From Concept to Reality: How Data-Centric Vehicle APIs Shape Software-Defined Vehicle

Adnan Bekan, Data Expert Team COVESA
Software-defined is a **hardware or service component that's improved or completely managed by software**. With software-defined technology, activities traditionally done by hardware are carried out by software.

Insight.com
Joining SD trend. With WSN and SDN.

- Exploit existing capabilities, align API.
- Replace HW with too many limitations on access level.
What is SDV?

A Software-Defined Vehicle is any vehicle that manages its operations, adds functionality, and enables new features primarily or entirely through software.

Blackberry (QNX)
How about SDV challenges?

Separate concerns

1. **Hardware Diversity and Abstraction:**
   1. Challenge: Integrating diverse hardware components from different manufacturers.
   2. Challenge: Optimizing resource allocation for efficient hardware use.
   3. Challenge: Meeting real-time requirements while abstracting hardware complexity

2. **OTA Updates and Compatibility/Integration:**
   2. Challenge: Managing software versions across diverse vehicle models.
   3. Challenge: Balancing user experience and cybersecurity during updates.

3. **Regulatory Compliance and Testing:**
   1. Challenge: Complying with evolving regulations for OTA updates.
   2. Challenge: Integrating third-party components with existing systems.
   3. Challenge: Rigorous testing and validation for error-free updates.

4. ……
Focus

Unlocking SDV Flexibility: Integrating Diverse Hardware Seamlessly Through Software.

• Seamless integration of diverse hardware components from various manufacturers.
  • Interface alignment
  • Protocol abstraction
• Challenging existing hardware capabilities and limitations.
Proposal

Most common challenges are Fragmentation, Usability, Integration (Test and Validation)
TECHNICAL CHALLENGES - ABSTRACTION.

• Understand existing communication patterns and protocols.
  • SOMEIP, CAN bus, nPDU tunnels ....

• Develop flexible deployment configuration setup.
  • Avoid usage of static C++ code binders, introduce flexibility on interface mapping.

• Define testing and validation pipeline.
  • Test in cloud before you deploy to the vehicle. Test and Validate every deployment file.

• Provide technology agnostic consistent interface for vehicle data and functions.
TECHNICAL SOLUTION.

DETAILS.

- Written in RUST.
- Has SomeIP serializer (RUST).
- Dynamic configuration and serialization based on config files.
- Support for VSS.
- Functional API described using ASYNC API/ OpenAPI STANDARD. Technology agnostic.
INTERFACE DEFINITION AND VALIDATION. (2022)

Virtual Car V1
- pennyBecker
  - Common Interface
  - Mapping files. V1
- Application/Service Consumer for testing

Virtual Car V2
- pennyBecker
  - Common Interface
  - Mapping files. V2
- Application/Service Consumer for testing

CLOUD testing infrastructure.

Deployment details, SP specific.

ASYNCAPI spec with VSS.

Client Experience.

AmbientLight:
- address:
- wdpPort: ...
- serviceId:
- attributes:
  - ambientLightActive: reliable: false
  - getterId:
  - setterId:
  - type: UInt8
  - ambientLightBrightness: reliable:
  - getterId:
  - setterId: ...
  - type: UInt8

asyncapi: 2.4.0
info:
- title: Clio API
- version: 1.0.0
servers:
- name:
  - mqtt://192.0.1688
protocol: mqtt
channels:
- Vehicle.Cabin.Lights.AmbientLight/set:
  message:
  - payload:
    - type: object
      properties:
        active:
          x-return:
          x-type: integer
          x-mapping:
            enum: [false, true]

Client Experience.
How to disrupt traditional Tier 1 and Tier 2 suppliers to enhance access to the full potential of existing hardware capabilities?
Thank you for your attention!
Backup