SOAFEE

SDV Challenges and Cloud-native System Design Approach

SOAFEE APAC Seminar

Wenhung Kevin Huang

DENSO CORPORATION

24th September 2024

Who am I

Wenhung Kevin Huang (黄　文鴻)
Project Assistant Manager

Tokyo Office, DENSO CORPORATION

Interest in software verification and safety-critical systems

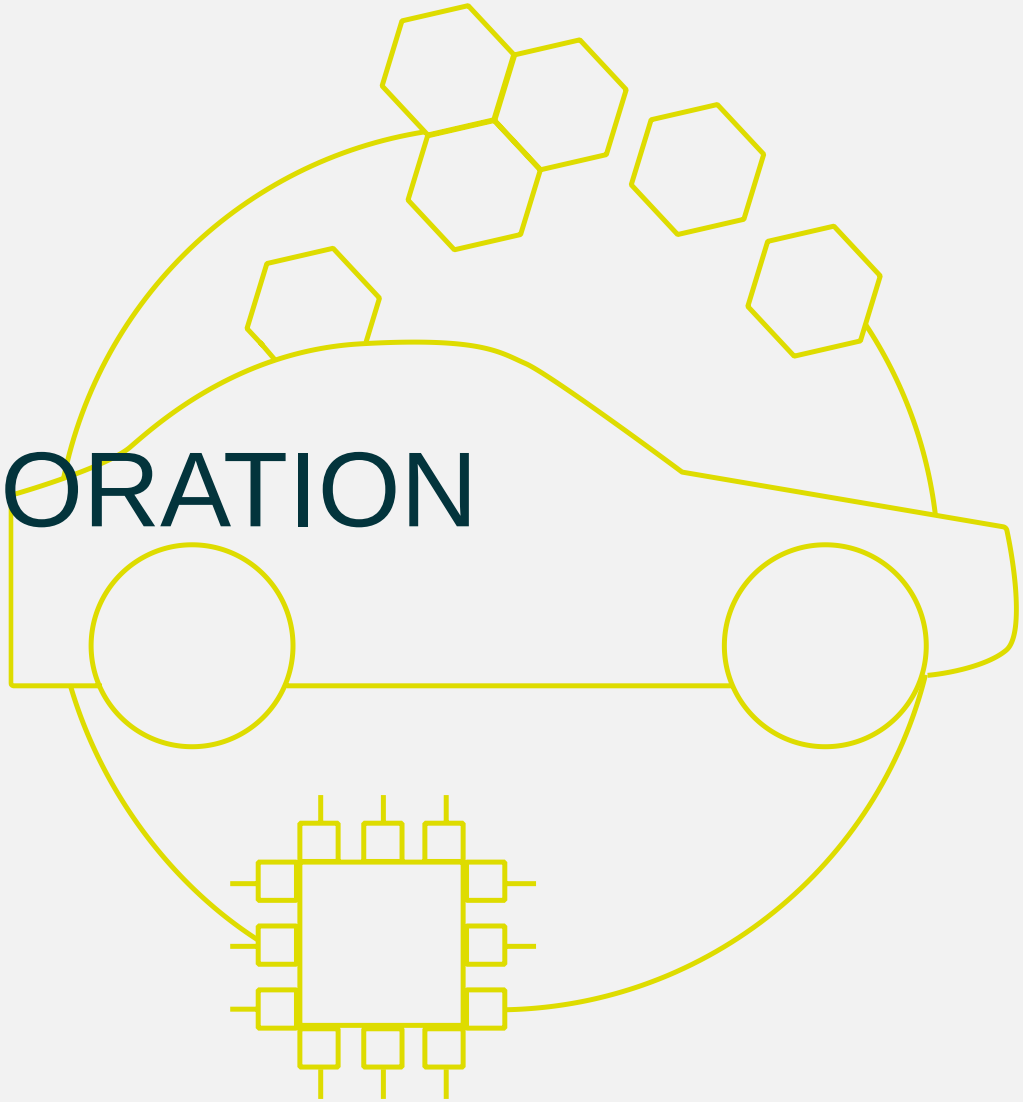
Love badminton, sauna, board game

Agenda

1. About DENSO CORPORATION
2. The Future of Mobility & SDV Use Cases
3. Changes & Challenges brought by SDV
4. Solution - Cloud-native System Design Approach
5. Summary & Next Steps

SOAFEE

1. About DENSO CORPORATION



About DENSO CORPORATION

DENSO
Crafting the Core

- Global Fortune 500 company
- Focus on advanced mobility
- Positively change how the world moves
- Contribute to greater well-being
- Broad product portfolio & widespread global impact

The DENSO Group
190 companies

Total number of employees
164,572 people

35 countries and regions

DENSO's Future Direction

4 core technologies

- Electrification
- Advanced Safety and Automated Driving
- Connected Driving
- Factory Automation /AgTech

denso.com/global/home/about-us/at-a-glance/

Green **Peace of mind**

Contribute to happiness for everyone through "green" and "peace of mind"

Software innovation by DENSO in the era of CASE | Newsroom | News | DENSO Global Website

Green

CO2 ±Zero
Aiming to become Carbon neutral by 2035

Monozukuri
(Manufacturing)

Realize complete carbon neutrality at our plants

Mobility Products

Realize an energy-recycling society through the development and popularization of technologies that make effective use of renewable energy

Energy Use

Contribute to the electrification of cars to reduce CO_2 emissions to the greatest extent possible

denso_brochure_en.pdf

Peace of mind

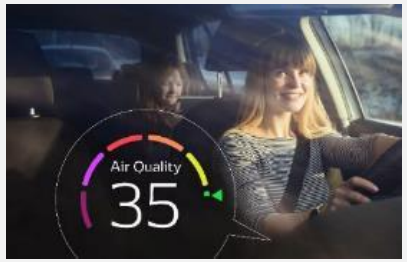
Without fatalities
Aiming to become a leading company that provides “Peace of Mind” to society

Elimination of Fatalities from Traffic Accidents

Popularize safety products through efforts focused on “depth” and “width,” thereby realizing free mobility without fatalities from traffic accidents

Creation of Comfortable Spaces

Enhance relevant technologies for creating peaceful, comfortable spaces



Support for Working People

Draw on the technologies we have calculated in the mobility domain to establish a society where people are supported and their potential is nurtured

Purpose of this presentation

As part of Mixed-Criticality WG, we would like to

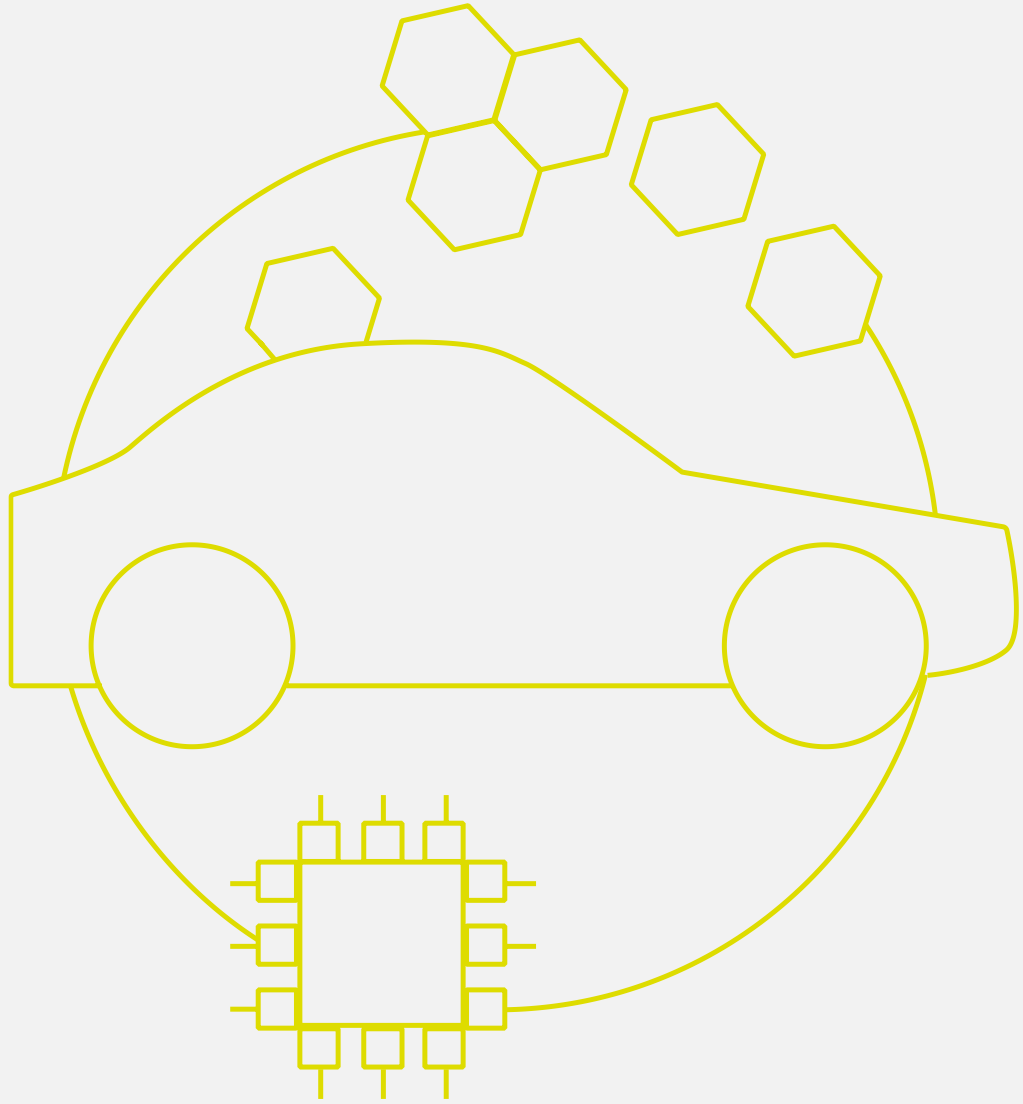
- **Share our current understandings and progress on the foreseen challenges of mixed-criticality drawn by the SDV use cases**
- **Propose our solution ideas to such SDV's mixed-criticality challenges**

Agenda

1. About DENSO
2. The Future of Mobility & SDV Use Cases
3. Changes & Challenges brought by SDV
4. Solution - Cloud-native System Design Approach
5. Summary & Next Steps

SOAFEE

2. The Future of Mobility & SDV Use Cases

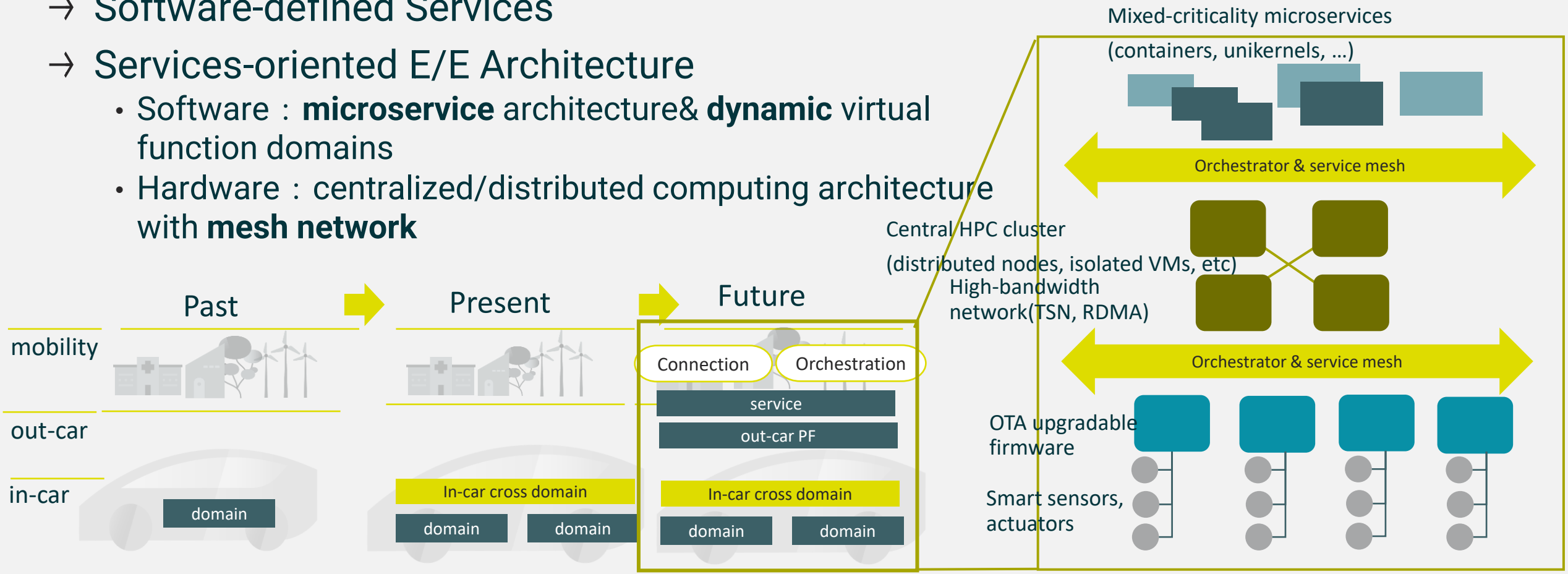


The Future of Mobility: Software-defined Services

→ Software-defined Services

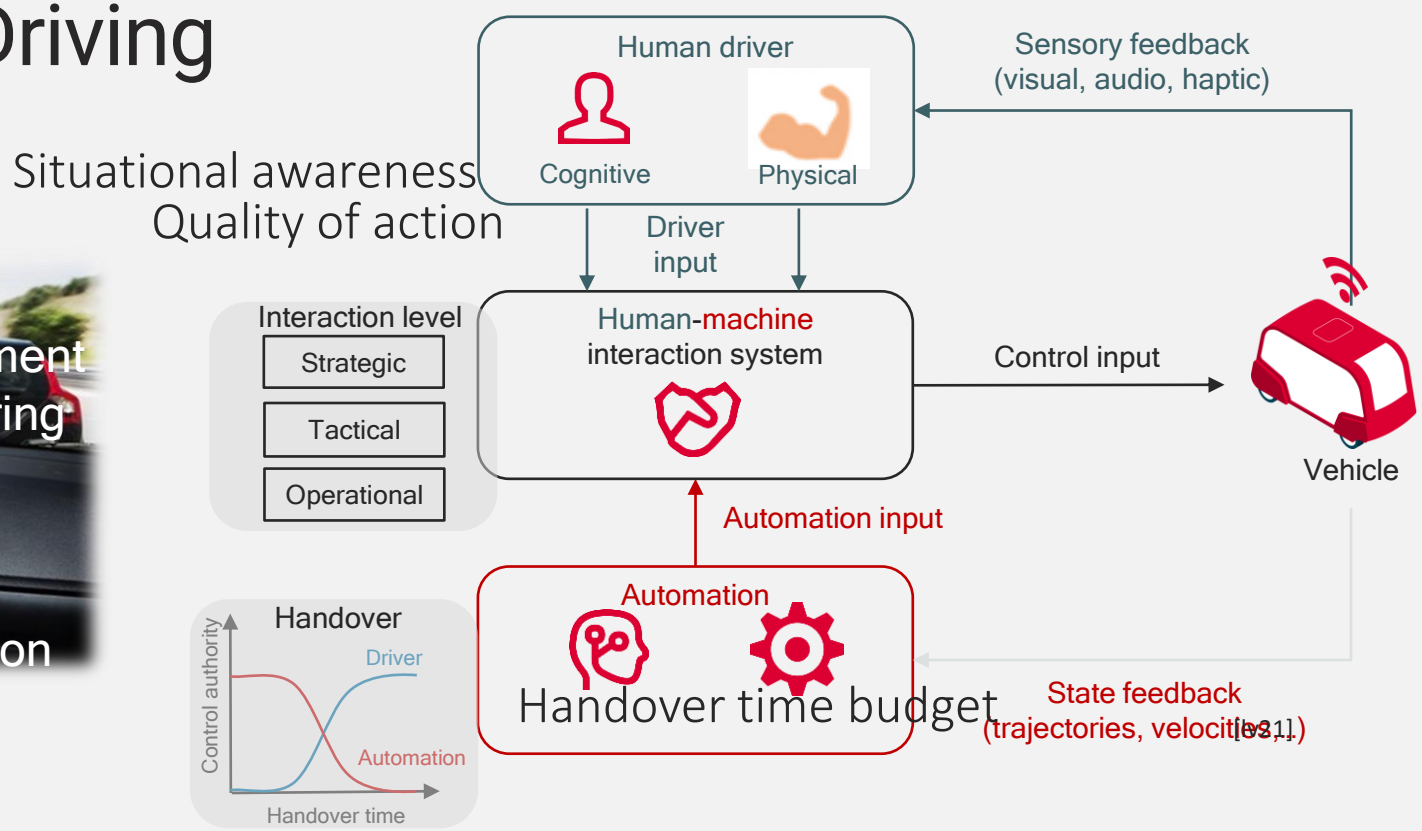
→ Services-oriented E/E Architecture

- Software：**microservice** architecture& **dynamic** virtual function domains
- Hardware：centralized/distributed computing architecture with **mesh network**



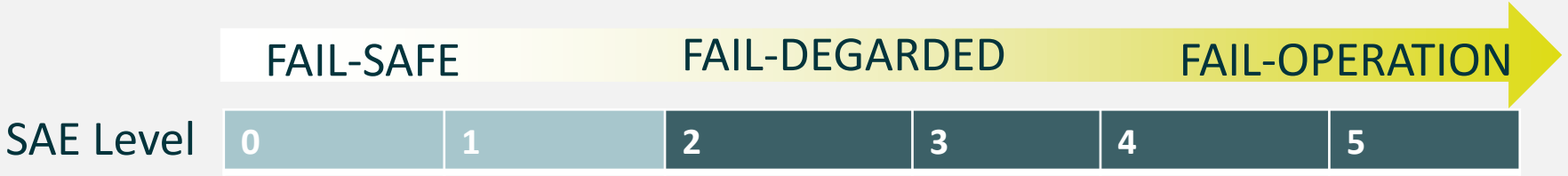
デンソーが推進するCASE時代におけるソフトウェア改革｜DRIVEN BASE（ドリブンベース）- デンソー (denso.com)

Use Case - Autonomous Driving



→ Safe operation when a fault happens

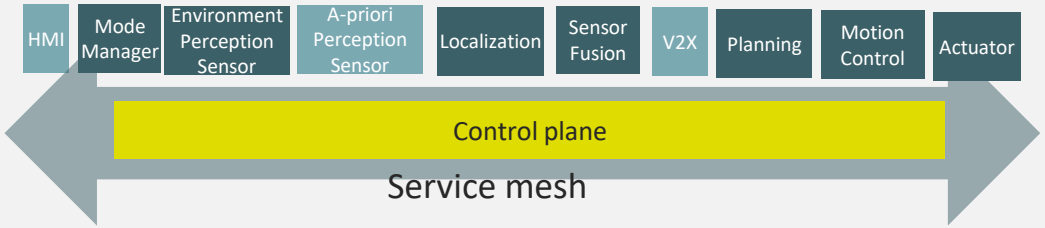
- Fail-Safe: a system stops operations and transitions to safe state
- Fail-Degraded/Fail-Operation: a system continues operation with below/at least nominal performance



Safety First for Automated Driving (SaFAD paper) (2019-07)

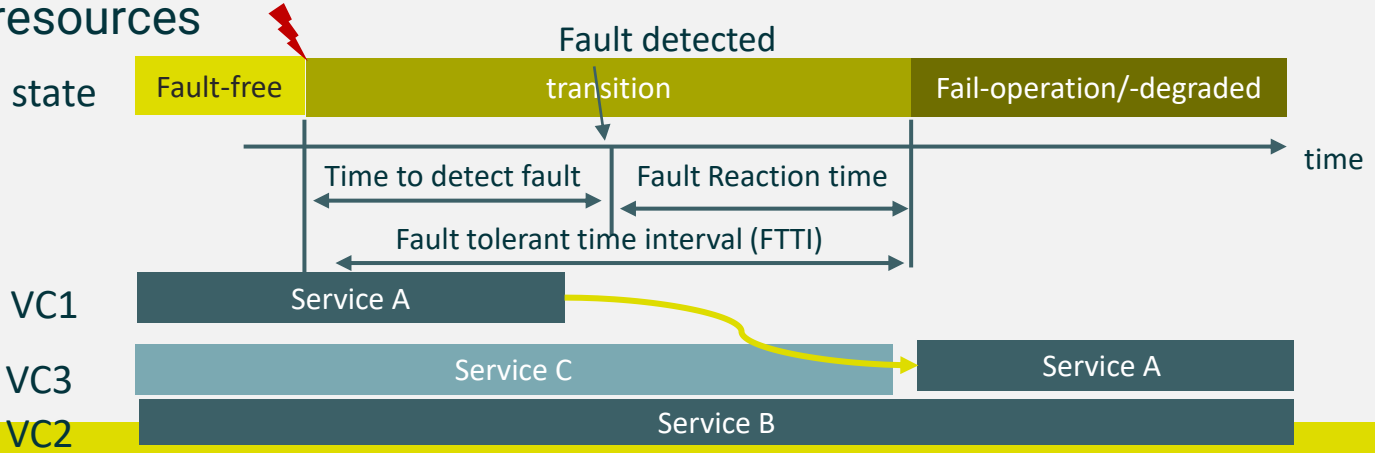
Use Case - Autonomous Driving

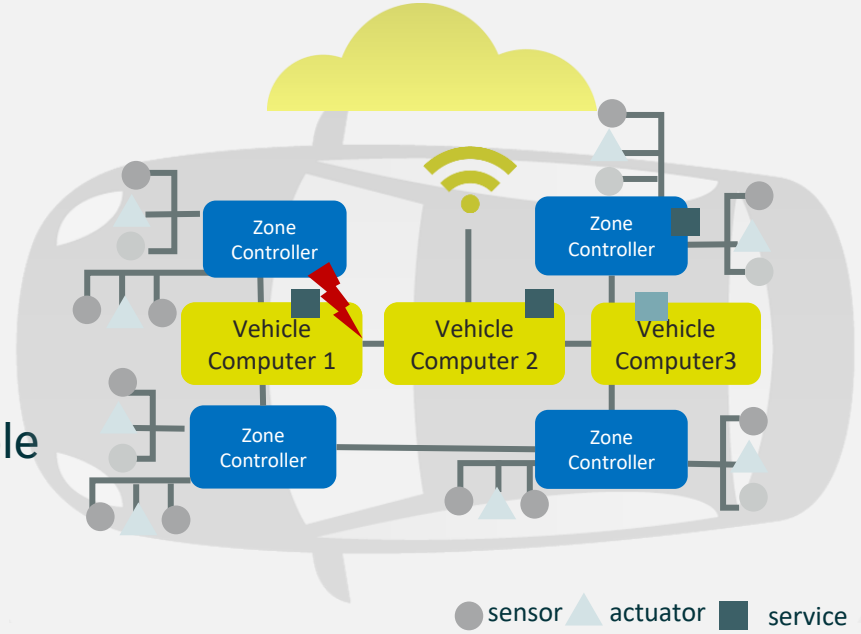
Fail-Degraded/Fail-Operation



→ Safety requirement
- A system continues operation with below/at least nominal performance+ timing requirement

→ Software-defined approach
- Functions are composed by services, running by different ECUs
- Zonal ECU concept matches the demands of service-oriented architectures
- Mutual monitoring, failover, safety-aware platform
- When a fault happens, rebalancing is performed according to available resources





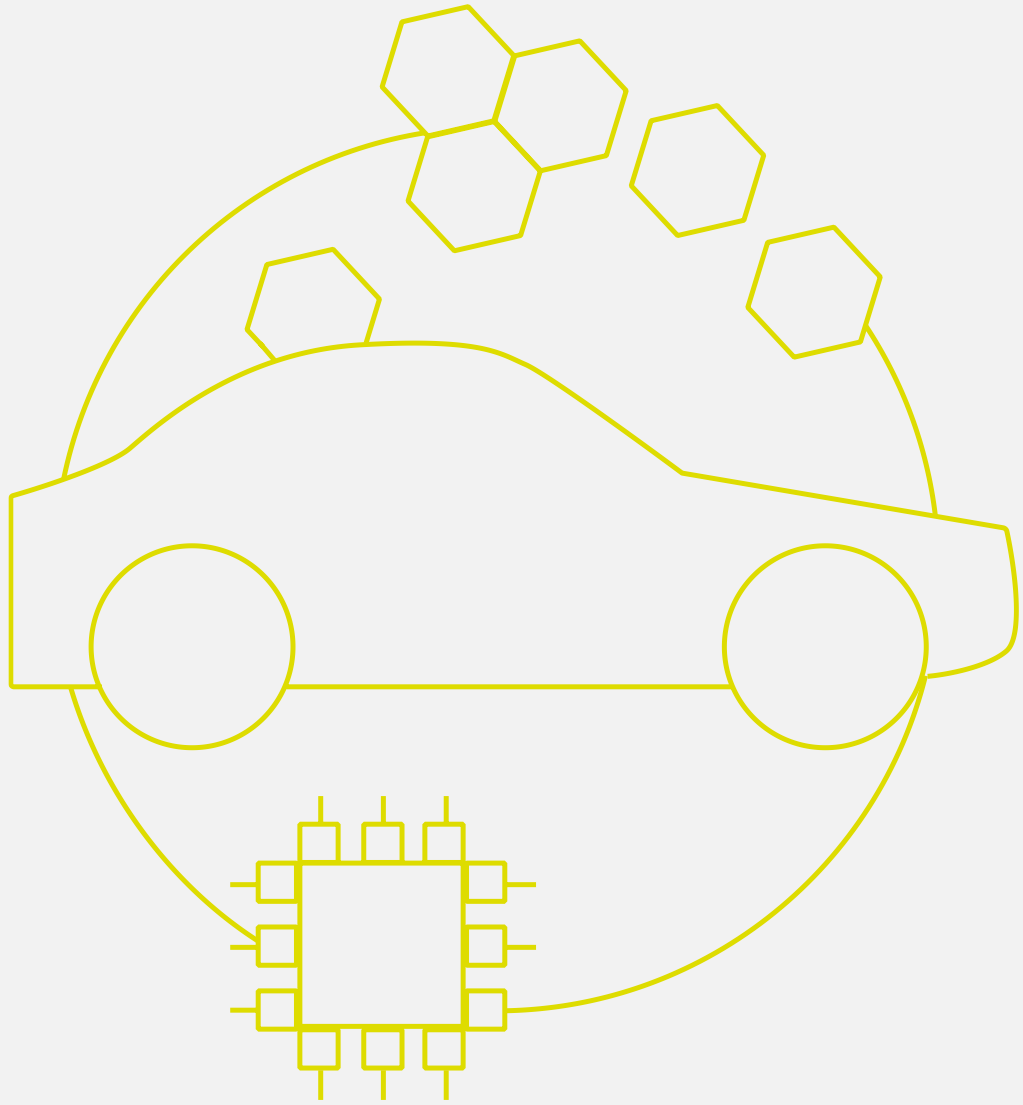
ASIL

QM

11

SOAFEE

3. Changes & Challenges brought by SDV



Changes brought by SDV to the automotive industry

Difference reduced between HW & SW development

More standardized development reduces the complexity of HW and SW integration

HW & SW decoupling

Under the SDV concept, OTA is given more attention, promoting the decoupling of HW and SW.

New biz models

Automotive industry is no longer just selling HW but bringing new profits to OEMs by providing services.

Challenges brought by SDV to the automotive industry

Architectural design

In order to achieve rapid development and iteration, it is necessary to design a **multi-modular automotive SW architecture with low correlation between modules.**

Functional safety

For SW upgrades, it is necessary to **test and verify functional modules with different safety requirements** to ensure safety.

Information Security

Information protection and control technology are required.
For example, verification of data sources, and verification of data correctness and timeliness.

SDV has had a significant impact on the development of automobiles, posing multiple challenges.

Shift in Automotive Supply Chain

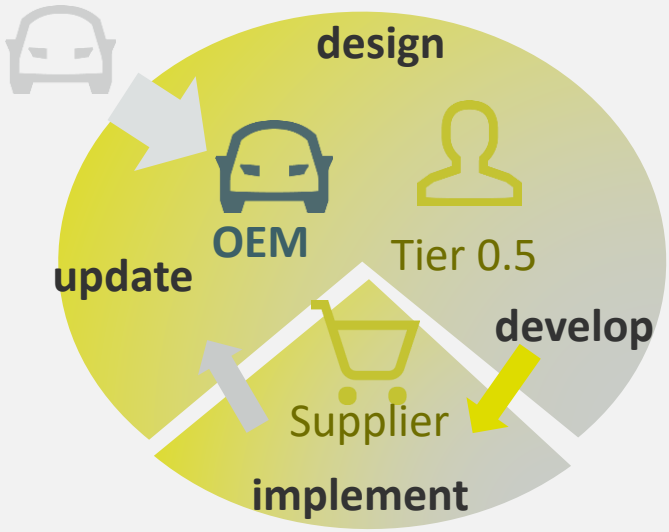
SDV pushes the automotive industry to shift towards Software Centric “Tier 0.5” approach

Manufacturing-Centric

- → OEM: detailed specs for manufacturing
- → Tier 1: all product development and system integration

Shift

Software-Centric

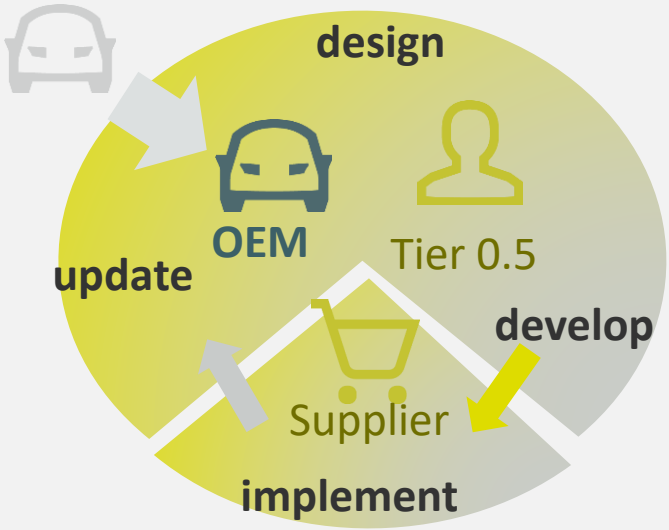


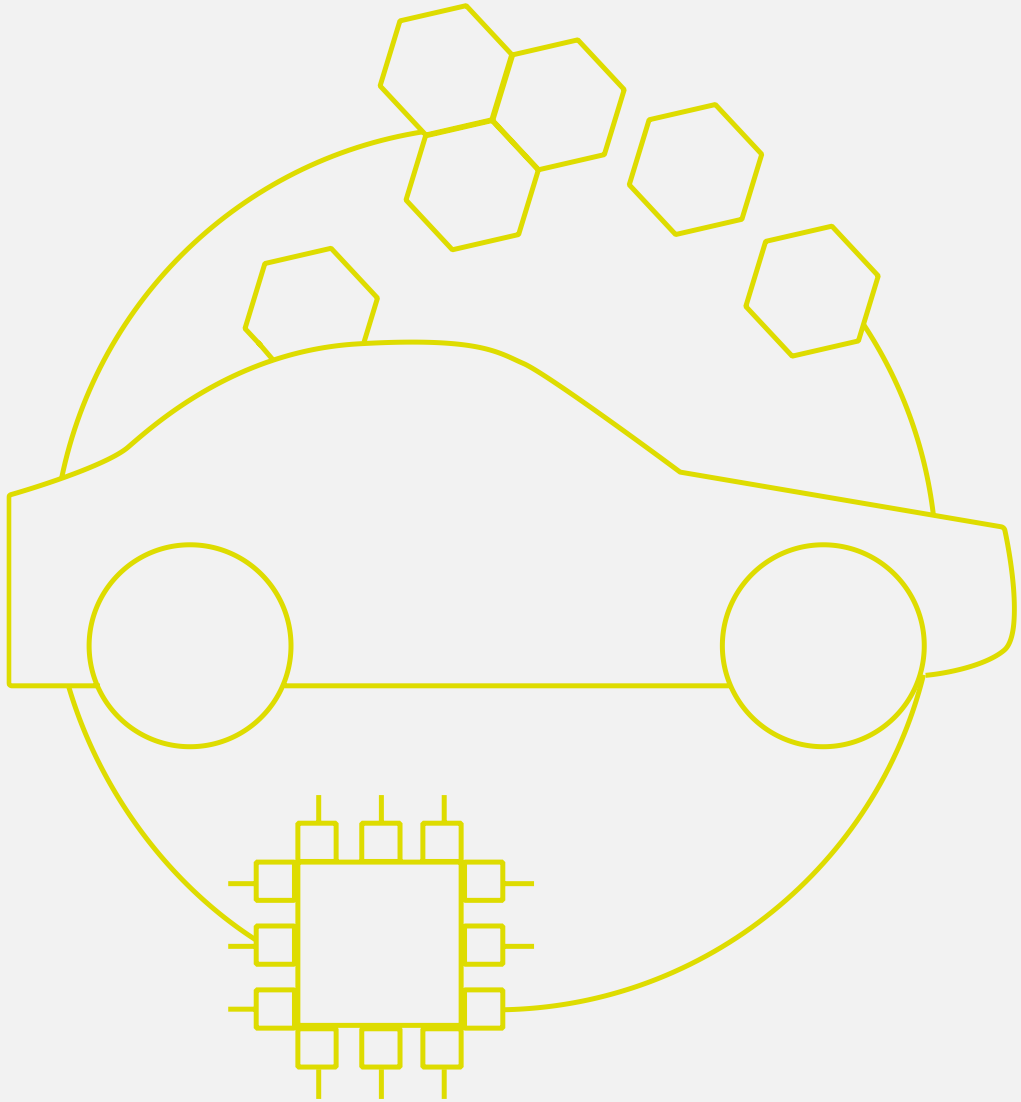
- → OEM: source “Tier 0.5” partners as domain experts for co-design & development
- → Long-term development & operational revenues

Preview - The Software-Defined Vehicle (sbdautomotive.com)

SOAFEE

4. Cloud-native System Design Approach



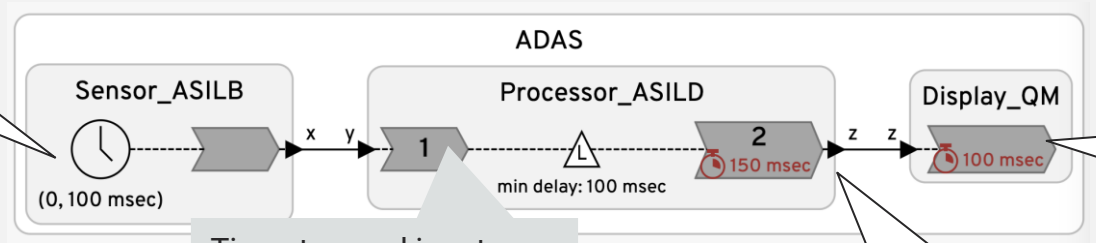


Lingua Franca

An actor-based synchronous reactive programming paradigm with a logical model of time

→ System Modeling
- Modeling software as reusable components

→ Deterministic scheduling
- Provide a runtime that enables efficient deterministic concurrency
- Support deadline-based error detection



Periodic sensor input

Deadline handler invoked if expected event not triggered within 100 msec

```
reactor Sensor_ASILB{
        timer t(0, 100 msec);
        output x:int;
        state count:int=0;
        reaction(t) -> x {=
                lf_set(x, self->count);
                self->count++;
        =}
}
```

Timestamped inputs

Logically instantaneous outputs

Local state variables

Reaction describing time encoded event behavior

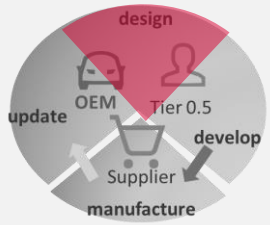
Target language code

Factoring in WCET of this **workload**

```
main reactor {
        sensor = new Sensor_ASILB();
        processor = new Processor_ASILD();
        ivi = new Display_QM();
        sensor.x -> processor.y;
        processor.z -> ivi.z;
}
```

Lingua Franca semantics allow us to model and develop deterministic application code

Design: System Modeling



- Model **AS** Code

vs

✓ Model **IS** Code

Templating Language YAML/JSON/TOML - k8s,CloudFormation

→ Pros
- Easy to read for human

→ Cons
- Too complex to be used for production-grade manifests

```
application:
  name: Sensor
  asil: B
  output:
    name: x
    type: int
    targetPort: Processor.y
 --
application:
  name: Processor
  input:
   name: y
```

Programming language/DSL - CDK8s, Lingua Franca

→ Pros
- More readable and production-grade manageable

→ Cons
- learning curve is steeper

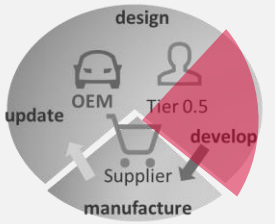
IDL

```
reactor Sensor_ASILB{
        timer t(0, 100 msec);
        output x:int;
}
```

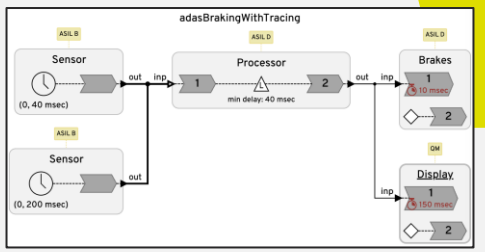
wiring

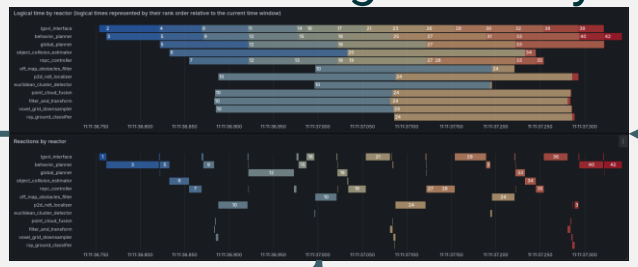
```
main reactor {
        sensor = new Sensor_ASILB();
        processor = new Processor_ASILD();
        sensor.x -> processor.y;
}
```


Develop: System Development and Verification



- → Timing analysis in a V-model development process
- → Cloud-native design development
- → Observability
  - helps you to determine if application is functioning normally





timing-analysis and assessment

Criteria and limits

System Design

System Specification

Component Specification

Component Design

Functional requirement and boundary conditions

Test cases and test criteria

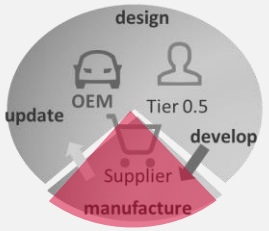
Integration Test Vehicle

Integration Test Component

Implementation

Recommended Methods and Practices for Timing Analysis and Design within the AUTOSAR Development Process

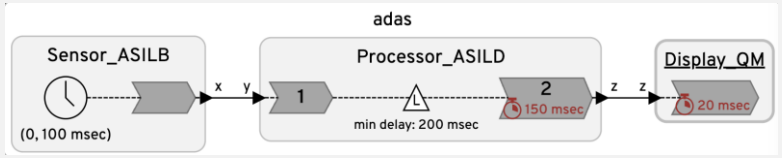
Implement: Deterministic Scheduling

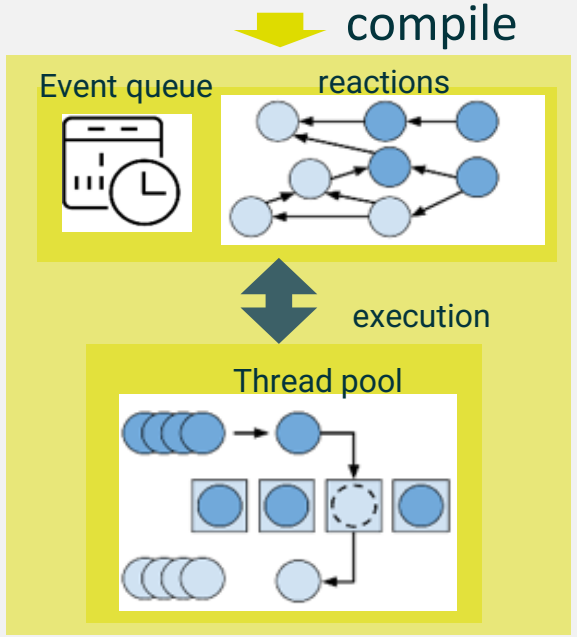


A runtime with determinism and parallelism enable

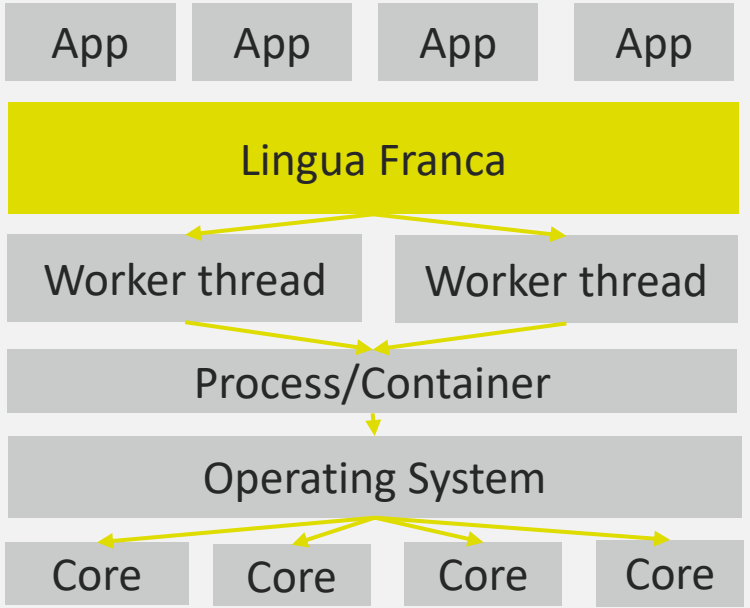
→ Determinism
- Components inform the scheduler at what logical time to trigger reactions

→ Parallelism
- The runtime exploits parallelism by the dependencies between reactions in the dependency graph





Control event flow through scheduling algorithms

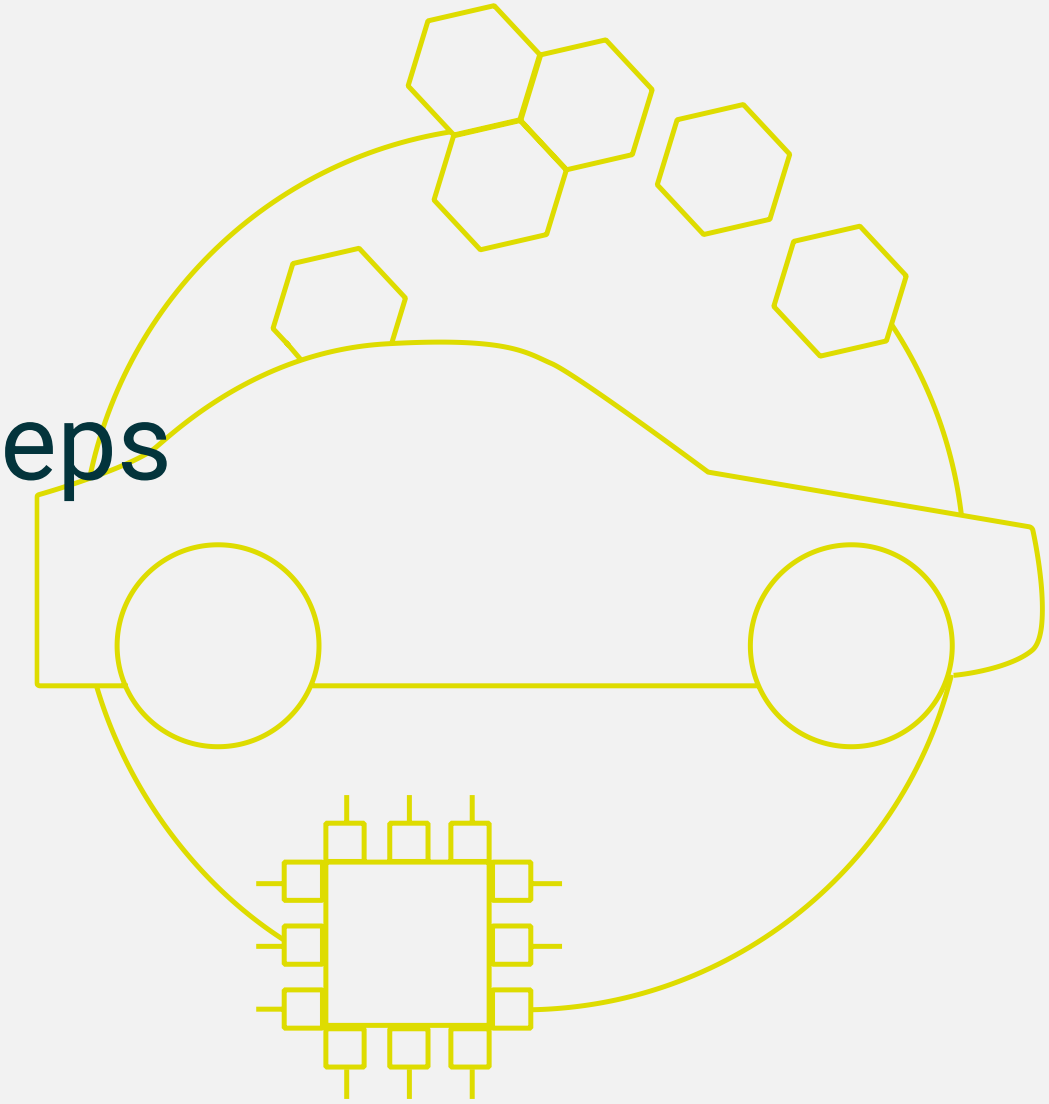


Demo of Automated Valet Parking using LF

→ Blueprint submitted to SOAFEE (to be released soon)

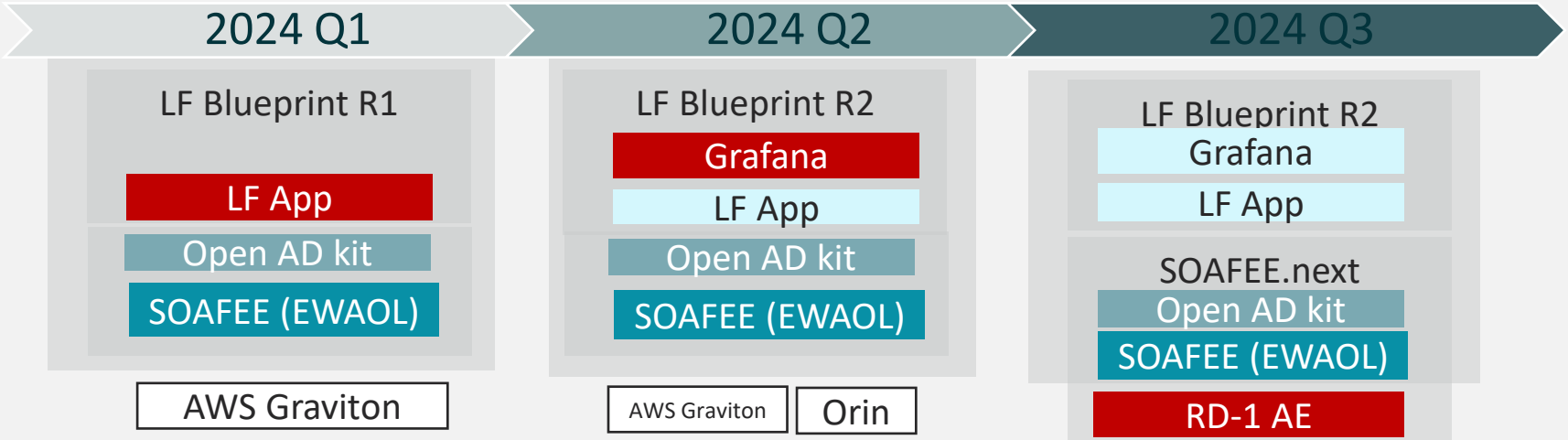
SOAFEE

4. Summary and Next Steps



Conclusions and Future Works

- Summary
  - Software-oriented E/E Architecture enables the future of mobility
  - We demonstrated LF as a mixed critical orchestrator solution on SOAFEE reference architecture using AVP
- Next steps
  - Proposal to MCO requirement
  - Integration of LF blueprint with SOAFEE.next



Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה

Automated Valet Parking: Problems and Approach

Automated Valet Parking

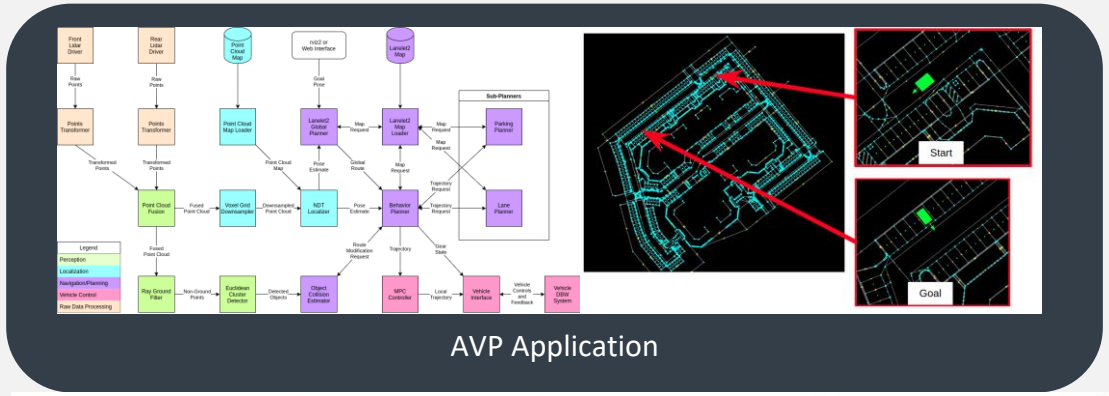
- AD application to autonomously park and return to a pick-up/drop-off area in a parking lot
- Autoware Foundation provided blueprint to show how such a service can be integrated with SOAFEE SDV reference architecture

Problems

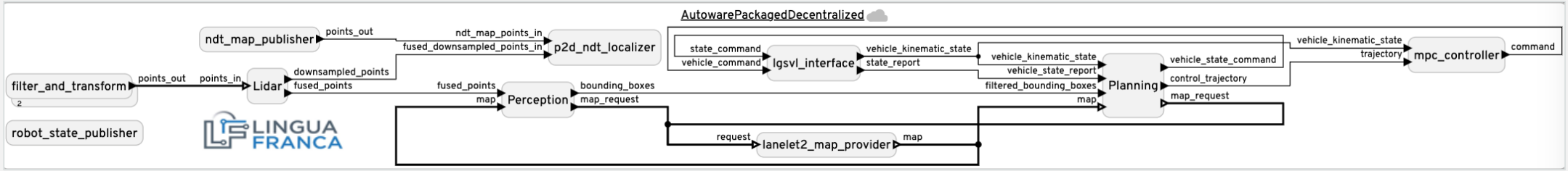
- Non-deterministic behavior (Eg: unresponsiveness, jitteriness, etc.) on SDV platform

Approach

- LF enforced deterministic scheduling to suppress observed issues in original demo



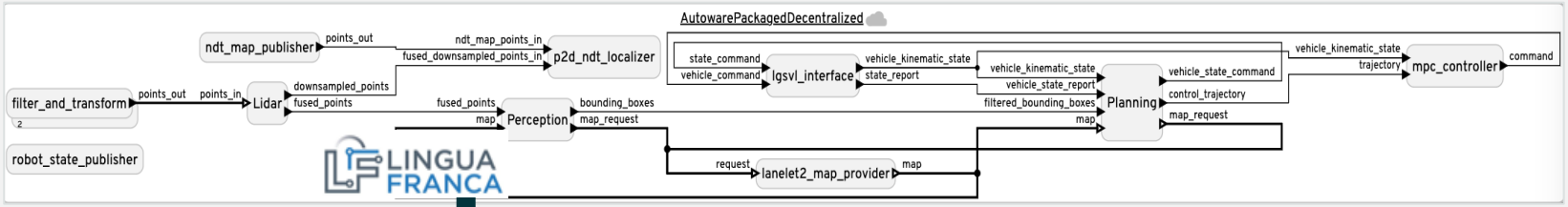
LF system modeling of AVP application



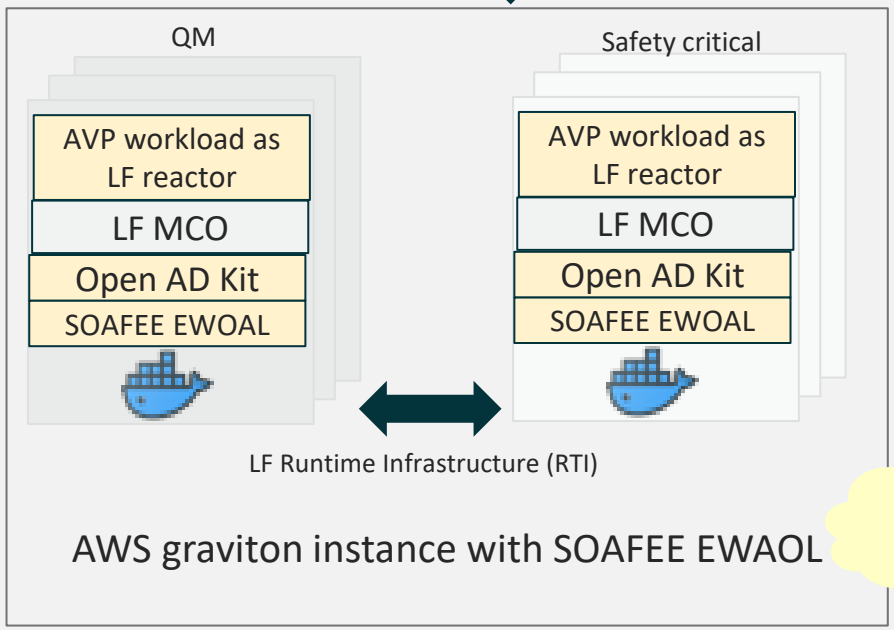
Integrated LF and Open AD Kit application on SDV

Demonstrate LF as a mixed critical orchestrator solution on SOAFEE reference architecture using AVP

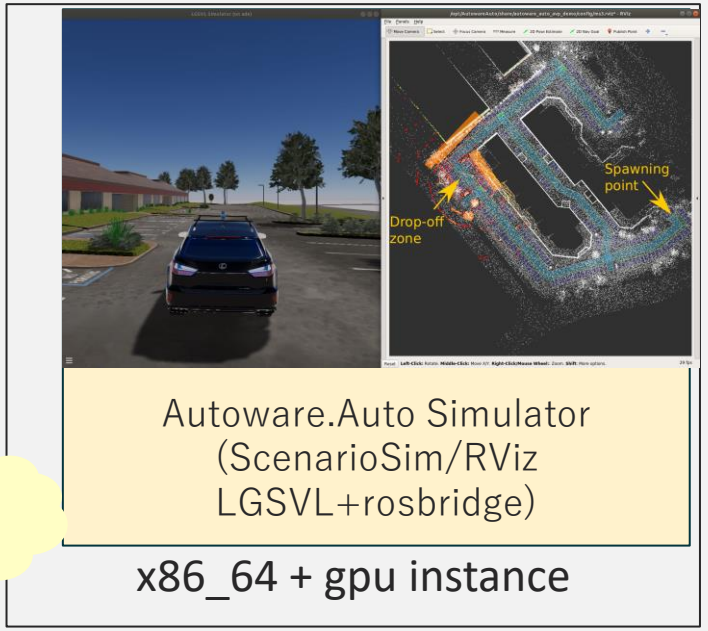
- LF Mixed Critical Orchestrator (MCO) manages the scheduling across containerized workloads
- Porting ROS2 nodes to LF
- In current configuration, safety critical and QM containers run on virtual High Performance Compute (HPC)
- The default Autoware simulator LGSVL is used



Deploy generated code as containers on cloud



Docker Swarm /VLAN



*Evaluation on mixed criticality hardware setup is the next step.
Testbed: NVIDIA Orin (as HPC) + (R-car S4 as Safety Island)