FleetOps BOF

PoC based on Eclipse SDV

OSS enablement Team
Agenda

1) Why we did this?
2) What is it?
3) Who is involved?
4) How to achieve it?
What?

Is it: use-cases details & ideas
Fleet Management Systems

**Problem Statement:**

Fleet Operators are required by law to collect certain types of data points like Fuel Consumption, Mileage, Driver Information and others from Commercial Vehicles.

**The Challenge:**

How do we get this very specific data out of the vehicles and into the Fleet Management System?
FMS > How it is done today

**Status Quo:**

FMS vendors and/or TIER’s build dedicated *Telematics Units* to put into commercial vehicles.

**The Challenge:**

This needs to be done for each and every commercial vehicle brand in an FMS proprietary way. This is slow and costly and particularly tedious across brands and FMS vendors.
FMS > How does SDV help?

SDV potential:

Having a generic SDV in-vehicle computer allows to adapt any given commercial vehicle HW environment to any backend FMS simply by deploying some SW

The Challenge:

We still however need to have different SW that is specific to each Commercial Vehicle brand and FMS
**Standards potential:**

Using standard APIs for accessing data allows for employing the same hardware & software components in Commercial Vehicles across brands and FMS vendors.

**The Solution:**

This is the final stage in making the transition towards a fully software defined vehicle by leveraging Open standards and technology we can now:

- Run Fleet SW on any vehicle
- Make changes to data reqs. & scale
- Simplify HW & focus on API’s
How?

To achieve it & what is done already
Hardware Abstraction

Deeply Embedded

Vehicle Computer

Off-board

App

App

SwC

Kuksa.val

Vehicle Signal Specification

Vehicle Signal Specification
Current Architecture

• What’s missing
  ➢ Service Discovery
  ➢ (FOSS) cloud environment
  ➢ more advanced edge client

• Topics for community discussions
  ➢ Identify relevant and missing VSS signals
  ➢ Map VSS signals to other standards
  ➢ Vehicle Application SDK + data-driven broker client
  ➢ Authentication & Authorization
  ➢ an onboarding point for new devs (tutorial etc)
  ➢ Something to use for demos
Current Architecture

- **Kuksa CANOpi**
  - Based in RaspberryPi Compute Module
  - Schematics available as Open Source
Current Architecture + Simulation

• What’s missing

- Service Discovery
- (FOSS) cloud environment
- more advanced edge client

• Topics for community discussions

- Identify relevant and missing VSS signals
- Map VSS signals to other standards
- Vehicle Application SDK + data-driven broker client
- Authentication & Authorization
- an onboarding point for new devs (tutorial etc)
- Something to use for demos
Future Possible Architecture

• Here’s what we’d like to do…
  - Use Eclipse Hono as cloud connectivity layer
  - Implement FMS Feeder as Eclipse Velocitas App

• Topics for community discussions
  - Auth using Eclipse Chariott
  - Onboarding point for new devs (tutorial etc)
  - Something to use for demos
  - Eclipse SommR based SOME/IP feeder
Thank You

OSS enablement Team