An Introduction to Eclipse Kuksa

Marco Wagner, Jörg Tessmer
Robert Bosch GmbH
Eclipse Kuksa

Agenda

- The Projects APPSTACLE and Eclipse Kuksa
- Technical details
  - Kuksa Cloud Platform
  - Kuksa In-Vehicle Platform
  - Kuksa Integrated Development Environment (IDE)
- Potential Deployment Scenarios
- Summary and Outlook
The Projects APPSTACLE and Eclipse Kuksa
Kuksa heritage
Publicly funded project APPSTACLE

- **APPSTACLE**: open standard Application Platform for carS and TrAnsportation vehicles
- **Eclipse Kuksa**: the Open Source project hosting all code developed
- **Objective**: Development of an *Open Source* Connected Car Ecosystem
  - Development of an open source automotive IoT Cloud Platform
    - Architectural considerations for the cloud platform
    - Establishment of standardized interfaces to the vehicle
  - Definition and development of Service enablers for car-to-cloud connectivity
    - Network infrastructure considerations
    - Next generation mobile networks
  - Development of an open source in-vehicle platform
    - Safe and secure gateway to the cloud
    - In-vehicle data access mechanism and application platform
Eclipse Kuksa
Providing a solid foundation

Providing a solid technical foundation routed in Open Standards and proven software will benefit everybody.

Create a **cross-vendor** connected vehicle platform that relies on open standards and uses **open source software** to leverage the potential of a **large developer community**!
Eclipse Kuksa is not trying to reinvent the wheel,

Instead we use and foster Open Source solutions to create a harmonized composition of existing Open Source projects enriched with specific Kuksa components
Some technical details...
Eclipse Kuksa
Kuksa Cloud Platform
Eclipse Kuksa
Kuksa In-Vehicle Platform

**Application layer:**
- Runs 3rd party apps on the platform
- Contains a Sandbox Environment & Additional Services

**Middleware layer (Yocto layer):**
- APIs to abstract the vehicles’ E/E architecture (W3C VISS, Sensoris...)
- Communication Services to manage network access and provide data from the vehicle
- Includes communication libs, protocols, security layers,...

**OS layer:**
- Reuse of OE’s existing services, layers, HW abstractions, AGL services, etc.
- Planned: Debian support (e.g. Apertis)
Eclipse Kuksa
Kuksa In-Vehicle Platform

Current focus topic: W3C VISS API

- Implementation of the functional elements is done
- Current focus lays on authentication & access management
- A first proposal for authentication & access has been discussed with the W3C Automotive Working Group

Try it yourself: [https://github.com/eclipse/kuksa.invehicle/tree/master/w3c-visserver-api](https://github.com/eclipse/kuksa.invehicle/tree/master/w3c-visserver-api)
Eclipse Kuksa
Kuksa IDE

- Based on Eclipse Che
- Allows Cloud and In-Vehicle Application development
- Platform independent
- Shared workspaces
- Almost configuration free
- Docker-based: VPN planned to allow remote / network independent cross compilation
Potential Deployment Scenarios
Eclipse Kuksa

Deployment Variants: Integrated

Vehicle Owner

Select Apps

Publish Apps

Install Apps

Data / Commands & Use Interaction

Data

3rd Party

Vehicle

Owner

App Store

hawkBit

ditto

App IDE

3rd Party

Big Data Analysis

Visualization

Data Management

Report Generation

Core Services

App Store

In-Vehicle Platform

W3C API

App 1

App 2

In-Vehicle Platform

App 1

App 2

W3C API

Automotive Grade Linux
Kuksa Demo

Integrated Deployment: Driver authentication
Eclipse Kuksa
Deployment Variants: Retrofit

Vehicle Owner

Select Apps

User Interaction

Data / Commands

Install Apps

3rd Party

Data

In-Vehicle Platform

OBD dongle

OBD

Standard HMI

Mirroring System

3rd Party

Data

App IDE

Plug-In eclipse

App IDE

App Store

hawkBit

ditto

In-Vehicle Platform

W3C API

App 1

App 2

Vehicle Owner

Select Apps

Publish Apps

Install Apps

Data / Commands

User Interaction

Standard HMI

Mirroring System

3rd Party

Data
Kuksa Demo

Retrofit Deployment: OBD Demonstrator

- Dongle running the Kuksa In-Vehicle Platform connected to the OBD port
  - First set-up: Raspberry Pi + Bluetooth OBD Dongle
  - Kuksa OBD Device is currently being developed

- Dataset is a bit limited and depends on the vehicle

- HMI using the Smartphone and Mirroring function of the Head Unit

- Brings your old car into the cloud
The next steps
Kuksa Roadmap
Where to go from here?

12.2017
Initial Kuksa In-Vehicle Setup

01.2017
Start of APPSTACLE

10.2017
Platform study finished → AGL

02.2018
BCX presentation & Hackathon

04.2018
Kuksa-AGL running with OBD-II

05.2018
Security concepts

06.2018
Initial Contribution at Eclipse

07.2018
App Management & W3C API

08.2018
MQTT support

09.2018
Driver Authentication Demo

10.2018
Docker support

12.2018
Retrofit Demonstrator

02.2019
Debian support

04.2019
Sensoris Interface

05.2019
BCX participation & Hackathon

12.2019
Kuksa 1.0
Kuksa Roadmap
Where to go from here?

- Eclipse Kuksa Website
  - https://www.eclipse.org/kuksa/

- Eclipse Kuksa Codebase
  - Eclipse Kuksa IDE: https://github.com/eclipse/kuksa.ide
  - Eclipse Kuksa Cloud: https://github.com/eclipse/kuksa.cloud
  - Eclipse Kuksa In-Vehicle: https://github.com/eclipse/kuksa.invehicle
  - Eclipse Kuksa Integration: https://github.com/eclipse/kuksa.integration

- Direct Contact
  - Jörg Tessmer Joerg.Tessmer@de.bosch.com
  - Pratheek Rai Pratheek.Rai@de.bosch.com
  - Sebastian Schildt Sebastian.Schildt@de.bosch.com
THANK YOU
BackUp
API Authorization Proposal

- App could be remote or local

1) Verify App Instance Certificate
2) Check authorization of App Instance for requested Vehicle ID
3) Generate token (JWT) with permissions, etc.
4) Sign token

Authorization Request:
1) Verify token
2) Update access control state according to permissions from token

Other command (get/set/pub/sub):
1) Check access control state
2) Deny access or execute command