Sharing signal data in automotive is complex and hindering collaborations.
RemotiveBroker (a.k.a. software ECU)
- the core component and data aggregator

Supports gRPC
Use the programming language of your choice including Python, Rust, C++ etc.

Supported network protocols
CAN (.dbc), SocketCAN, FlexRay (fibex/arxml), LIN (.ldf), UDP arxml, LDF.

Record and Playback
Easy to record signals - replay locally or in the cloud

Remote access
Access the product from anywhere over the Internet

Hardware of your choice
Any Linux/Docker-capable HW
Just download the Docker-image, e.g. to development PC or Nvidia Drive

Host Mobility
Preconfigured
- HMX
- MX-4 T30 FR

RemotiveBox
Raspberry Pi + CAN shield
Platform architecture
- enabling easy access to vehicle data from anywhere

RemotiveCloud - COLLABORATION
- Collaborate internally and with partners
- Store and share drive cycles
- Signal databases and user data
- Run/Provision RemotiveBrokers
- Replay drive cycles
- Real-time mirroring of local brokers

IN-VEHICLE
- Advanced Engineering/Prototyping
- Record drive cycles
- Real-time mirroring through the cloud

TEST RIGS
- Continuous Integration
- Restbus simulation
1. RECORD DRIVE CYCLES
A RemotiveBroker records drive cycle scenarios including potential errors in any vehicle, that could be a mule in early stage development.

2. UPLOAD DRIVE CYCLES
The recorded drive cycles are uploaded and stored in the cloud.

3. FILTER AND SHARE
The relevant signal set is filtered out and signals are renamed according to COVESA VSS in order to hide proprietary information.

4. REPLAY DRIVE CYCLE
The custom Vehicle HAL connects to the RemotiveCloud and starts the playback of the drive cycle; with the result that the AAOS stack thinks it is running in the vehicle where the drive cycle was recorded.
Android Automotive debugging

A custom Vehicle Hardware Abstraction Layer integrates with the RemotiveCloud and engineers can select which recordings to playback, start/stop and seek in a user-friendly way.

- Feed real car signals to an AAOS stack
- Reduced need to have a car platform present
- VSS - filtering out proprietary/irrelevant signals incl. renaming/scaling
Share signals according to VSS

- No issue with different naming conventions
- OEMs choose exactly what to share
- Collaborate, innovate & get stuff done!

Outcome: COVES & VSS

Proprietary OEM data

Filter VSS data

RemotiveCloud

Recorded data

Live data

Broker

3rd parties/partners
AAOS with Vehicle drive playback

- Android Automotive Library used
  - android.car

- Android SDK version/API level
  - Android SnowCone (S)
  - API level: 32

- Android Virtual Device (AVD) details
  - AVD created from Vanilla AOSP-12 with custom VHAL adaptations that uses VSS over gRPC

- HIDL based Vehicle HAL server

- Vehicle Properties used
  - PERF_VEHICLE_SPEED_DISPLAY
  - INFO_MAKE
  - INFO_MODEL
  - INFO_MODELYEAR
  - INFO_VIN
  - TURN_SIGNAL_STATE
Conclusion

• **Collaborate** - use standardised signal names VSS so it gets easier to work together

• **Innovate** - everybody that needs should have access to data to try ideas

• **Get stuff done** - enable partners to do application development

Try out the RemotievLabs platform at [https://demo.remotivelabs.com/](https://demo.remotivelabs.com/)

Contact details: [emil.dautovic@remotivelabs.com](mailto:emil.dautovic@remotivelabs.com) and [renjith.rajagopal@volvocars.com](mailto:renjith.rajagopal@volvocars.com)