# RAPID PROTOTYPING WITH CANOPI, APERTIS AND ECLIPSE KANTO

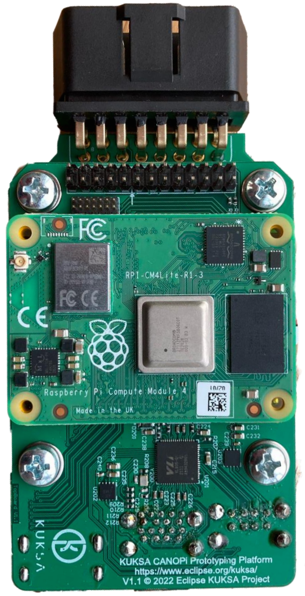
FIELD REPORT

MICHAEL.DOERING2@DE.BOSCH.COM

# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Prototyping – but why?

- **Why prototype at all?**
  - Typical answer:
    - Integral part of agile development!
    - Get user feedback, check hypothesis, ...
  - But much more:
    - Makes your idea/concept/solution more tangible
    - Improves internal and external confidence
    - Requires integration, which fosters interaction, collaboration, and learning
    - Boosts your teamwork


Raspberry Pi Compute Module 4
KUKSA CANOPi Prototyping Platform
https://www.eclipse.org/kuksa/
V1.1 © 2022 Eclipse KUKSA Project

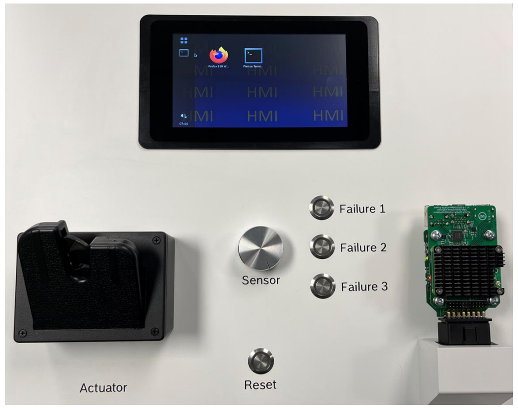
- **Why prototype with CANOPi?**
  - Plugs to almost any car's OBD port
  - Widely used RasPi CM4 compute module
  - Lots of OSS available:
    vehicle abstraction, edge agent, container runtimes, several Linux distros, ...
  - Pre-integrated images available at
    https://www.apertis.org/reference_hardware/rpi_cm4_canopi_setup/

# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Background

- About us: “Embedded IoT Linux@Bosch”
  - Interconnecting and integrating internal Open Source activities, sharing internally and externally
  - Provide integrated Linux reference systems and building blocks
    (e.g. Kanto, secure boot, RAUC, ...) for long lifecycle products
    - Based on Apertis (much more than a distribution, long story, details at our exhibition booth)
    - Recently started to provide reference images also for CANOPi

- This talk is our field report on setting up an SDV-ish reference use case
  - CANOPi- & Kuksa.val-based vehicle interface via CAN/OBD
  - Container deployment and backend interaction based on Eclipse Kanto
  - Integration of generic sensors and actuators
  - “Deeply embedded ECU”-emulator for quick demo setups



# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Our setup

- CANOPi running Apertis reference image
  - Package based, long-term maintainable, suitable (and proven) for product development
  - + integrated and maintained Kanto packages (public package release WiP)

- Reference container
  - Example use case: retrofit “package delivery to trunk”
  - Kuksa.val to map to VSS (e.g. Vehicle.Body.Trunk.IsOpen)
  - Some additional I/O – generic solution for reference
  - Implemented VSS/OBD adapter to operate trunk actuator

- Emulated deeply embedded ECU
  - Not every developer has access to real ECUs and tooling
  - Implemented mockup/emulator (and willing to share)

KUKSA

kanto

APERTIS

BOSCH

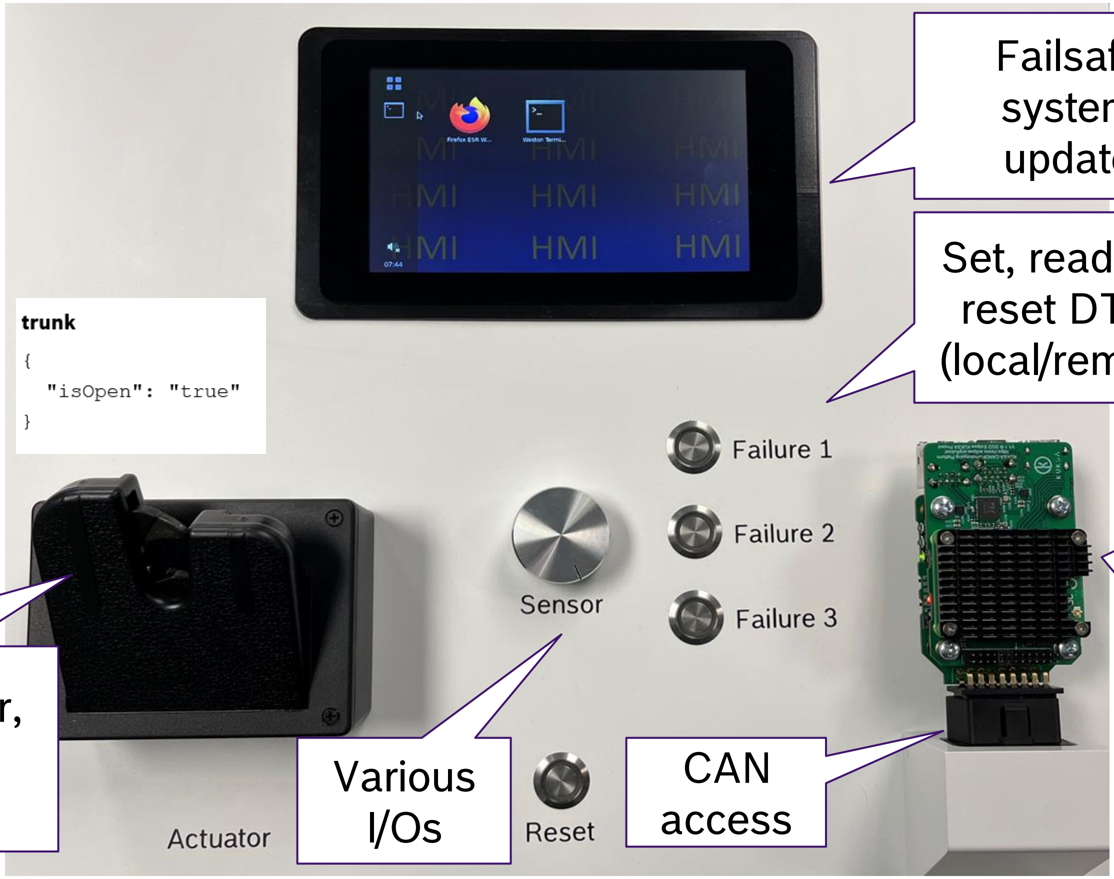
# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Example use case: details and capabilities

**Storyline:**

Aftermarket feature "package delivery to trunk" is retrofitted with CANOPi and deployed by Kanto.

(+ some reference goodies)



**trunk**

```
{
  "isOpen": "true"
}
```

Failsafe system update

Set, read and reset DTCs (local/remote)

**OBD**

```
{
  "controlModuleVoltage": "12.
  "dtc": "{['B1601', ''], ['B1
}
```

- Pre-integrated, maintained Apertis images with Kanto
- Deploy containers
- Kuksa.val, VSS
- Connect to backend

Remotely operate actuator, read state in VSS (Vehicle.Body.Trunk.IsOpen)

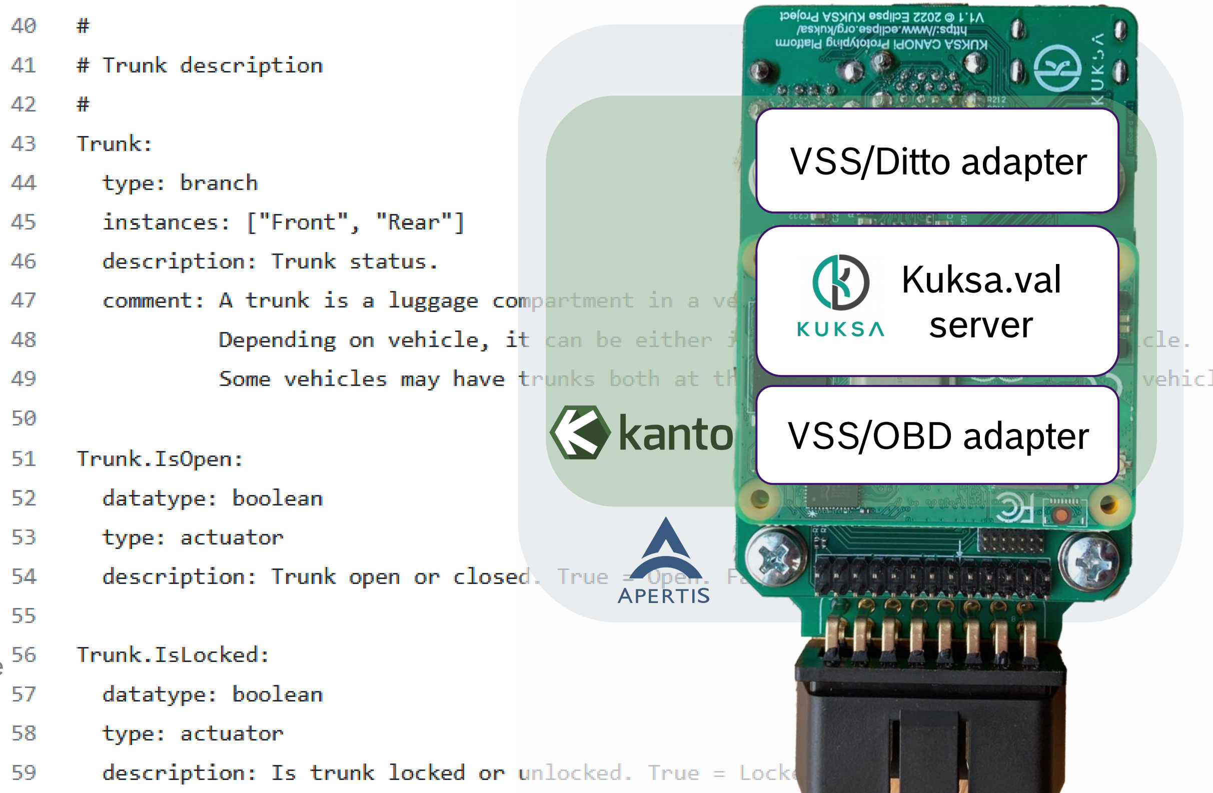
Various I/Os

CAN access

# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Integration with Kuksa.val

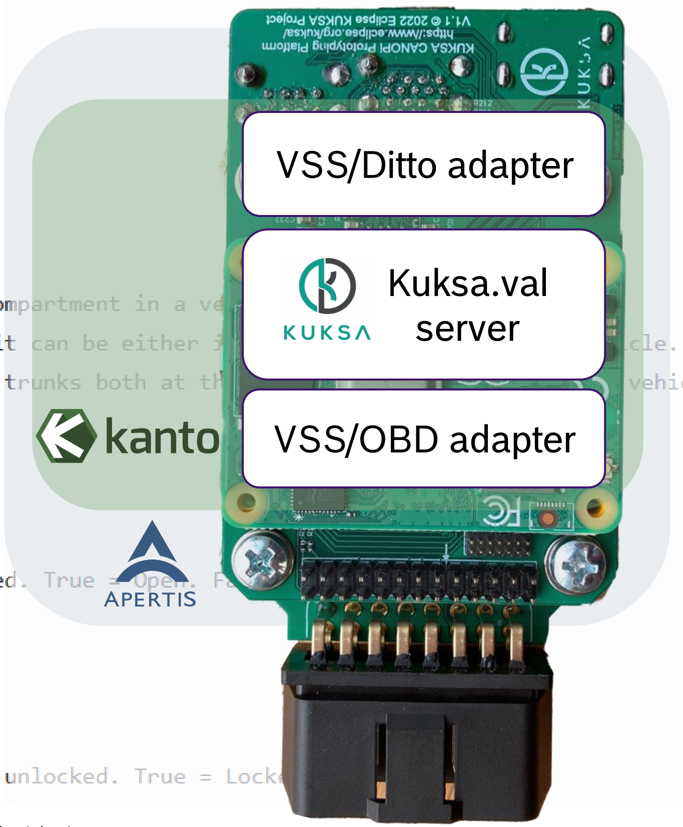
- kanto Essential device IoT enablement
  - Bosch IoT Suite cloud connectivity
  - OTA containers deployment and management
- VSS/Ditto adapter (prototype)
  - Bidirectional VSS handling via a Ditto digital twin
- KUKSA Kuksa.val server
  - Mapping to standardized Vehicle Signal Specification e.g. Vehicle.Body.Trunk.IsOpen
- VSS/OBD adapter (prototype)
  - To inject and receive our CAN messages via OBD
  - Notice: “Open Trunk” via OBD is an example use case!



```
#
# Trunk description
#
Trunk:
  type: branch
  instances: ["Front", "Rear"]
  description: Trunk status.
  comment: A trunk is a luggage compartment in a ve
           Depending on vehicle, it can be either i
           Some vehicles may have trunks both at th

Trunk.IsOpen:
  datatype: boolean
  type: actuator
  description: Trunk open or closed. True = Open. Fa

Trunk.IsLocked:
  datatype: boolean
  type: actuator
  description: Is trunk locked or unlocked. True = Locke
```



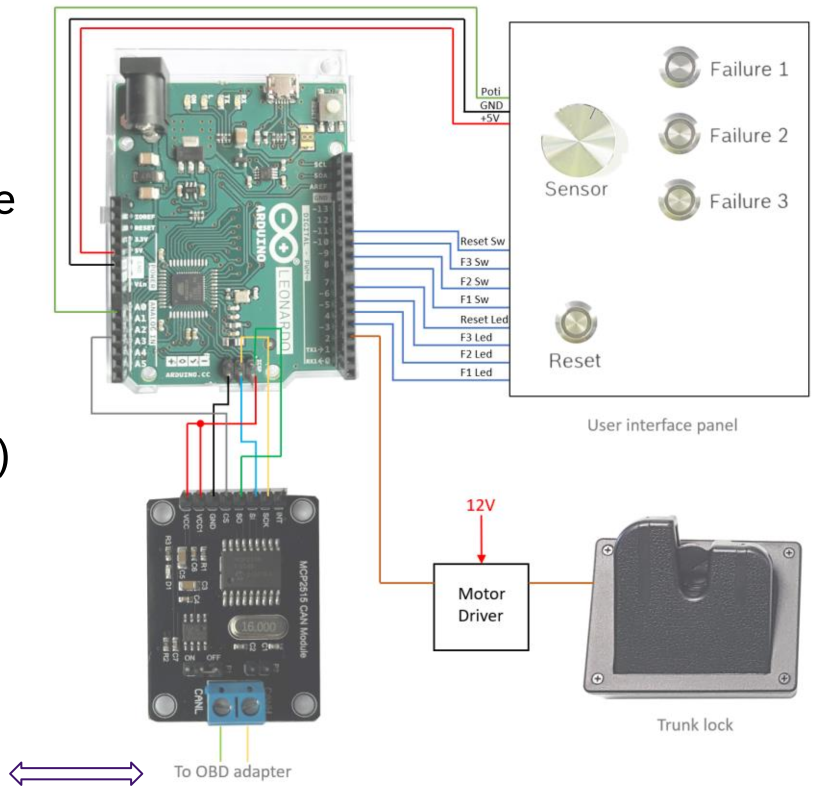
https://github.com/COVESA/vehicle_signal_specification/blob/master/spec/Body/Body.vspec

BOSCH

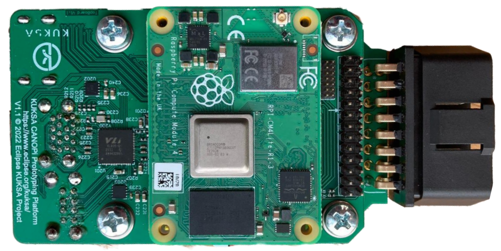
# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Not ready to use a real car / ECU yet?

- Sourcing & using a real ECU (or car!) can be time consuming
  - For early stages emulator more efficient
  - Wanted reference setup available also to non-automotive audience
  - Widely available components, free tools, large maker community
    - No rocket science at all!
  - Convenient and generic building block
  - Also useful for your prototypes, demos, hackathons? (let us know!)



  - Components
    - Arduino Leonardo
    - MCP2515 CAN Module
    - SW: ~750 LoC, ~9kB bin
    - Public release planned



# Rapid prototyping with CANOPi, Apertis and Eclipse Kanto

## Summary

- Our experience with CANOPi
  - User/developer perspective:
    - Great for retrofitting, easy to use, well documented
    - Supply shortage
  - Integrator perspective:
    - Everything north of OBD-plug is easy going
    - Mods on vehicle/ECU much more effort (only if needed for use case)
    - ECU emulator
      - Initial development: some weeks
      - Replication/customization: some hours to days

- Our offer
  - Pre-integrated Apertis images for CANOPi
    https://www.apertis.org/reference_hardware/rpi_cm4_canopi_setup/
    - Some missing packages available soon
  - ECU emulator: willing to share!

- Our exhibition booth
  - See & touch our CANOPi setup
  - Talk with us about embedded IoT Linux!

# THANK YOU!

AND SEE YOU AT OUR EXHIBITION BOOTH ☺

HTTPS://WWW.APERTIS.ORG/REFERENCE_HARDWARE/RPI_CM4_CANOPI_SETUP/

MICHAEL.DOERING2@DE.BOSCH.COM