Linaro Connect
MADRID 2024 | MAY 14-17 2024

DENSO's Approach for Mixed Critical Systems in SDV

Motohiro Shibakawa
DENSO AUTOMOTIVE Deutschland GmbH

Agenda

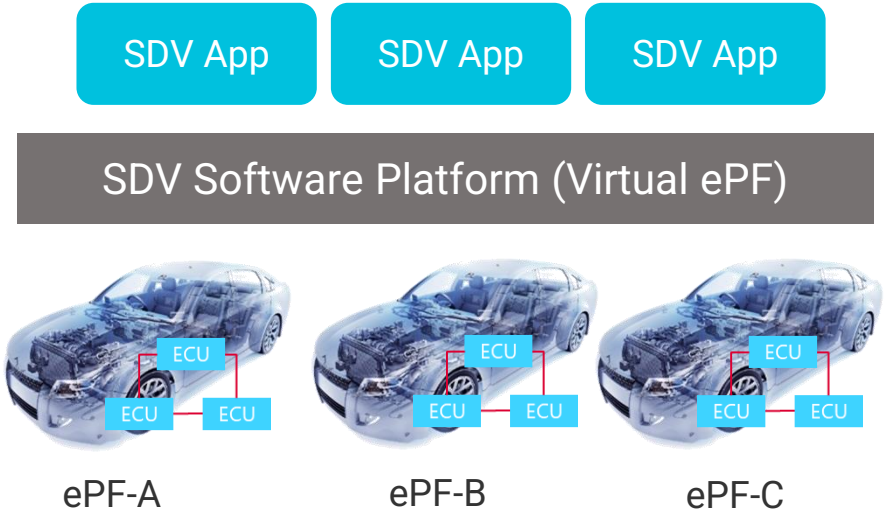
- What is Software Defined Vehicle(SDV)?
  - Software Defined Vehicle(SDV)
  - Mixed Criticality System
  - Technical Challenges
- SDV technical areas and open community
  - Technical areas required for SDV
  - SOAFEE
- DENSO’s Approach for Mixed Critical Systems
  - Overview of DENSO’s Approach
  - Proposal for Mixed Criticality (MC)
  - Enabling Tech Candidate: Lingua Franca
  - Brief Overview of LinguaFranca
  - Demo video
  - Automated Valet Parking: Problems and Approach
  - Integration with systems engineering
- Conclusion

What is Software Defined Vehicle(SDV)?

SDV (Software Defined Vehicle)

- SDV stands for Software Defined Vehicle, which means "a car defined by software.".
- In the past, cars improved their performance by improving the hardware centered around the combustion engines, but in the future, the software in the car will determine the value of the car.

Concept structure of SDV



- The concept and mechanism of **abstracting vehicle hardware**
  - ECUs, in-vehicle networks, sensors, and actuators with virtualization technology
- **Software controlling** these vehicle resources
- In other words, "How to **separate apps, platform, and hardware**"

The development/production of vehicle has shifted from hardware centric to software centric

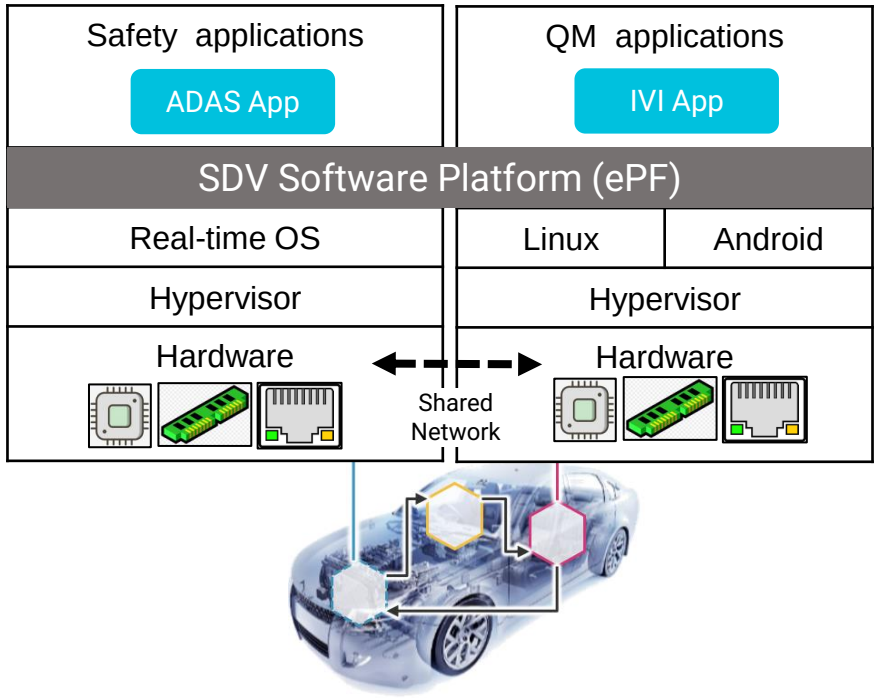
Mixed Criticality System

System Integrating components with different levels of safety criticality

Mixed Criticality will be a fundamental requirement for SDV

- Evolution of the ECU architecture makes it likely that applications will interfere with each other
  (Isolated ECU -> Domain ECU -> Zonal ECU)
- Hardware abstraction brings a unified design, and verification approach and platform for various applications
- New functions are expected from cross-network applications, such as working with the cloud, V2V, V2X, etc

Apps interacting and working as one system



Technical challenges of SDV

- Handling of MC system runtime behavior (due to execution times, network latency, etc.) on various hardware
- Mixed critical app orchestration with consideration of time-critical event
- Satisfy non-functional requirements (Repeatability, testability, reliability, etc.)

There is many uncertainties in the SDV

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

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

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

Guaranteeing SDV system deterministic behavior is crucial

SDV technical areas and open community

Technical areas required for SDV

- Wide-ranged SDV technical area cannot be solved by one company alone
- Consortiums to develop common standards and technology
- Accelerate SDV development through active participation in open 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

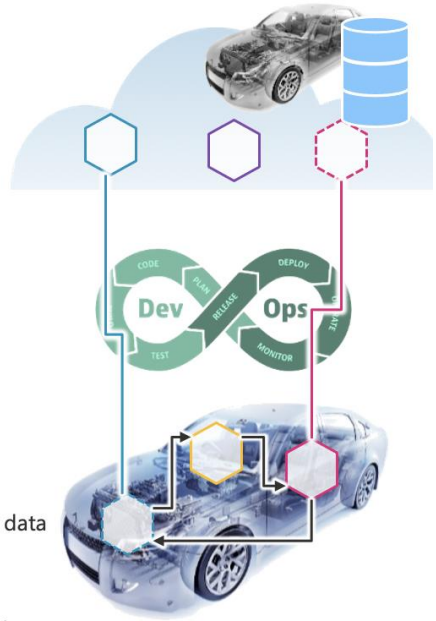
- Continuous testing, continuous delivery
- Efficient use of cloud resources

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

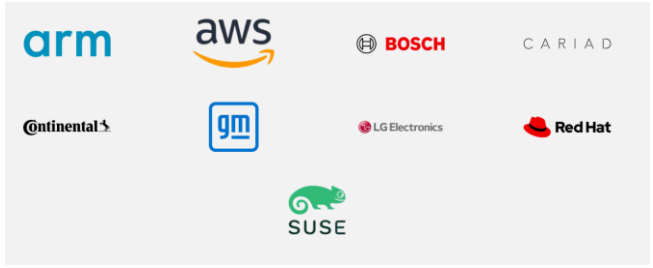


Need for Standardization

SOAFEE

- **S**calable **O**pen **A**rchitecture **F**or **E**mbedded **E**dge project
- Established in September 2021 led by Arm

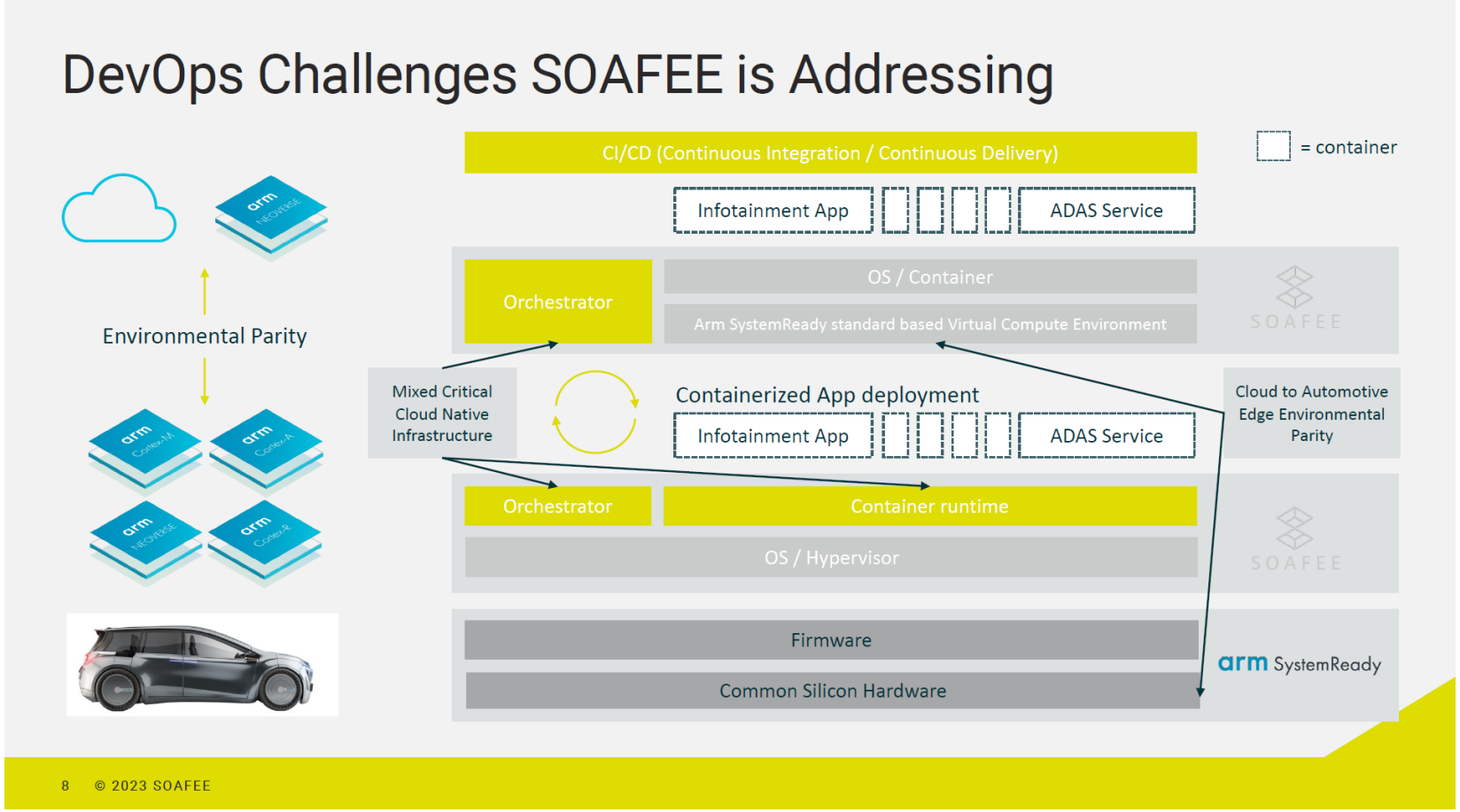
Governing Body Members: 9 companies
Voting Members: 108 companies
(as of September 2023)



Scope & Purpose of Activity

- Define a software architecture and reference implementation for **deployment of mixed critical system**
- A platform for seamless cloud-to-automotive edge software development, designed to **maximize environmental parity**.
- The creation and contribution to **industry standards** that support cloud-native development-

Challenges SOAFEE is working on



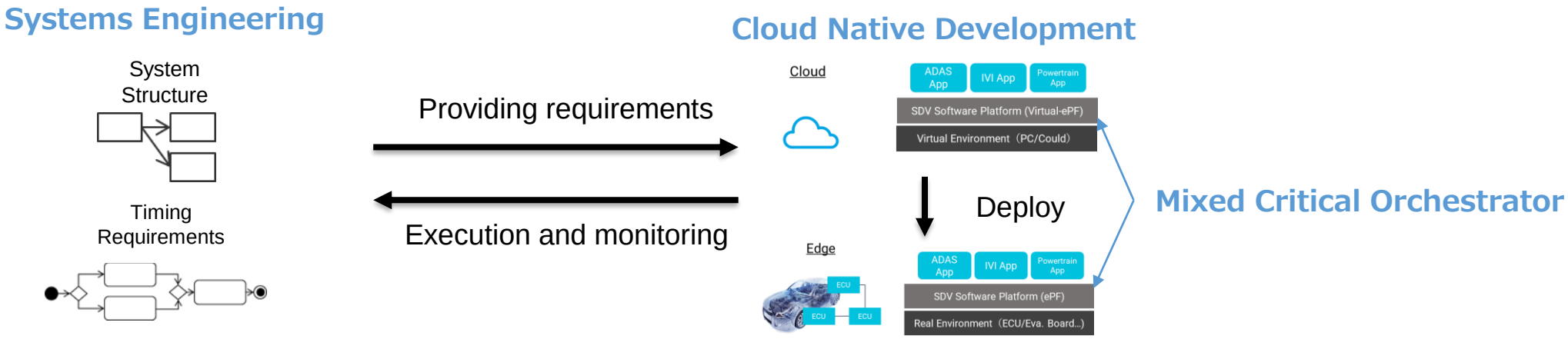
Aiming to develop a Mixed Criticality Aware Orchestrator

Source: https://www.soafee.io/files/soafee-seminar-open-and-arm-presentation.pdf

DENSO's Approach for Mixed Critical Systems

Overview of DENSO's Approach

1. **Systems Engineering:** Modeling safety criteria of each component using timing abstractions
2. **Mixed Critical Orchestrator:** Runtime enforcement of timing requirement through system modeling
    1. Real-time application monitoring to detection of deadline violations of distributed workload execution
3. **Cloud Native Development:** Life cycle management of application on virtual hardware
    - Functional validation, provisioning of cloud resources and deployment



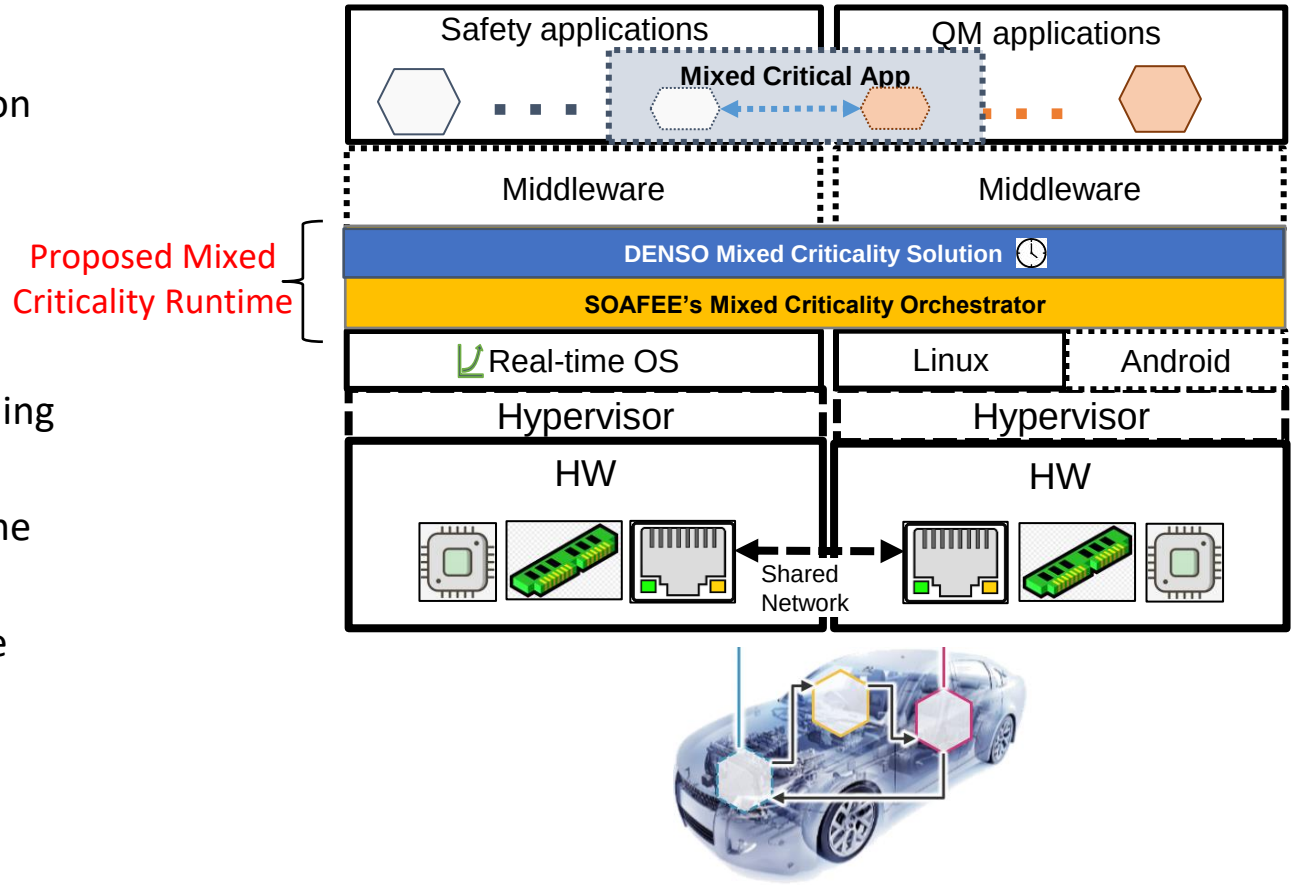
Mixed Critical Orchestrator

SOAFEE's Mixed Criticality aware orchestrator

- Hardware abstractions for criticality agnostic application
- Integration of IT industry orchestration tools into the automotive edge

DENSO's Mixed Criticality solution

- Provides an application-level safety envelope for handling uncertainties
- Deterministic scheduling methods for handling real-time requirements of the application
- Safety violations detected at runtime and compile time



The combination of these two technologies is key to the realization of Mixed Critical Orchestrator

Enabling technology: Lingua Franca

Lingua Franca is modeling language and runtime to enrich programing language with ability to specify timed behavior

Open Source Project developed by UC Berkeley

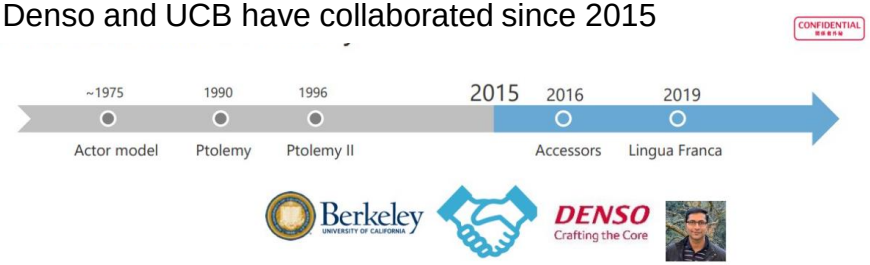
- https://www.lf-lang.org/
- https://github.com/lf-lang/lingua-franca

Collaborator:
Prof. Edward Lee

Main Features:

- **Modeling language for concurrent system** that ensure determinism, eliminating concerns about thread management, synchronization.
- **The scheduler automatically generated** from the model accurately handles time-sensitive tasks without the complex timing logic typically required in concurrent programming.

Denso and UCB have collaborated since 2015



LF integrates complex systems with reliability, repeatability, and testability

Brief overview of Lingua Franca

- **Reactor** represents a functional component that is **time encoded**
- **Deterministic scheduler** provides a runtime that enables deterministic concurrency

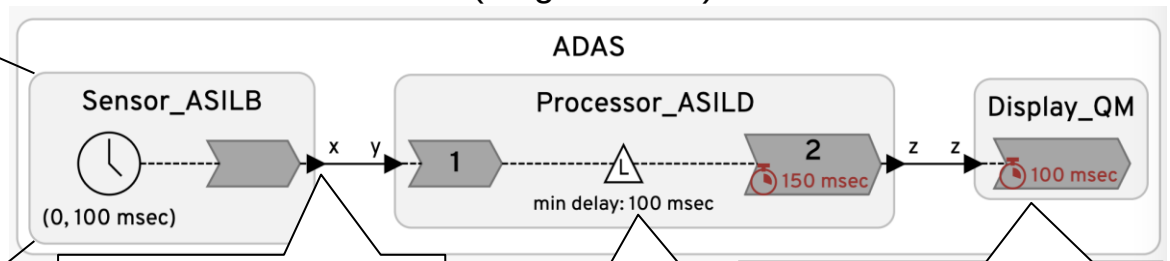
Single reactor
(code view)

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

Periodic trigger

Application logic is written in the supported languages : C/CPP/Python/TS/Rust

Top-level reactor
(diagram view)



Timestamped message

Factoring in WCET of this **workload**

Deadline handler invoked if expected event not triggered within 100 msec

Lingua Franca Compiler

Generated code

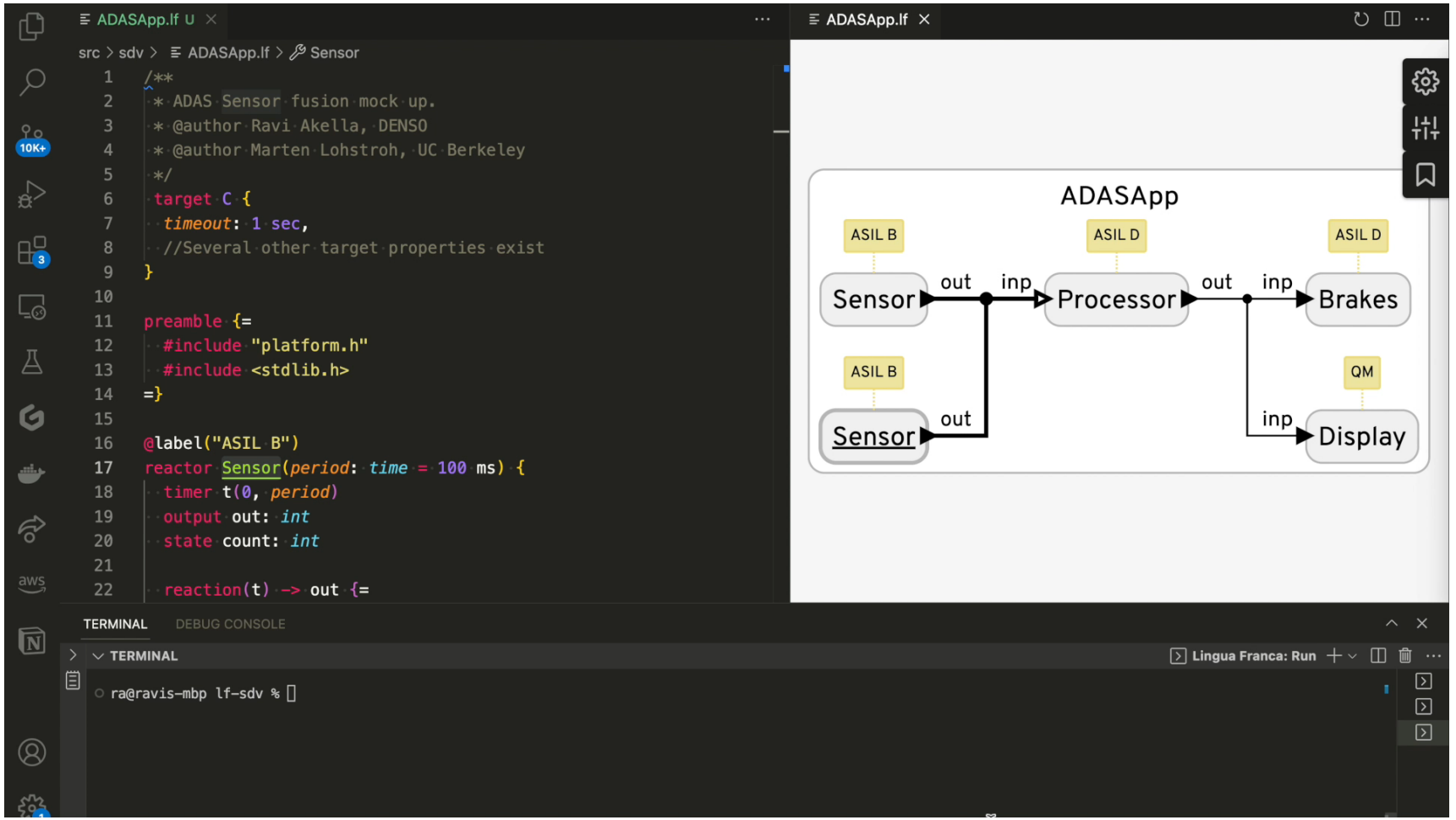
LF Deterministic Scheduler

Target Platform

The **deterministic scheduler** determines the logical time to trigger reactor and monitors for violations of timing.

Lingua Franca allow us to model and execute deterministic application

Demo video


ADASApp.lf U
src > sdv > ADASApp.lf > Sensor
/**
* 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 {=
ADASApp.lf
ADASApp
ASIL B
ASIL D
ASIL D
Sensor
out
inp
Processor
out
inp
Brakes
ASIL B
QM
Sensor
out
inp
Display
TERMINAL
DEBUG CONSOLE
TERMINAL
Lingua Franca: Run
ra@ravis-mbp lf-sdv %

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 provides blueprint to show how such a service can be integrated with SOAFEE SDV reference architecture

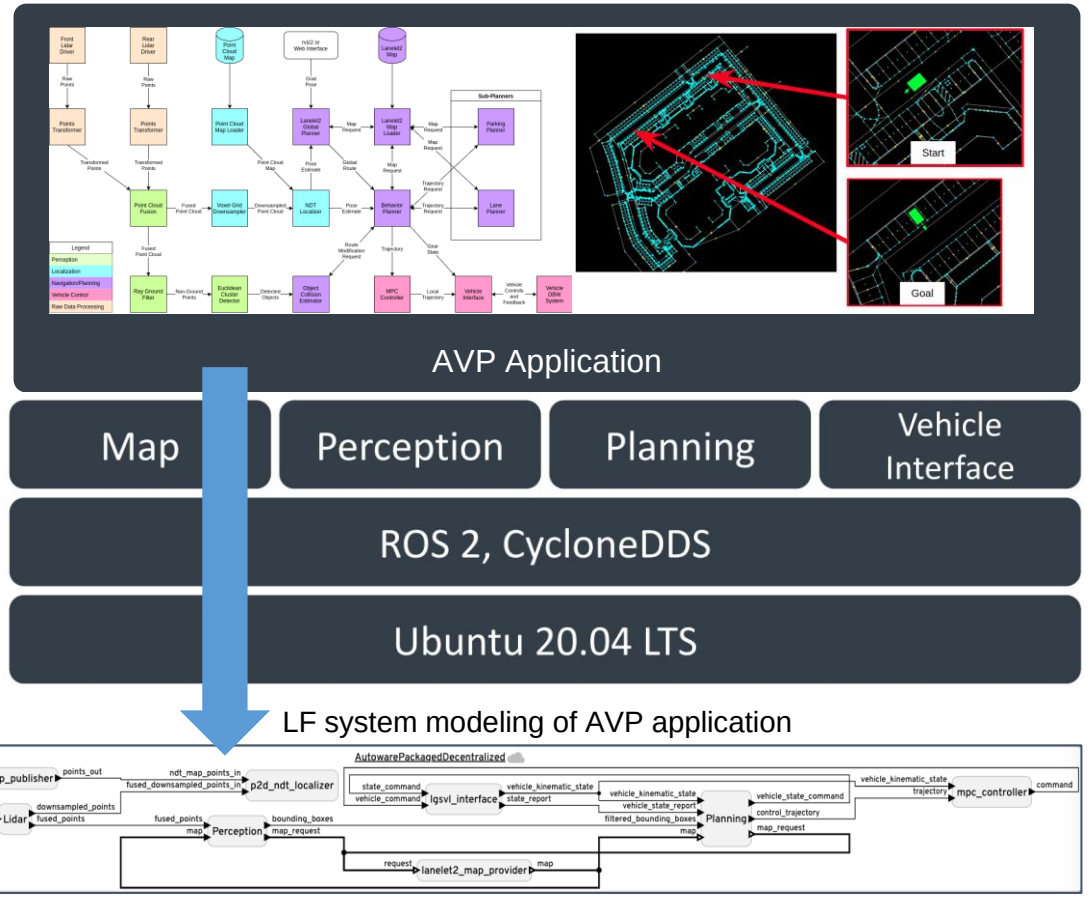
Problem

- AVP application occasionally exhibits non-deterministic behavior (Eg: Order of arrival of messages and execution of nodes)

→ This problem highlights the importance of deterministically scheduling

Approach

- Porting the AVP to Lingua Franca - basically wrapping it up as a reactor. It can guarantee that AVP will run reliably, exactly as we design it.



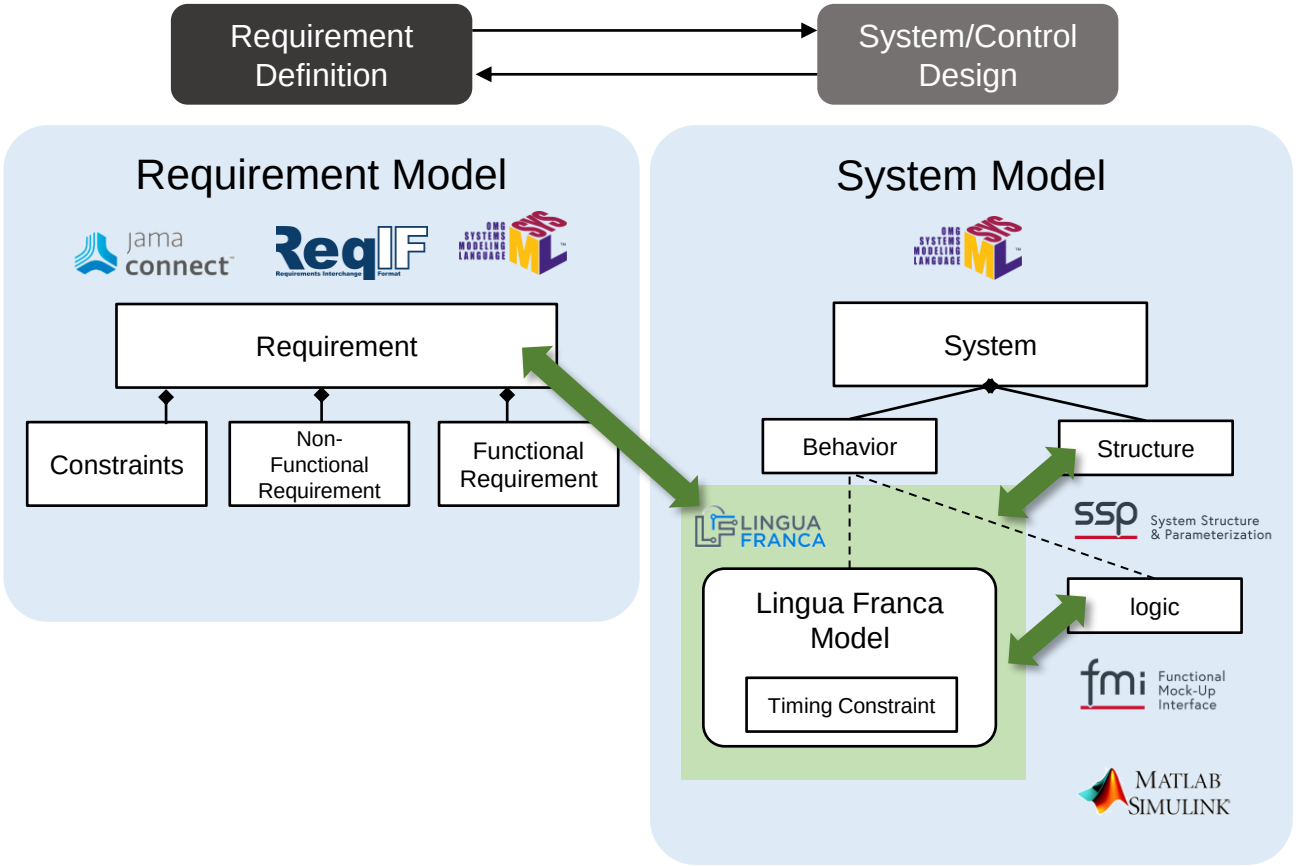
Proposing Lingua Franca for SOAFEE AVP Blueprint to achieve deterministic behavior

Integration with systems engineering

- Systems Engineering is widely used for designing and testing complex systems in aerospace, military, and automotive industries.
- Lingua Franca, a part of Systems Engineering, focuses on modeling time requirements

Our Idea

- Integrate Lingua Franca with requirements models, structural models, and control logic to validate and refine time requirements early in the design process
- This reduces the risk of discovering time issues later, prevents rework, and improves system consistency and optimization.



Use Lingua Franca to refine time requirements in the early design phase (Shift-left)

Conclusion

- The development of SDV technologies through open community activities is accelerating in the automotive industry.
- Our SDV activity focuses on mixed critical system and system modeling
- Proposing Lingua Franca as an essential solution for realizing “mixed critical orchestrator” in SOAFEE
- Integration of Systems Engineering and LF is planned for the purpose of shift-left


Linaro Connect
MADRID 2024 | MAY 12-17 2024
Thank you