SOAFEE

System Design and Verification for Mixed Critical Systems in SDV

SOAFEE APAC Seminar

Akihito Iwai
Software Production Innovation Div.
DENSO CORPORATION

21st September, 2023

© 2023 SOAFEE

Agenda

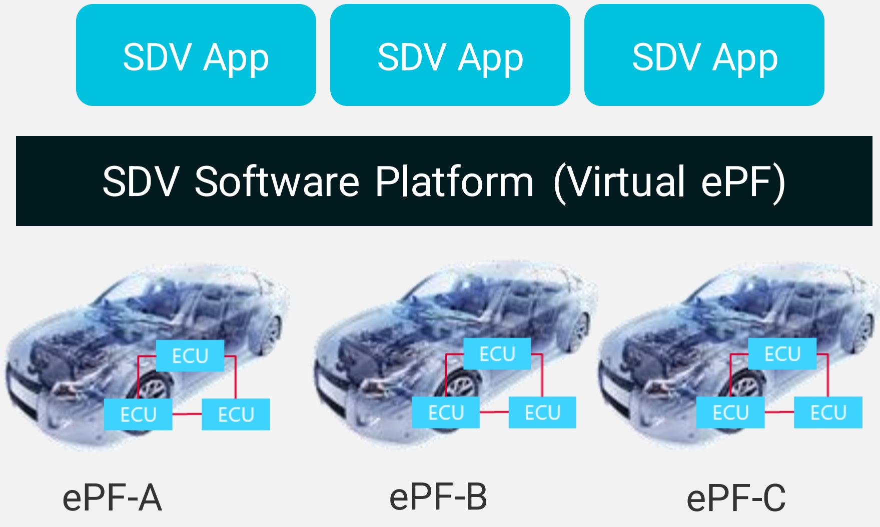
1. What is SDV (Software Defined Vehicle) ?
2. Cloud Native Development
3. Challenges
4. Need for Standardization
5. Mixed Criticality
6. Enabling Tech Candidate: Lingua Franca
7. Cloud Native Development with Lingua Franca
8. Demo
9. Conclusion

© 2023 SOAFEE

SDV (Software Defined Vehicle)

- The concept and mechanism of **abstracting vehicle hardware** (electronic PF -> Virtual ePF))
  - ECUs, in-vehicle networks, sensors, and actuators with **virtualization technology**
- **Software controlling** these computer resources
- In other words, "How to **separate apps, software, and hardware**"

Concept structure of SDV system



SDV App

SDV App

SDV App

SDV Software Platform (Virtual ePF)

ePF-A

ePF-B

ePF-C

Objectives:

- Reduce development costs
- Provide new value to customers

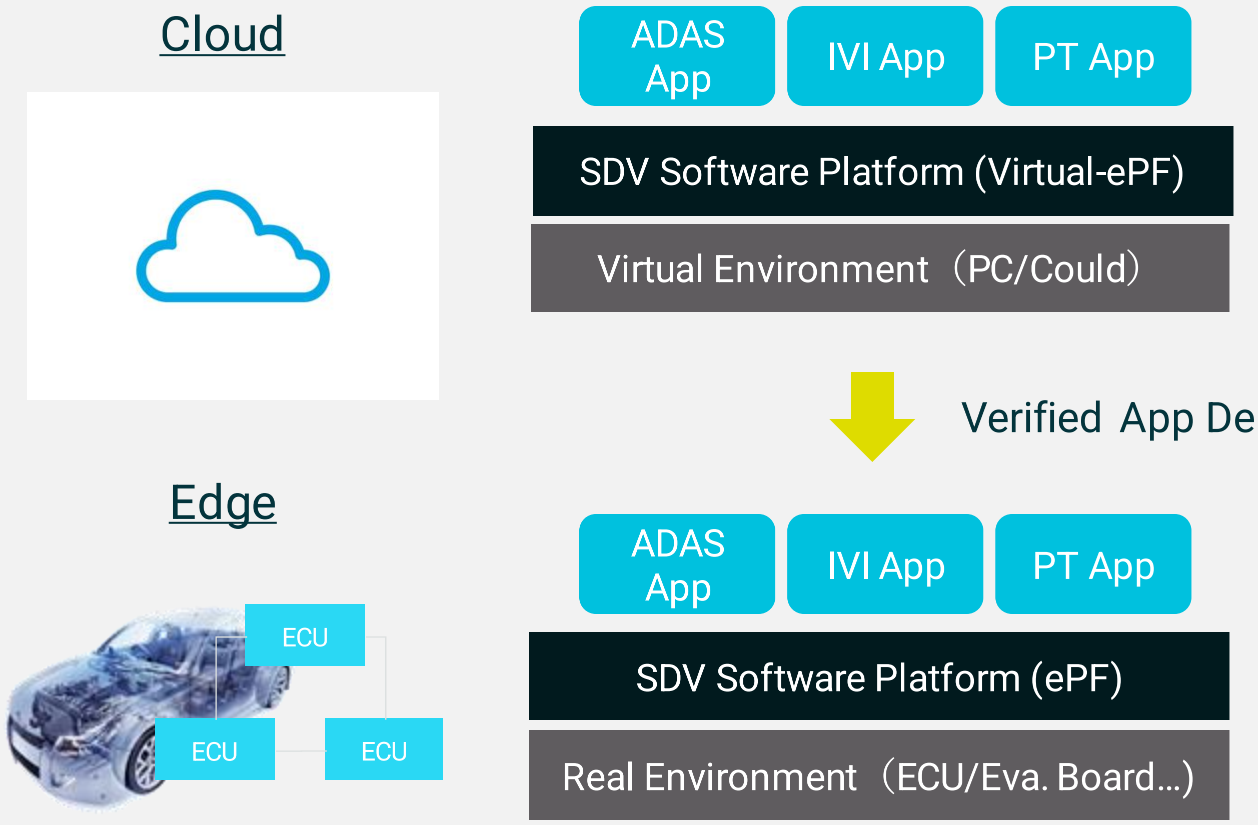
New Requirements:

- Cross-domain applications
- Mixed Criticality*
- Cost-effective scalability
- On demand software upgrade

**Mixed Criticality:*
Integrating components with different levels of safety criticality

Vehicle Virtualization is a key technical challenge!

Cloud Native Development



Cloud

ADAS App | IVI App | PT App

SDV Software Platform (Virtual-ePF)

Virtual Environment（PC/Could)

Verified App Deployment

Edge

ADAS App | IVI App | PT App

SDV Software Platform (ePF)

Real Environment（ECU/Eva. Board…)

System Design and Verification on Virtual Environment

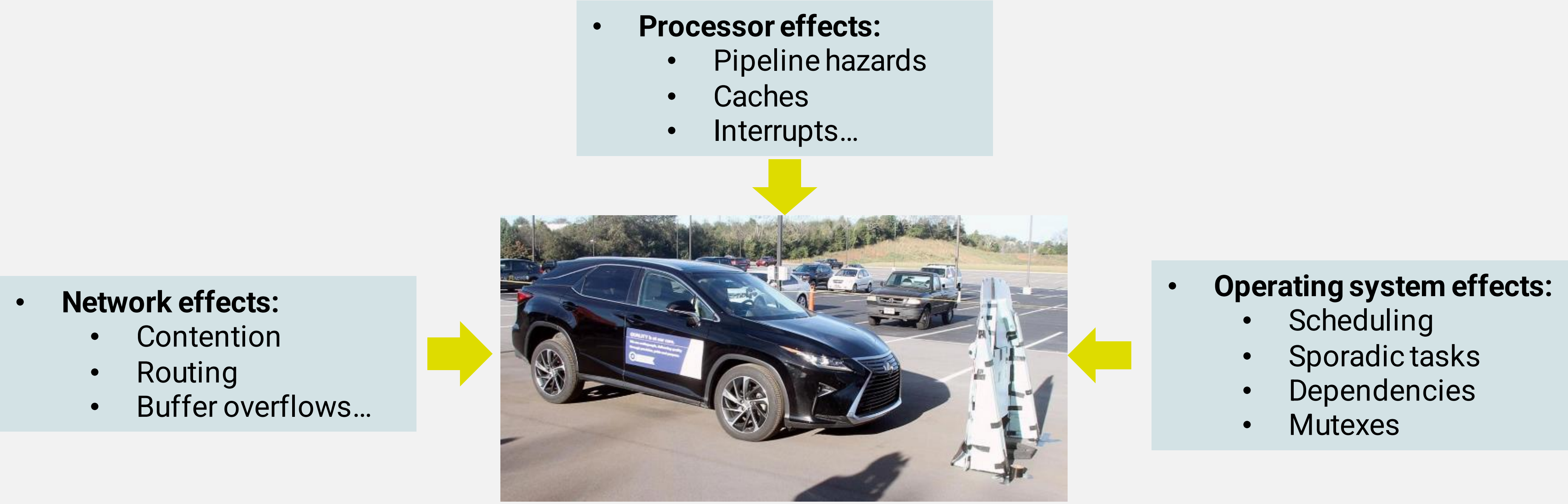
Validation and Evaluation on Real Environment

Utilize virtual environments to reduce system development time

© 2023 SOAFEE

Challenges

- **Handling of application runtime behavior** (due to execution times, network latency, etc.) in cloud native SDV environment
- Mixed critical workload orchestration **with time-critical event handling**
- **Satisfy non-functional requirements** (Repeatability, testability, reliability, etc.)



- **Processor effects:**
  - Pipeline hazards
  - Caches
  - Interrupts…

- **Network effects:**
  - Contention
  - Routing
  - Buffer overflows…

- **Operating system effects:**
  - Scheduling
  - Sporadic tasks
  - Dependencies
  - Mutexes

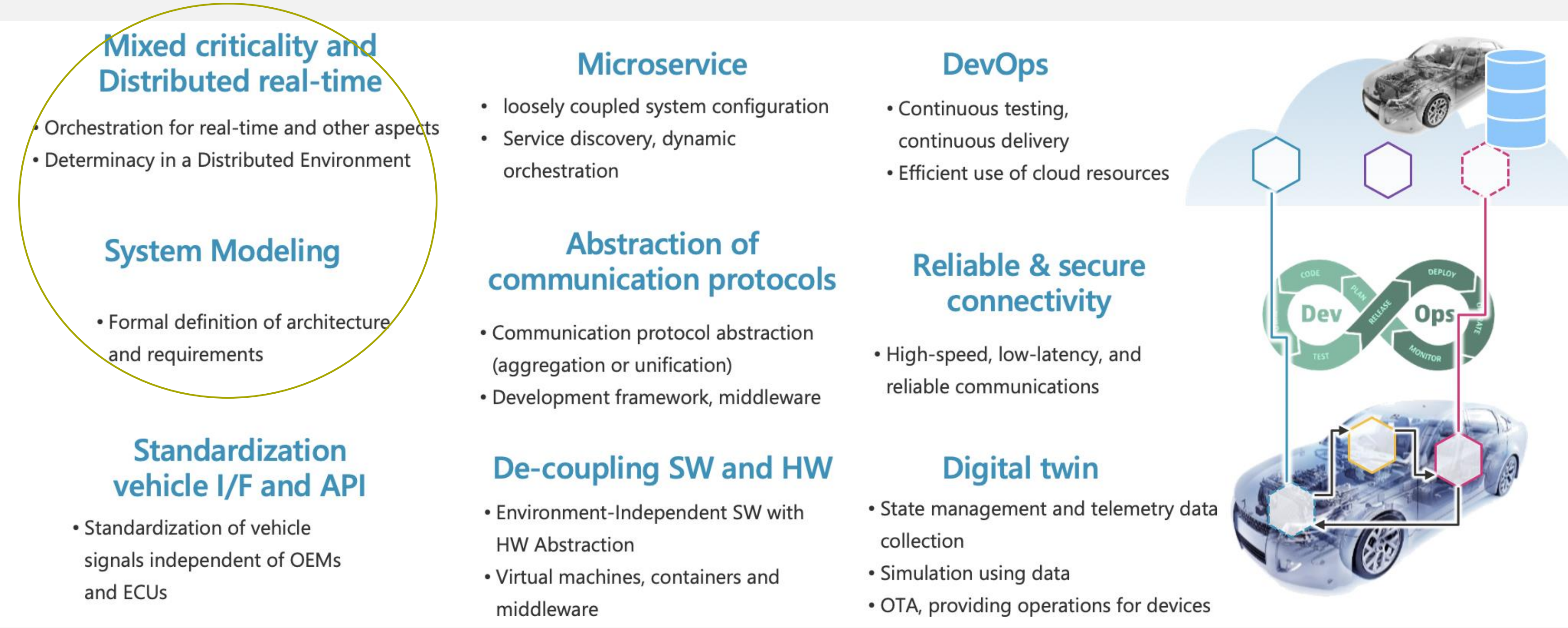
Hard to model SDV system behavior deterministically !

© 2023 SOAFEE

Need for Standardization

- **Wide-ranged SDV domain** cannot be solved by one company alone
- Consortiums to develop **common standards and technology**
- **Accelerate SDV development** through active participation in such consortia

Areas where DENSO could contribute



Mixed criticality and Distributed real-time

- Orchestration for real-time and other aspects
- Determinacy in a Distributed Environment

System Modeling

- Formal definition of architecture and requirements

Standardization vehicle I/F and API

- Standardization of vehicle signals independent of OEMs and ECUs

Microservice

- loosely coupled system configuration
- Service discovery, dynamic orchestration

Abstraction of communication protocols

- Communication protocol abstraction (aggregation or unification)
- Development framework, middleware

De-coupling SW and HW

- Environment-Independent SW with HW Abstraction
- Virtual machines, containers and middleware

DevOps

- Continuous testing, continuous delivery
- Efficient use of cloud resources

Reliable & secure connectivity

- High-speed, low-latency, and reliable communications

Digital twin

- State management and telemetry data collection
- Simulation using data
- OTA, providing operations for devices

DENSO proposal to MCO* team based on our core capabilities

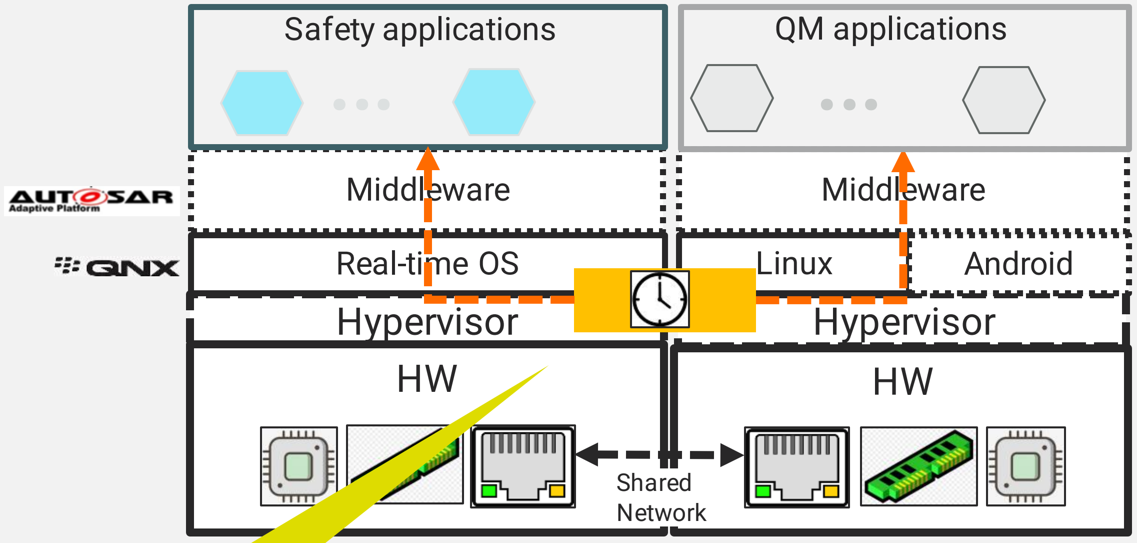
**Mixed Criticality Orchestrator*

© 2023 SOAFEE

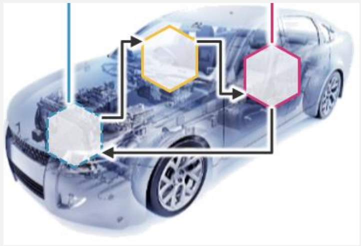
Mixed Criticality (MC)

Inter-dependence among certified and non-certified SW and HW components (Functional Safety)

- Mixed Criticality is an important problem to solve for SDV
  - Unify approaches for development and runtime execution of safety and quality managed (QM) processes
  - Transition to evolving ECU architectures (Isolated Domain controllers → ECU consolidation)



Application specific process to handle MC



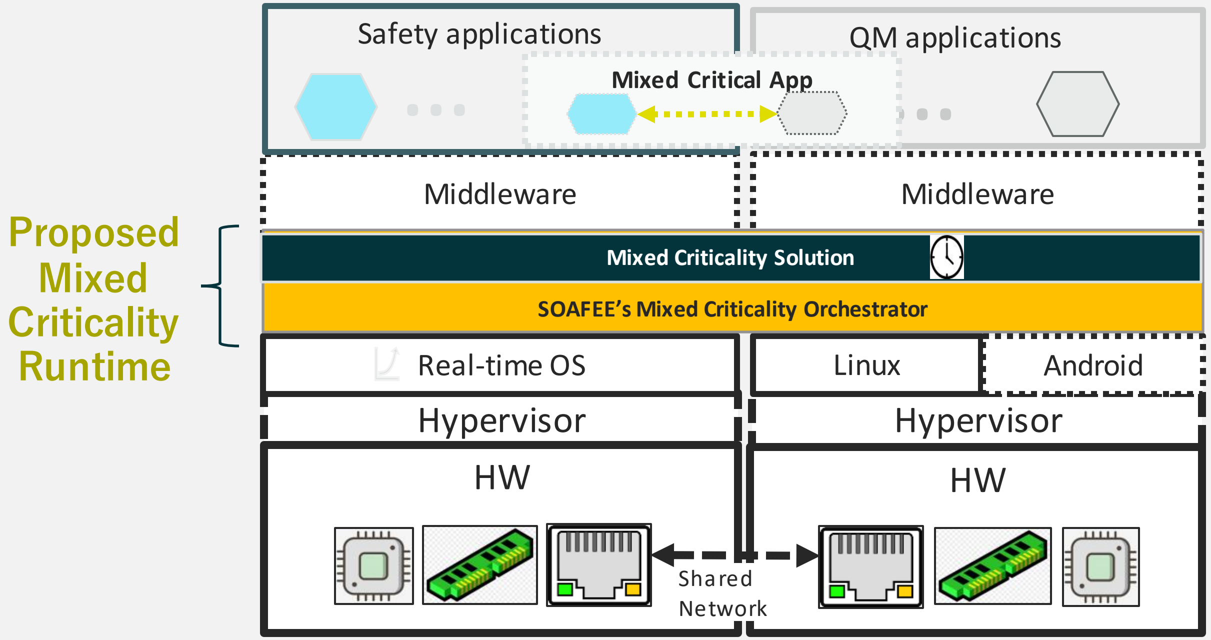
Criticality aware design and development could be provisioned containerized and microservices in-vehicle

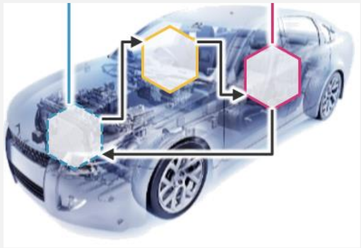
© 2023 SOAFEE

Mixed Criticality (MC)

- **SOAFEE's Mixed Criticality orchestrator concept**
  - Hardware abstractions for criticality agnostic application interface
  - Advanced virtualization methods involving resource management

- **DENSO's Mixed Criticality solution:**
  - Provides an **application-level safety** envelope for handling uncertainties
  - **Deterministic scheduling** methods for handling real-time requirements at the application interface
  - **Safety violations** detected at runtime (and compile time)





The combination of the two concept is key to the realization of MC applications

© 2023 SOAFEE

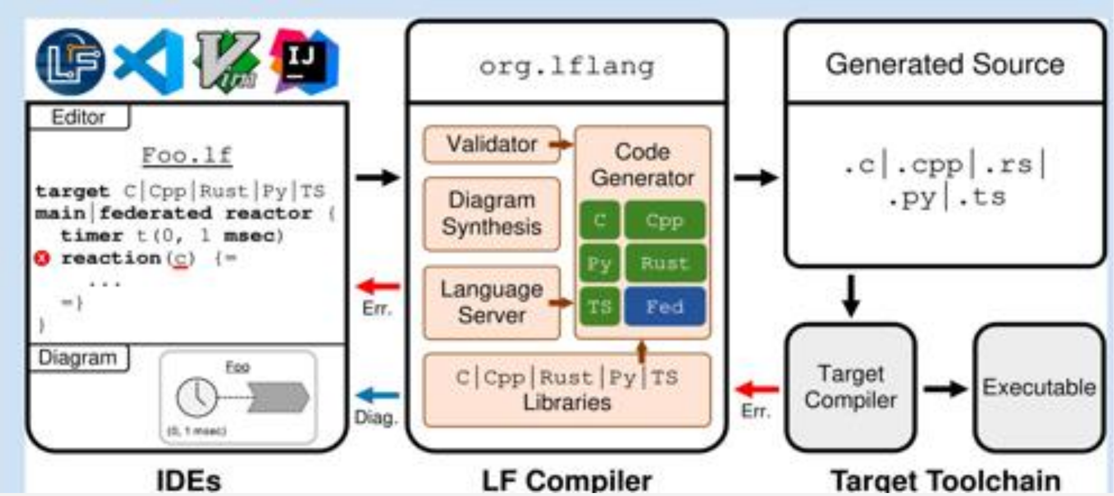
Enabling Tech Candidate: Lingua Franca

Lingua Franca (LF) is a polyglot coordination language for reactive, concurrent, and time-sensitive applications.

- **Open Source Project** developed by UC Berkeley
  - https://www.lf-lang.org/
  - https://github.com/lf-lang/lingua-franca

- **Main Features:**
  - Handles application data flow complexity
    - **Distributed** Cyber-Physical System
    - **Dynamic** software components
  - Guarantees precision time coordination
    - **Time encoded** specification
    - **Distributed event scheduler** for various communication patterns

Collaborator:
Prof. Edward Lee



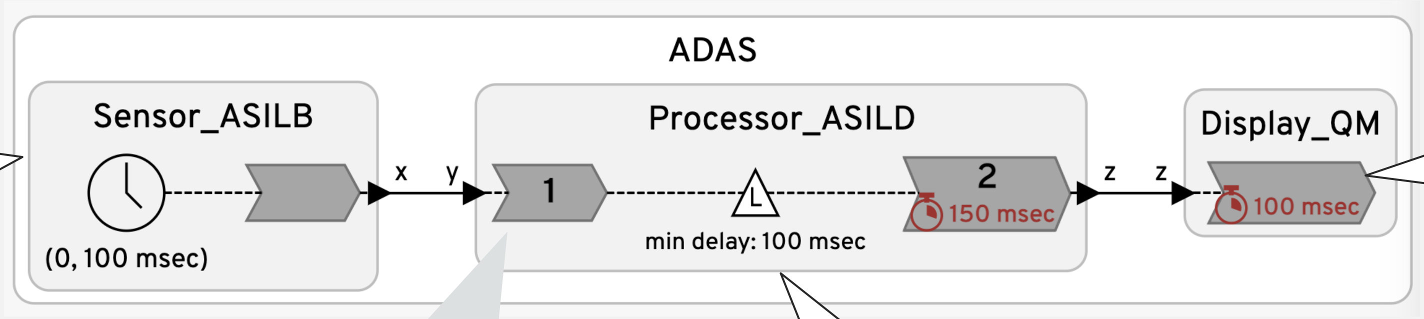
```
target C|Cpp|Rust|Py|TS
main|federated reactor {
  timer t(0, 1 msec)
  reaction(c) {=
    ...
  =}
}
```

Integrating complex subsystems with adequate reliability, repeatability, and testability

© 2023 SOAFEE

Brief Overview of LF

- **Reactor** represents a concrete functional block that is **time encoded**
- **Compositionality** used to build **data flow** in the system



Periodic sensor input

Deadline handler invoked if expected event not triggered within 100 msec

Timestamped inputs

Factoring in WCET of this **workload**

```
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++;
        =}
}
```

Logically instantaneous outputs

Local state variables

Reaction describing time encoded event behavior

Target language code

```
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

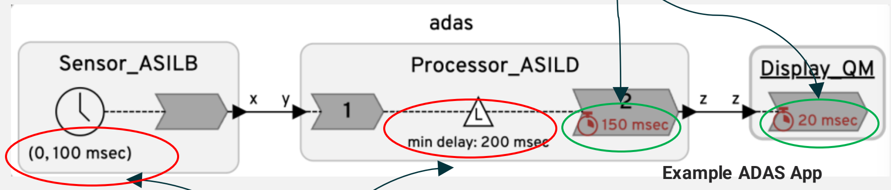
Cloud Native Development with Lingua Franca

System modeling: Encode timing properties based on safety criteria of each component

Generated Target Code

Application functional requirements
(App designer)

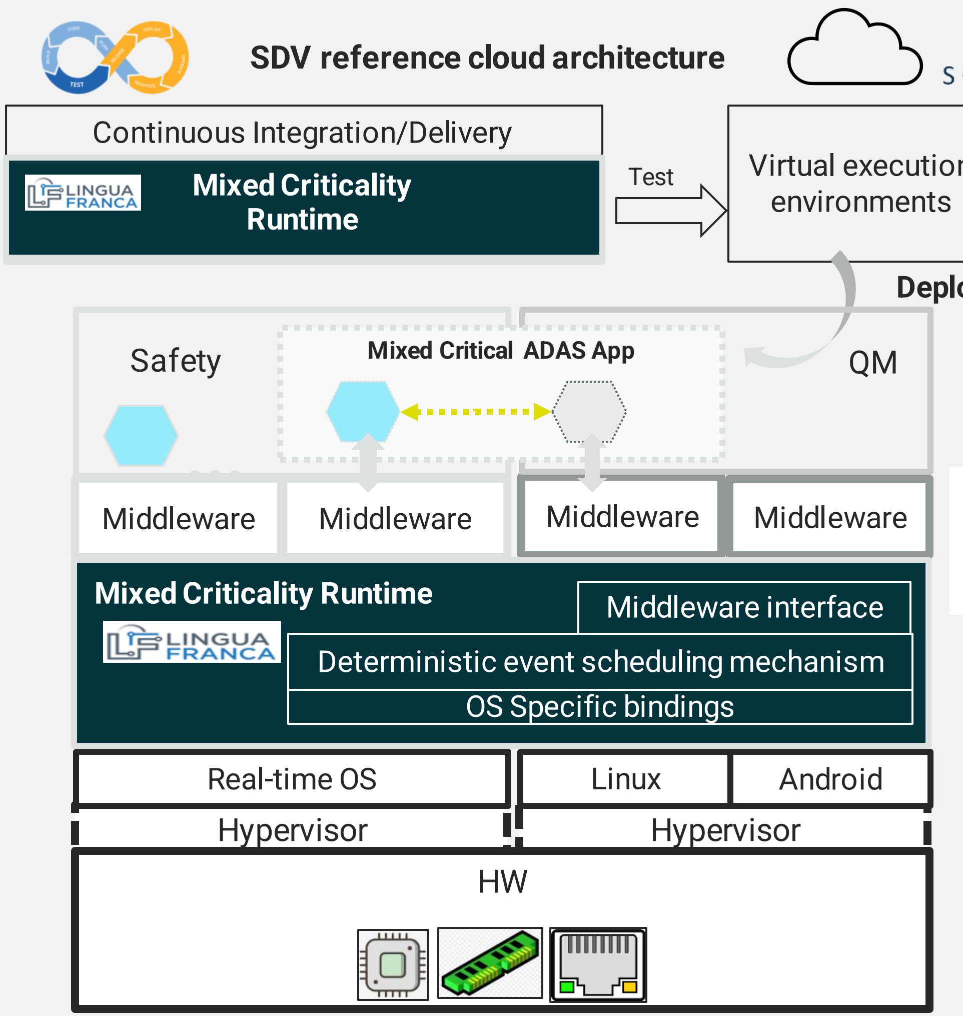
Response times, schedules, etc.



Example ADAS App

WCET* based on HW resource constraints, periodic behavior, etc.

System design
(System architect)



SDV reference cloud architecture

SOAFEE

Continuous Integration/Delivery

Mixed Criticality Runtime

Test

Virtual execution environments

Deploy Validated applications

Safety

Mixed Critical ADAS App

QM

Middleware

Middleware

Middleware

Middleware

Mixed Criticality Runtime

Middleware interface

Deterministic event scheduling mechanism

OS Specific bindings

Real-time OS

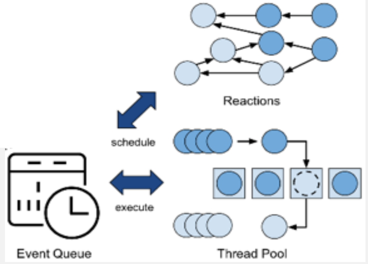
Linux

Android

Hypervisor

Hypervisor

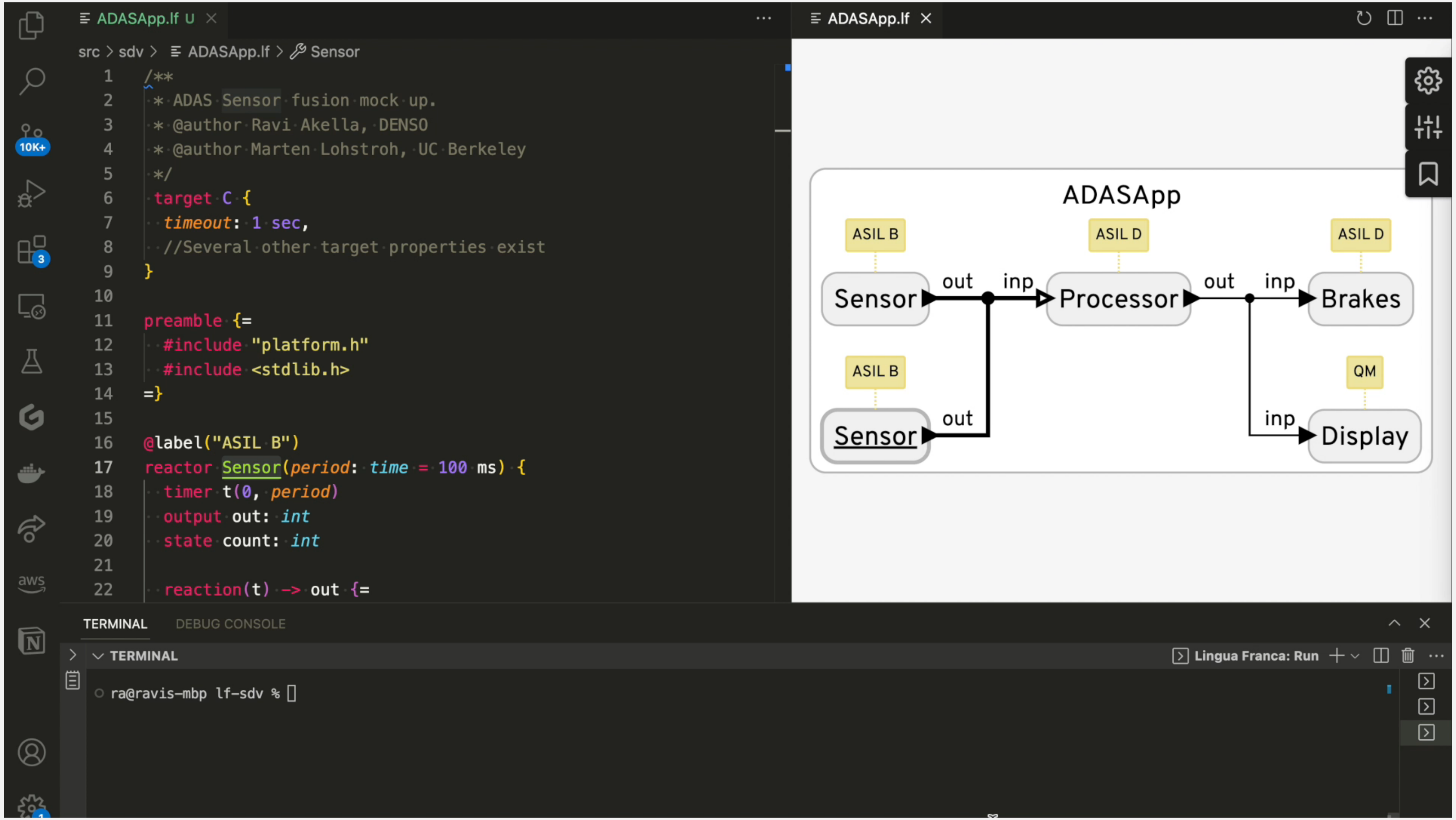
HW



Control event flow through scheduling algorithms

Mixed Critical run-time: Detect runtime violations of specified properties and invoke fault handler dynamically

DEMO



```
/**
 * ADAS Sensor fusion mock up.
 * @author Ravi Akella, DENSO
 * @author Marten Lohstroh, UC Berkeley
 */
target C {
  timeout: 1 sec,
  //Several other target properties exist
}

preamble {=
  #include "platform.h"
  #include <stdlib.h>
=}

@label("ASIL B")
reactor Sensor(period: time = 100 ms) {
  timer t(0, period)
  output out: int
  state count: int

  reaction(t) -> out {=
```

```
ra@ravis-mbp lf-sdv % 
```

© 2023 SOAFEE

Conclusion

→ Our SDV activity focuses on System modeling for Mixed critical applications

→ We are working with SOAFEE Mixed Critical Orchestrator working group

→ Proposing Lingua Franca as an essential solution for realizing “mixed critical orchestrator”

→ We are looking for application scenarios for blueprint submission

© 2023 SOAFEE

SOAFEE

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

© 2023 SOAFEE