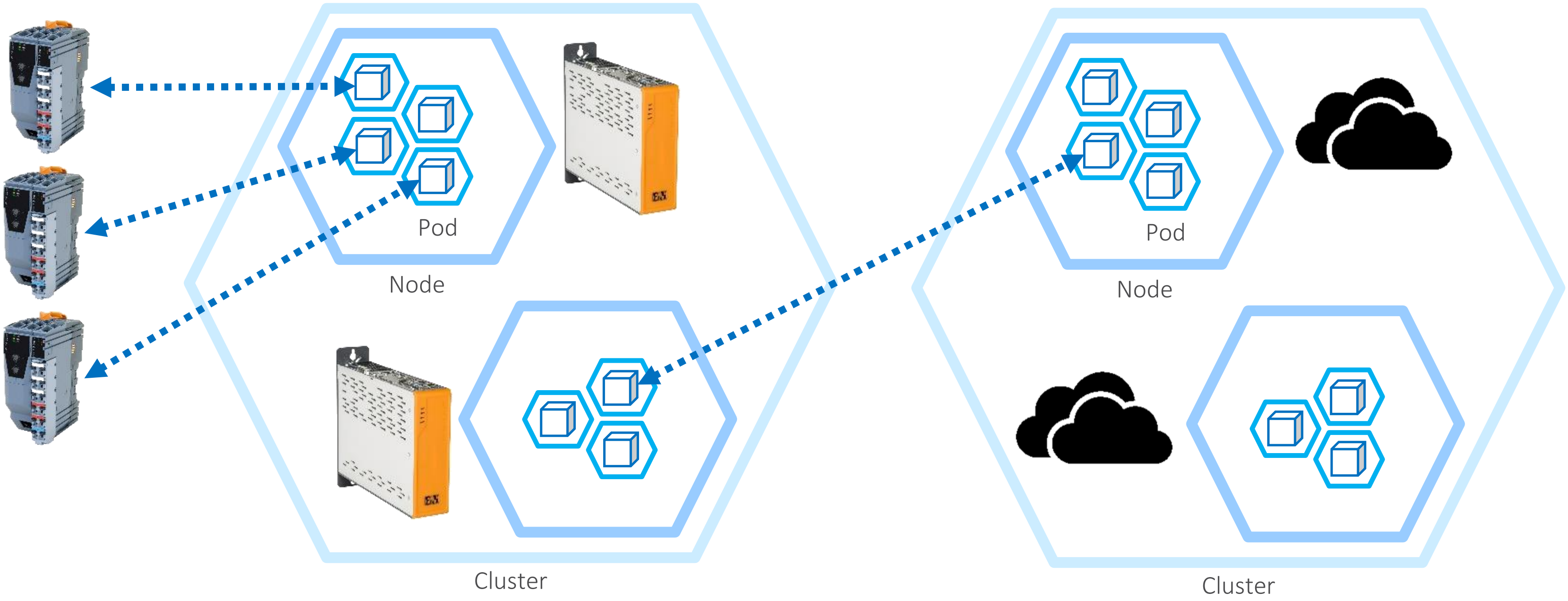


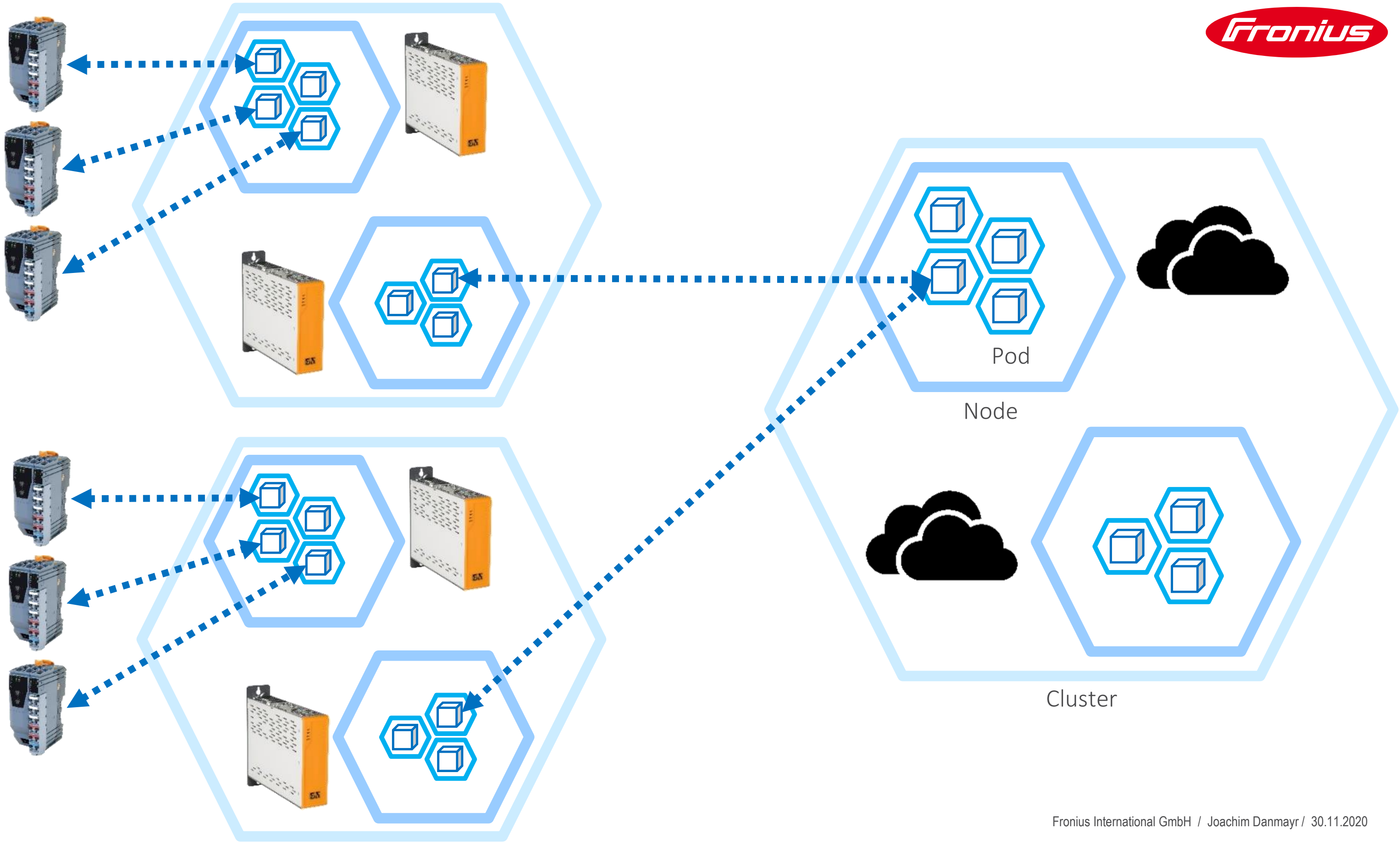
Agend system …

agend| agend microservices are used for representing an embedded system as microservice in the high level system.

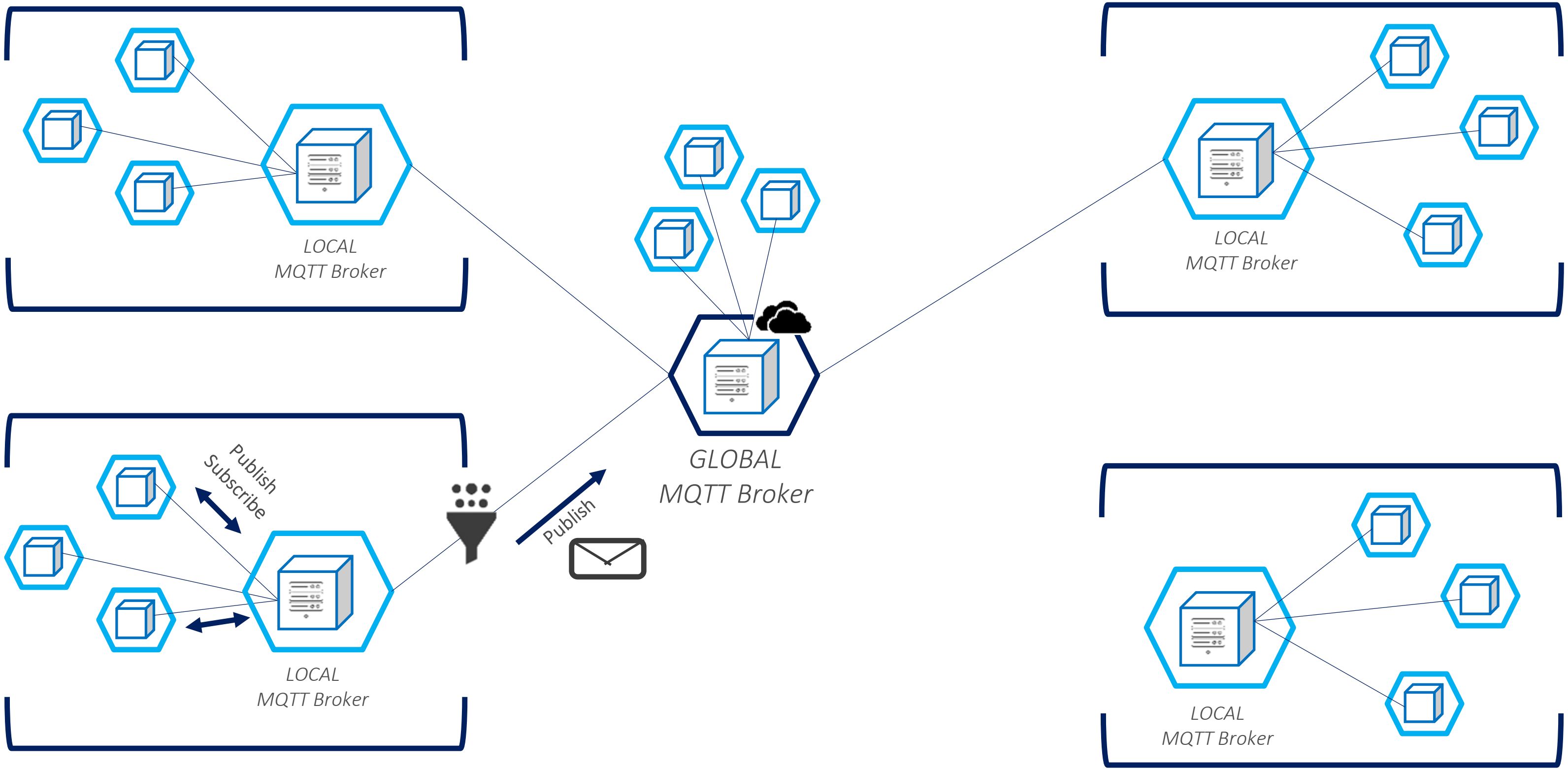


Connection to cloud…



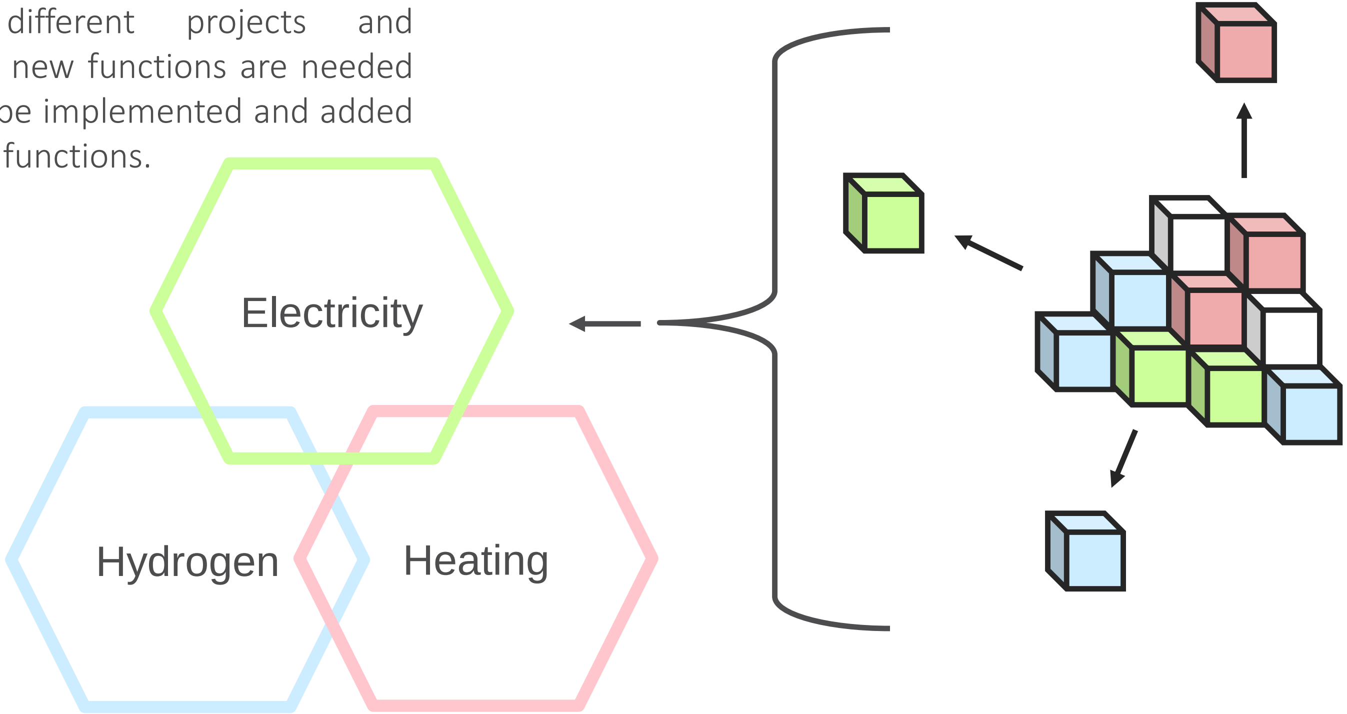

Pod
Node
Cluster

MQTT ...



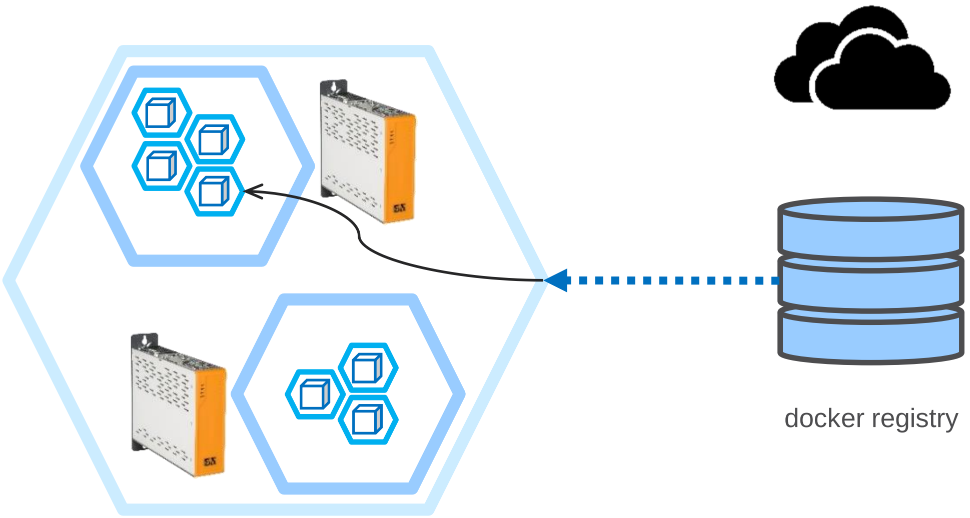
LOCAL
MQTT Broker
Publish
Subscribe
LOCAL
MQTT Broker
Publish
GLOBAL
MQTT Broker
LOCAL
MQTT Broker
LOCAL
MQTT Broker

reusability| the microservices can be used for different projects and application. If new functions are needed they can just be implemented and added to the pool of functions.

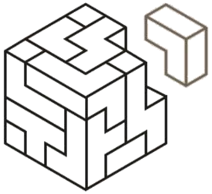


How does this work?

registry| there is a centralized Fronius docker registry in the cloud. This registry can be accessed from all Fronius devices. The registry contains all available microservices. Kubernetes downloads the needed images from this registry and starts the contains within its cluster.

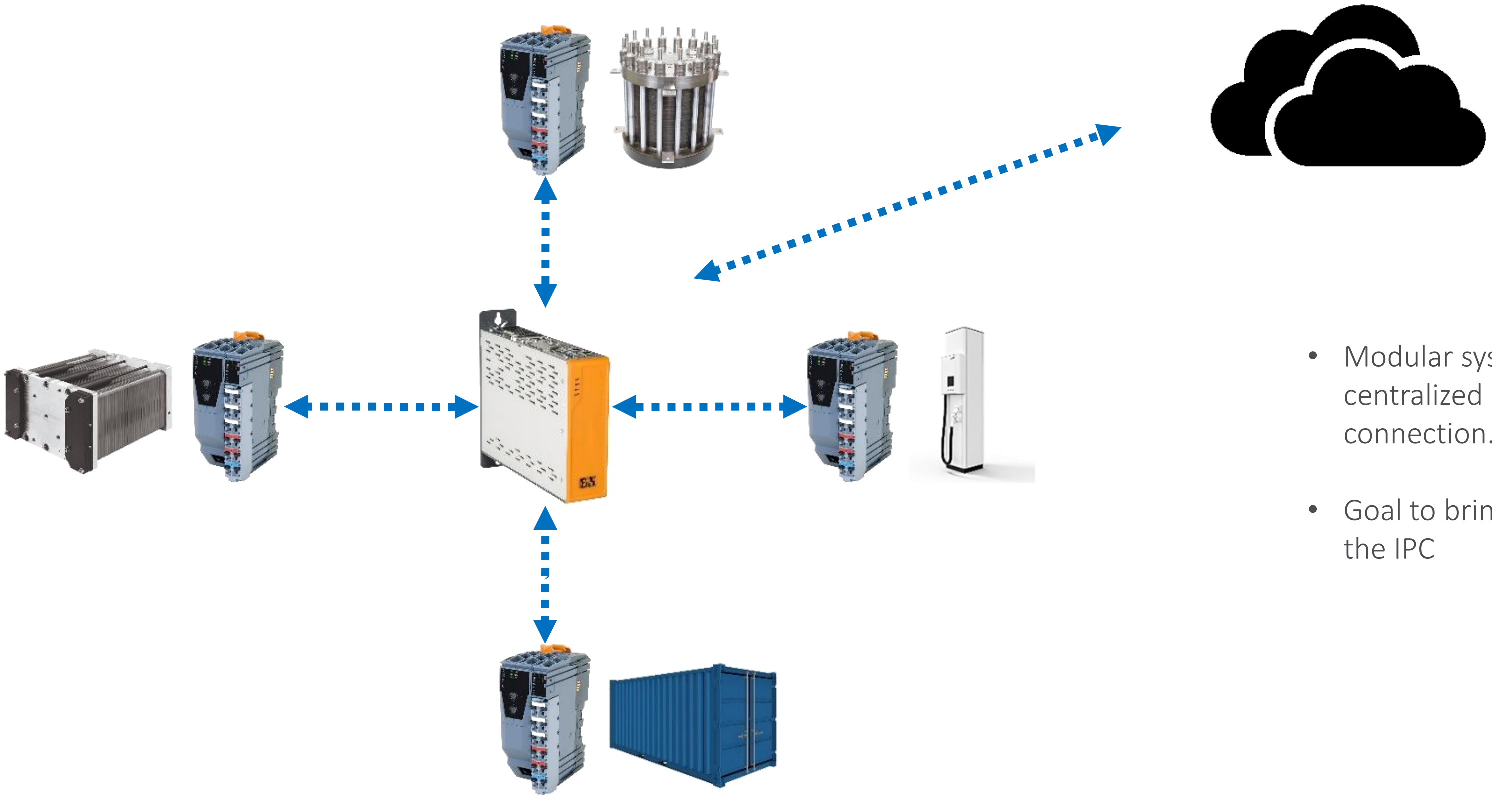



```
>> docker pull
```



SYSTEM TESTING

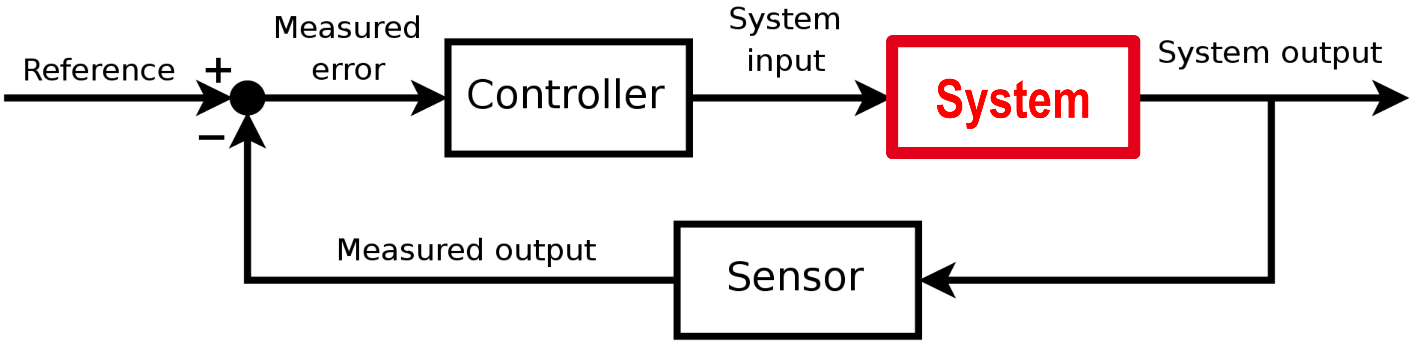
Overview…



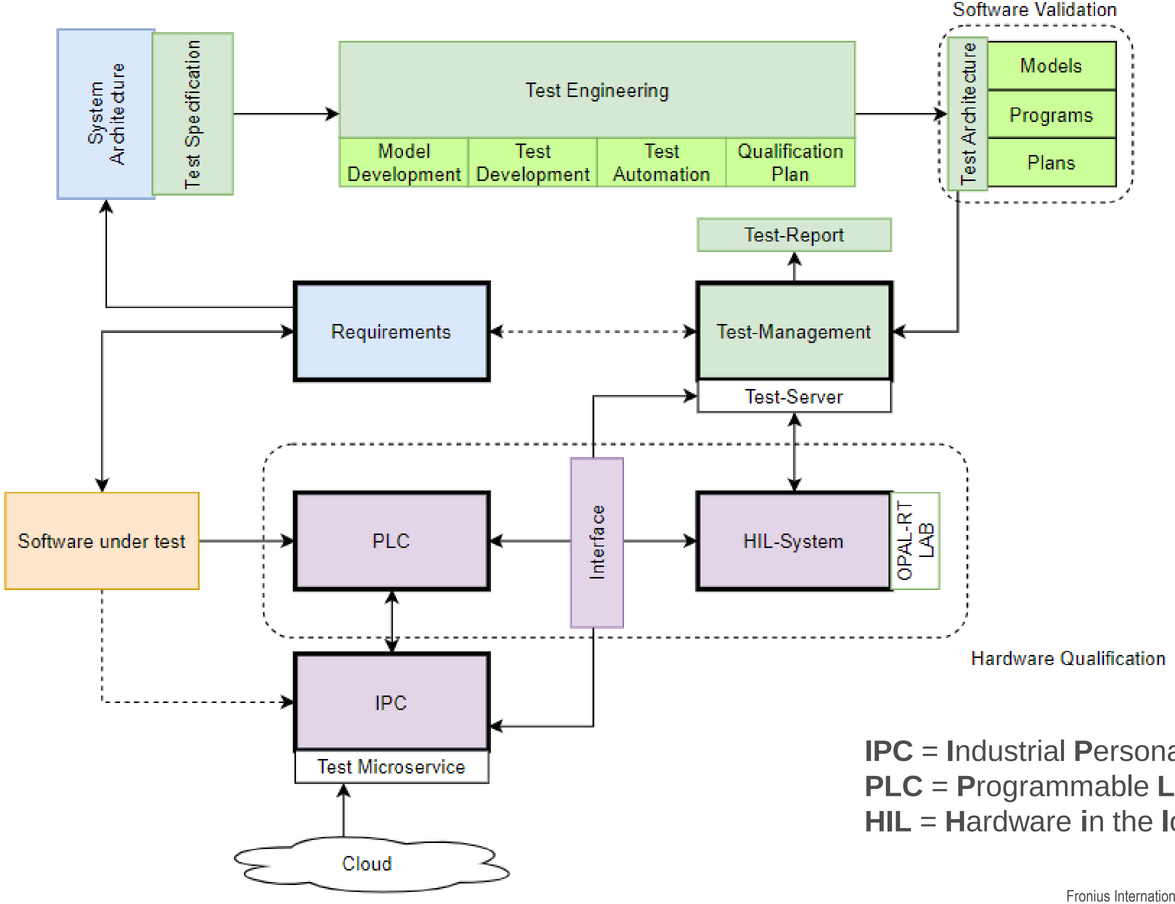
- Modular system design with centralized IPC and cloud connection.
- Goal to bring the complexity to the IPC

BASICS: HIL-SYSTEM

- HIL = Hardware in the loop
- QA for software controlled components according to 61508
- Real controller / simulated system

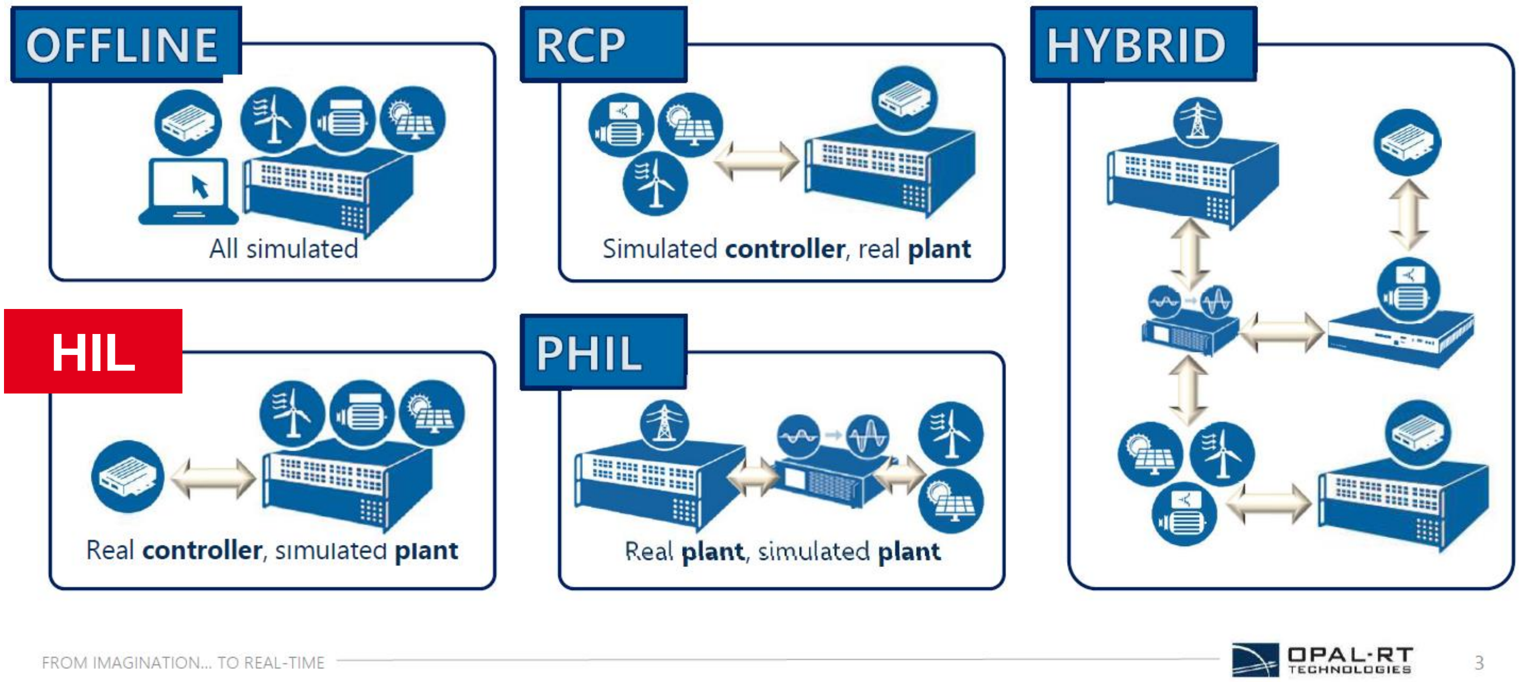


TEST-SYSTEM

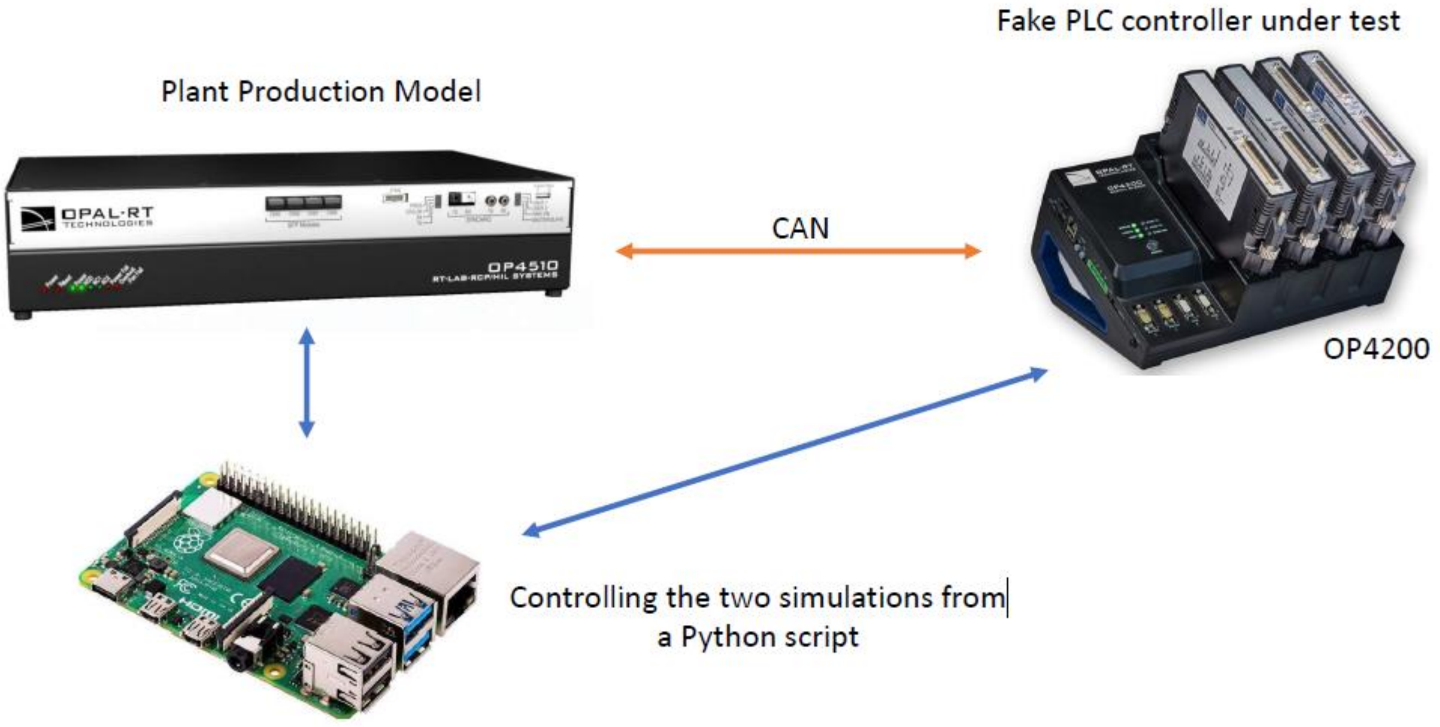


IPC = **I**ndustrial **P**ersonal **C**omputer
PLC = **P**rogrammable **L**ogic **C**ontroller
HIL = **H**ardware **i**n the **l**oop

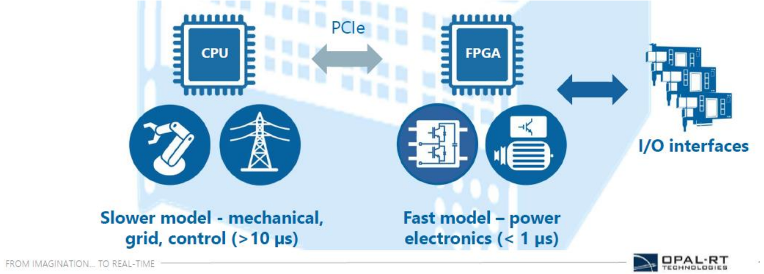
OPAL-RT SYSTEM



ACTUAL STRUCTURE

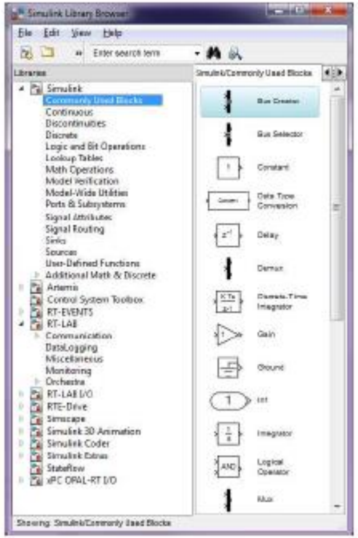


ACTUAL STRUCTURE

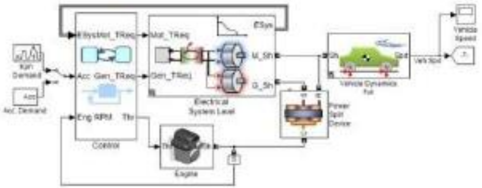


Simulation environment:

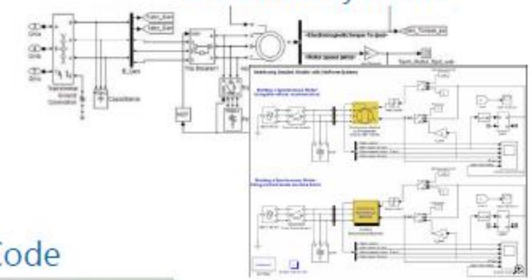
Matlab/Simulink, Simscape



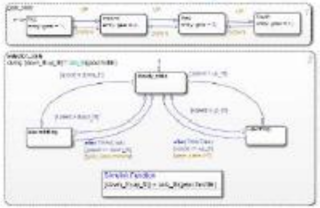
Physical models



Electrical systems

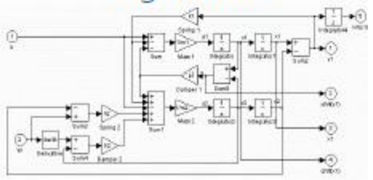


State diagrams

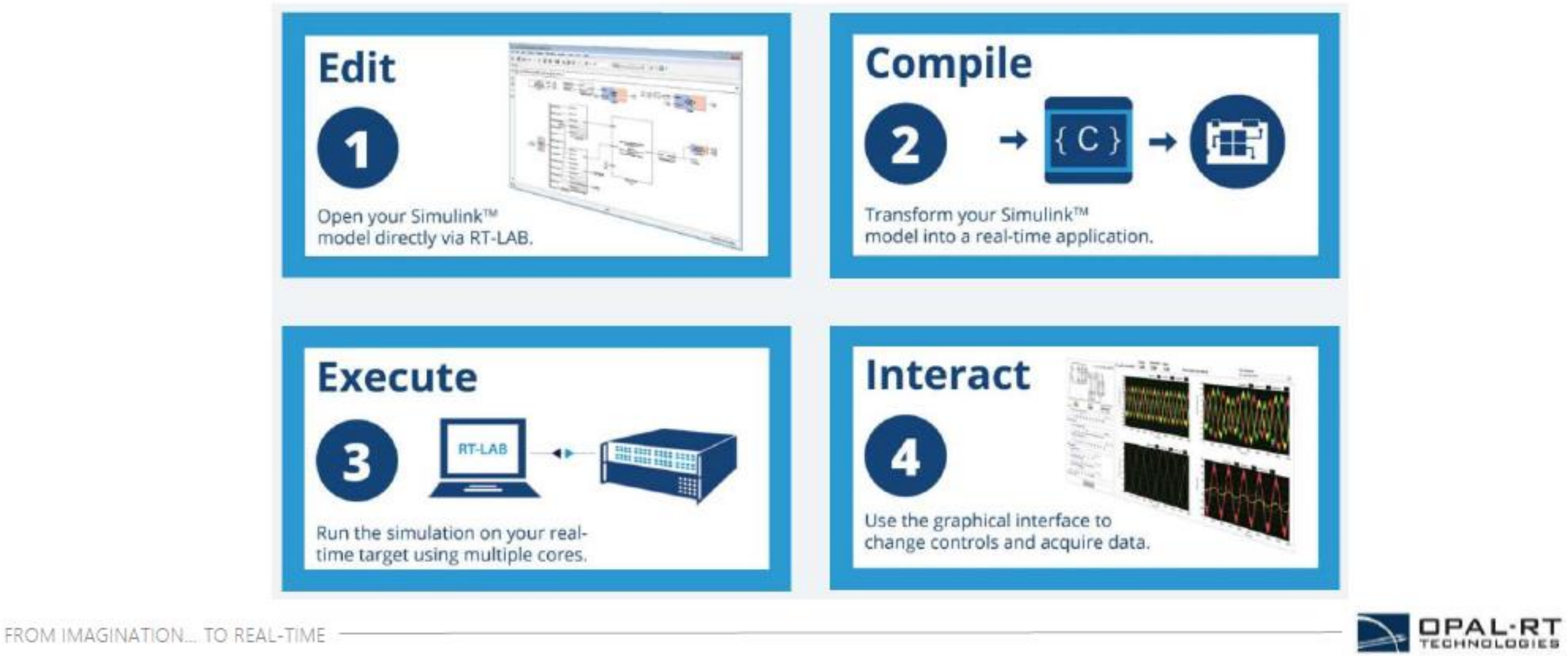


Code

Algorithms



FROM MODEL TO REAL TIME



CE-COMPLIANT TEST ARCHITECTURE

AREA	DESIGN	VERIFICATION	TEST
HIL (OPAL-RT Rack)	I/O Table including sampling rate	Factory calibration Design verification	Qualification Recalibration
Signal Conditioning	Requirements	Initial calibration	Recalibration
OPAL-RT Lab (Software)	Standards Proven on the market	NA	Newest release (maintenance agreement)
Mathworks --> C	Proven on the market State of the art	Code review (sample)	Newest release (maintenance agreement)
Models	Tracing back to physical and mathematical principles	Review (comparison with measured data)	Operation within defined limits
	Derivation from measured data	Review	Operation within defined limits
Test - Programs	Styleguides	Review	Implicit in test execution

/ Perfect Welding / Solar Energy / Perfect Charging