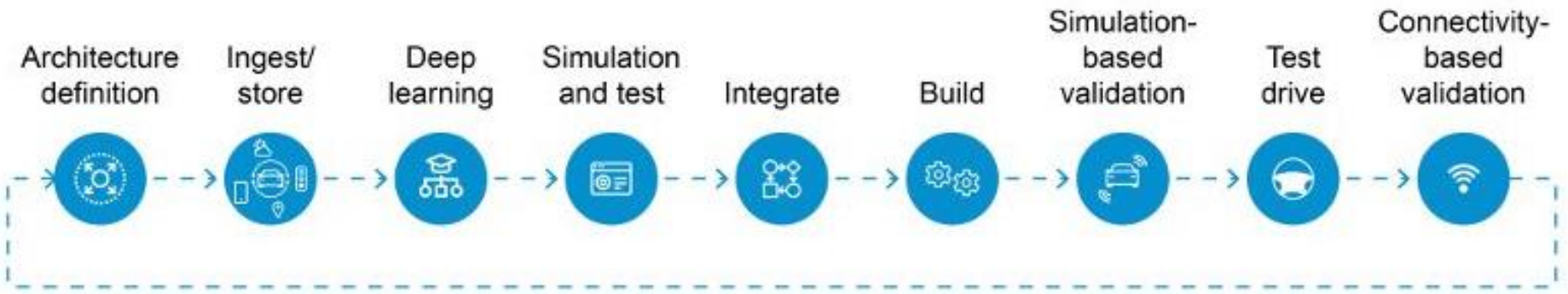
INTEGRATING ECLIPSE KUKSA WITH AN OPENADX TOOLCHAIN

Eclipse OpenADx WG
OpenADx Toolchain

- A community effort to enable compatibility between toolchain components for implementing autonomous driving functions

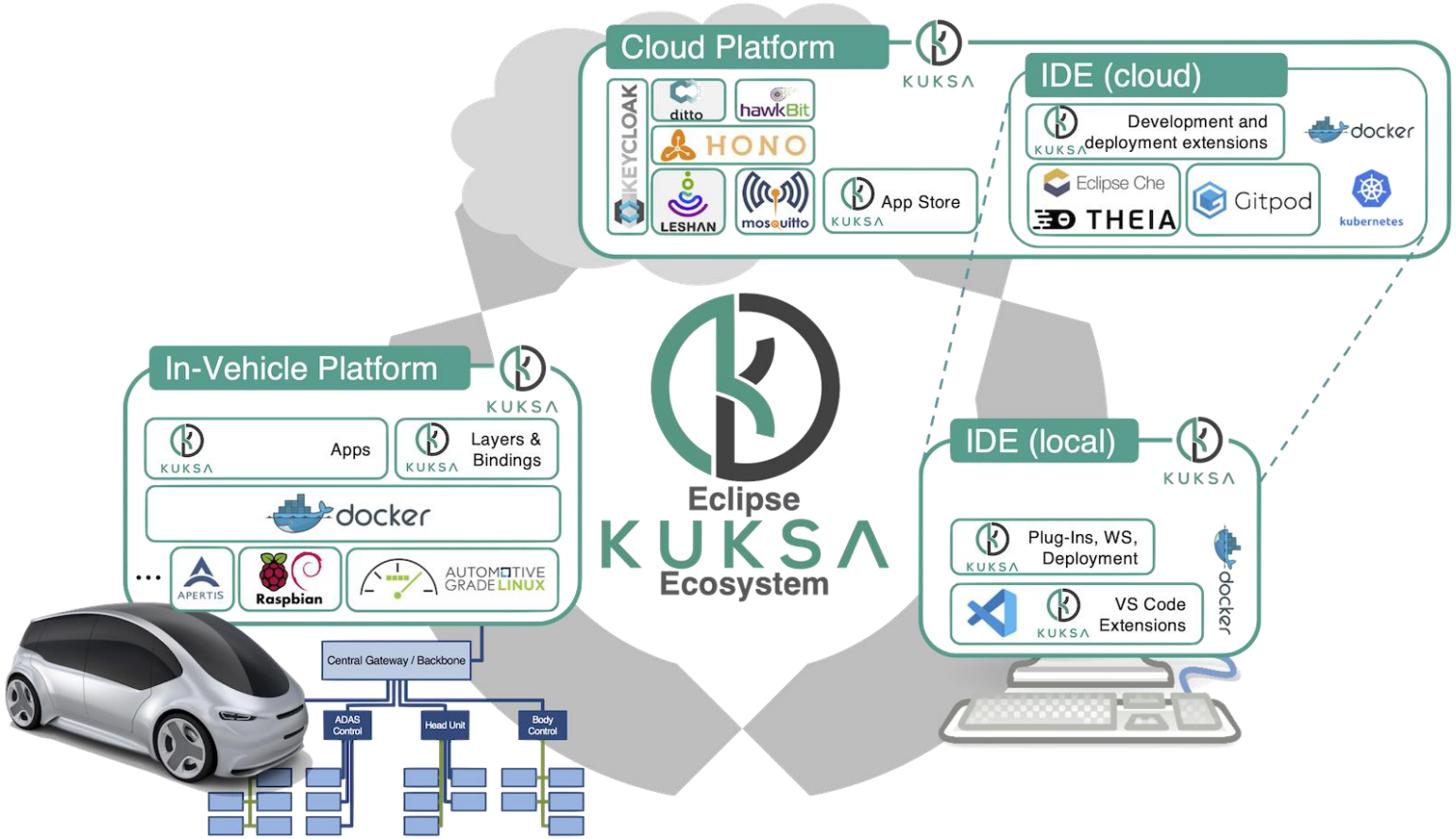


Eclipse Kuksa
Vision

*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**!*

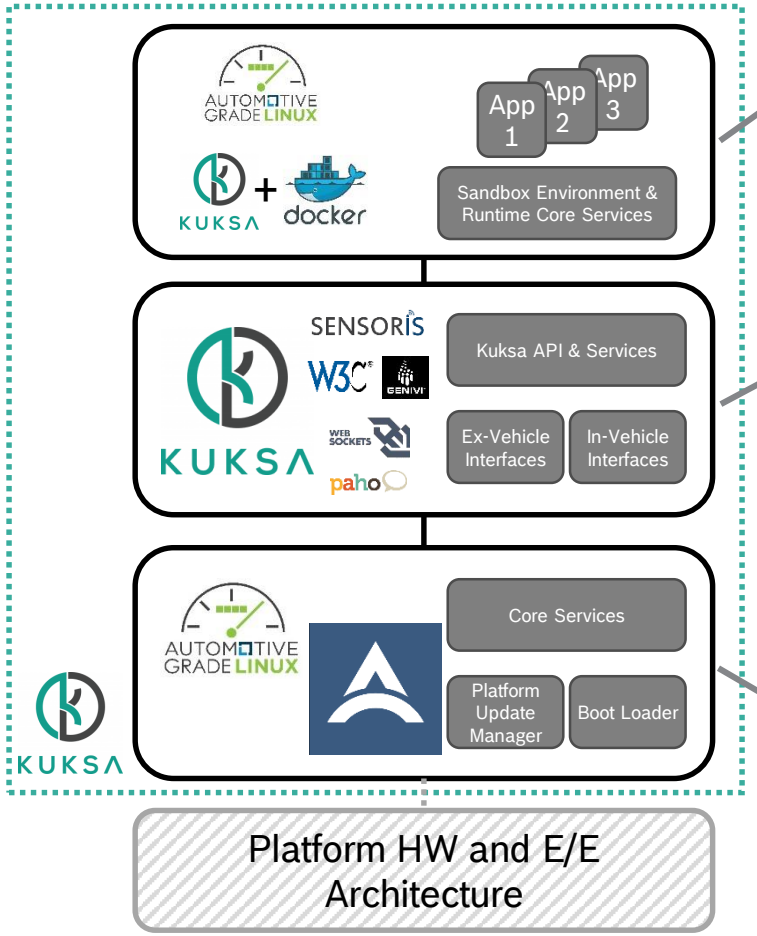
eclipse iot +  = KUKSA

Eclipse Kuksa
Ecosystem



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):

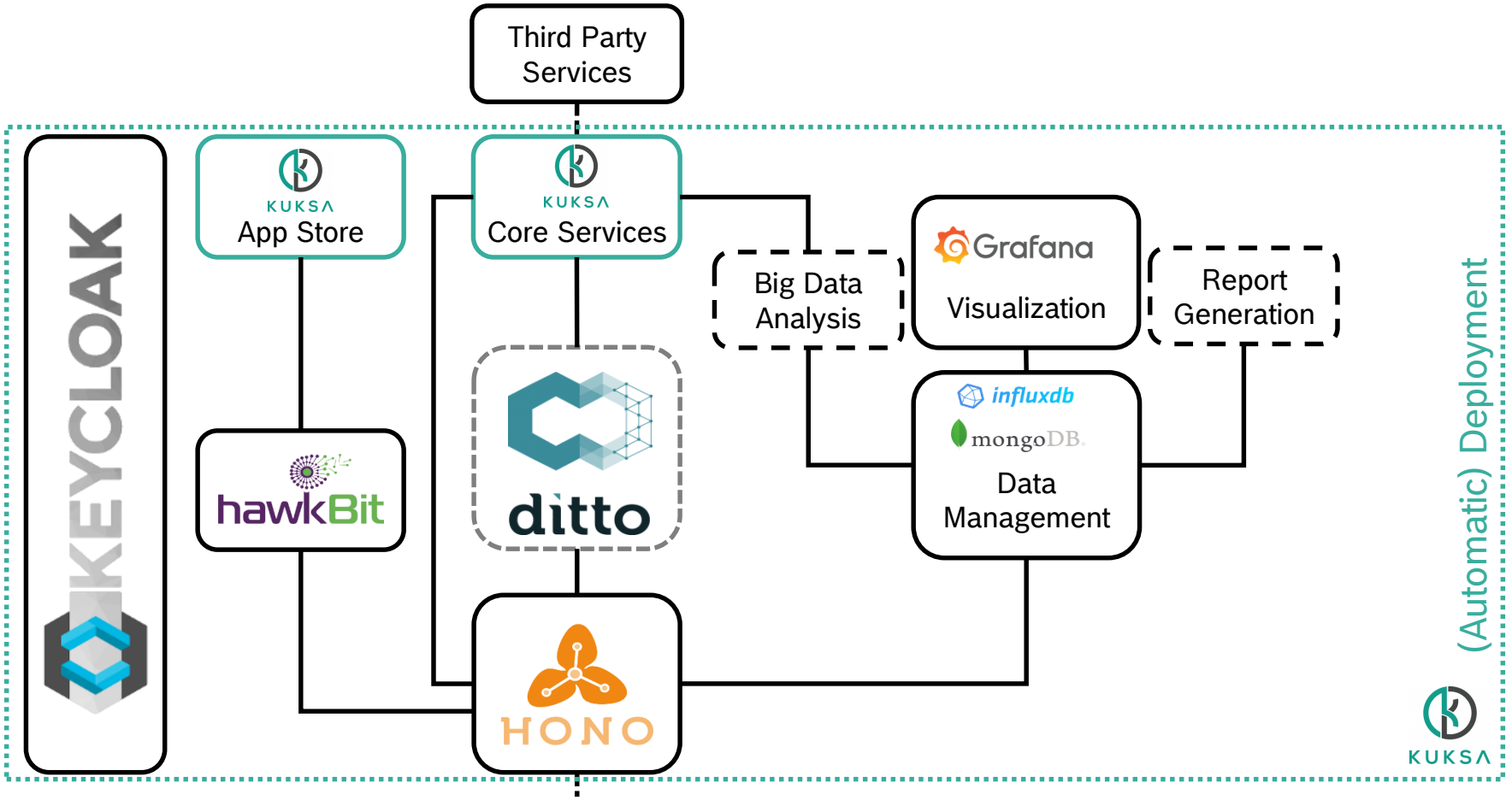
- APIs to abstract the vehicles' E/E architecture (W3C VISS, Sensoris…)
- Communication Services to manage network access and provide data from the vehicle
- Includes communication libs, protocols, security layers,…

OS layer:

- Reuse of OE's existing services, layers, HW abstractions, services, etc.

KUKSA

Eclipse Kuksa
Kuksa Cloud Platform

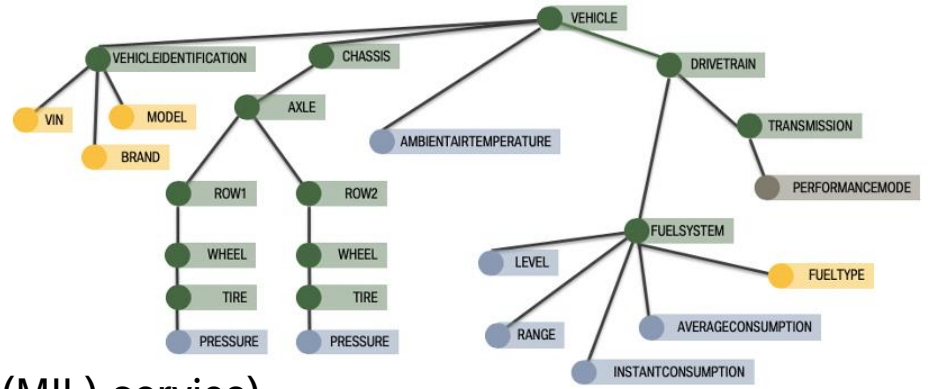


KUKSA

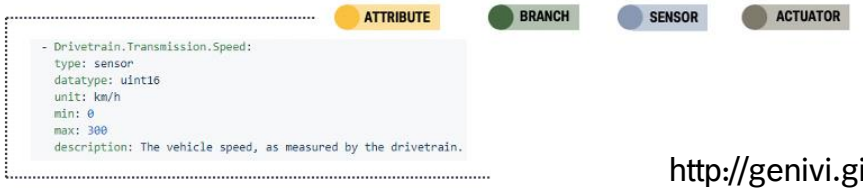
Eclipse Kuksa
Kuksa Val (Vehicle Abstraction Layer)

- Implements Vehicle Signal Specification (VSS) data model proposed by GENIVI
  - e.g. web-socket interface based on W3C VISS (Vehicle Information Service Specification) which relies on VSS data model

- VSS specifies a domain taxonomy for vehicle signals
  - 43 car attributes, 451 branches and 1060 signals currently



- Examples:
  - *Vehicle.OBD.Status.MIL* (Used by Malfunction Indicator Light (MIL) service)
  - *Vehicle.Cabin.InteriorLights.Row1.Right.IsPassengerOn* (toggle right passenger light)
  - *Vehicle.Engine.EOT* (engine oil temperature)

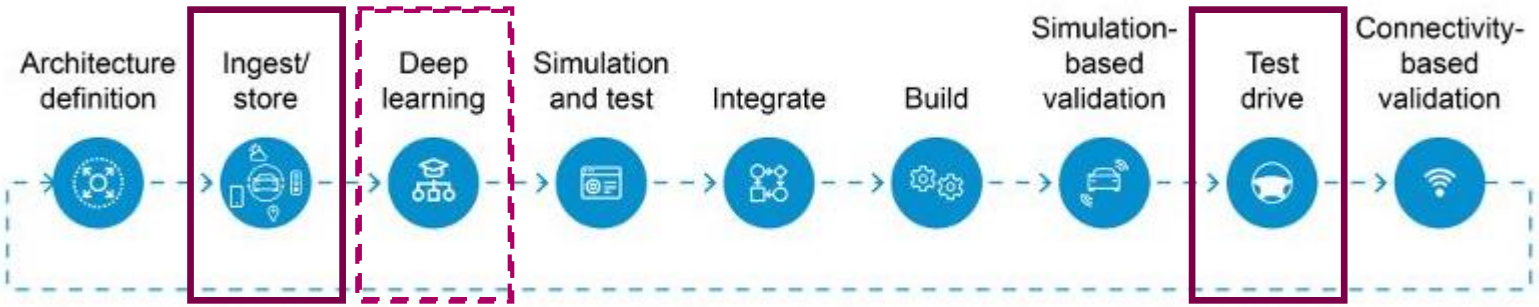


http://genivi.github.io/

KUKSA

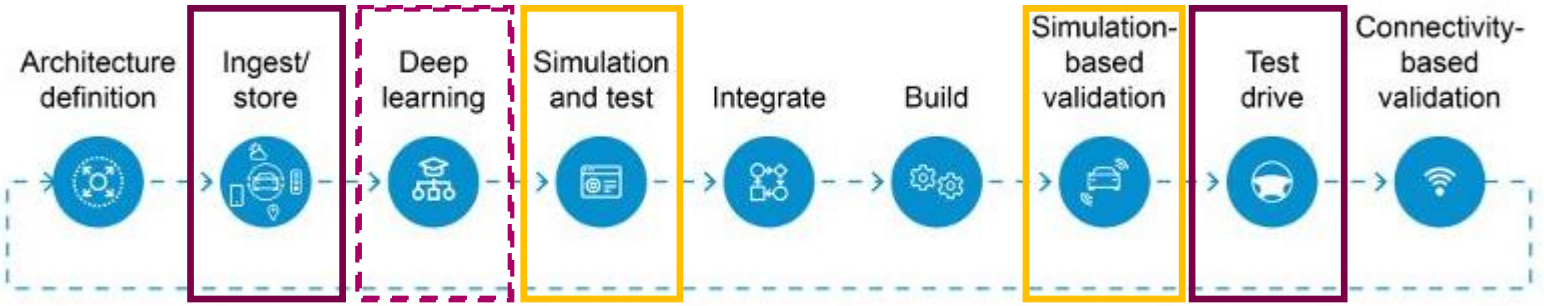
Touchpoints between Eclipse Kuksa and OpenADx

- Ingest/Store:
  - Ingest data from In-Vehicle platform in model from Val and store in Cloud
    [Kuksa.Cloud: Eclipse Hono & InfluxDB / Kuksa.Val: VSS implementation]
- Test drive:
  - Use In-Vehicle platform for execution of actual function and test-drive aspects (e.g. data recording)
    [Kuksa.InVehicle]
- Optional Deep Learning:
  - Realization of Deep Learning component as specific function of general concept of function development and modeling for In-Vehicle applications and Cloud services
    [Kuksa.Cloud: Appstore / Kuksa.Apps: Eclipse HawkBit & Eclipse Che]

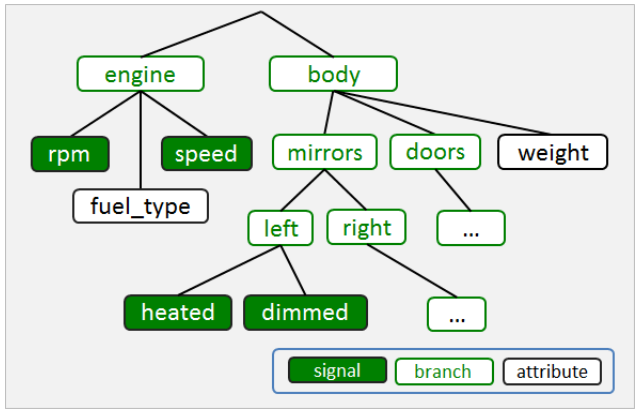


Touchpoints between Eclipse Kuksa and OpenADx
Proposed Integration

- Simulation and test / Simulation-based validation:
  - Assure functionality of developed Kuksa In-Vehicle applications and Cloud services by using simulations
  - Especially next-generation mobility services in the context of autonomous driving require sophisticated validation
  - Simulation of all relevant aspects of the physical world required:
    - Vehicles, bicycles, pedestrians, and further traffic participants
    - Traffic lanes, intersections, cross traffic, traffic rules etc.
    - Environmental conditions such as weather or daytime



Example Use Case
Kuksa Cloud Dashboard



Example Vehicle Signal Tree [1]

- Diagnostic Trouble Code
- Speed
- RPM
- Fuel Status

Kuksa Cloud

Grafana

Kuksa Cloud Dashboard

HTTP

HONO

Hono MQTT Adapter

MQTT

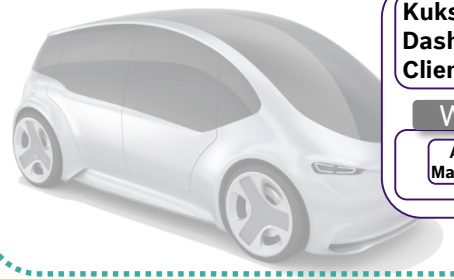
AMQP

Cloud Dashboard Service

spring boot

influxdb

Kuksa In-Vehicle



Kuksa Dashboard Client

docker

W3C API

App Manager

KUKSA

In-Vehicle Platform

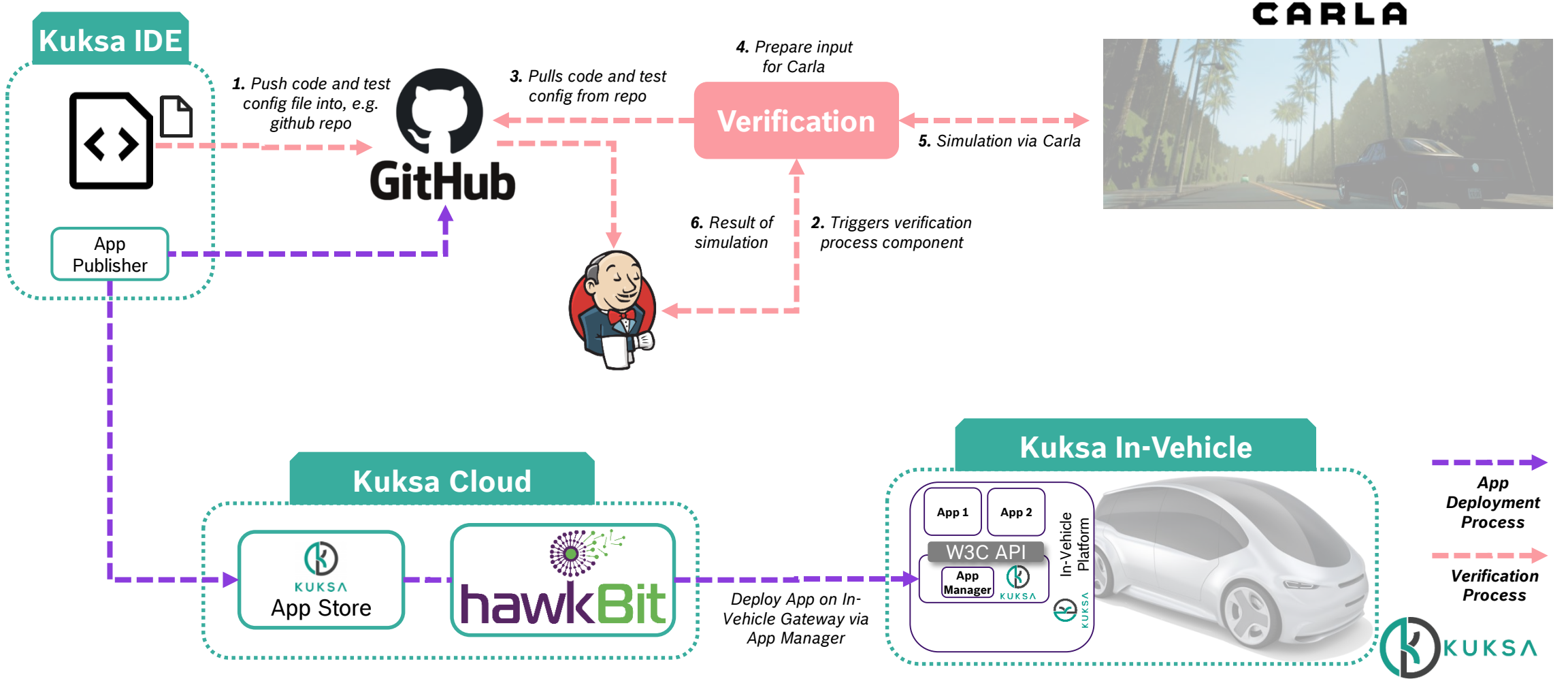
KUKSA

Simulation of Client Application to assure its functionality

[1] https://www.w3.org/TR/vehicle-information-service/

KUKSA

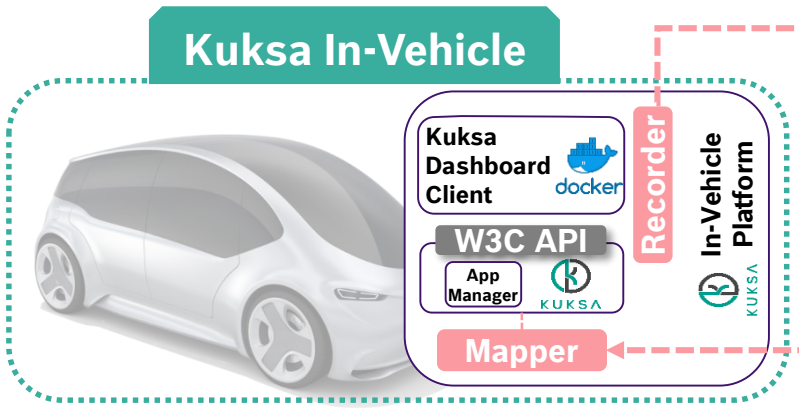
Integrating simulation steps
Architecture of proposal



Integrating simulation steps
Testing Kuksa In-Vehicle Applications

- Feed In-Vehicle application with sensor values from a simulated vehicle to assure functionality
- Enable assessment of software in early development phases
- Challenges:
  - Connecting Kuksa In-Vehicle platform within Carla
    - Carla-ROS-Bridge as blueprint
  - Formal simulation scenario description (*Test Config*)
  - Matching of Carla sensor values with, for example, VSS
  - Only subset of scenarios are supported due to available sensor values

CARLA

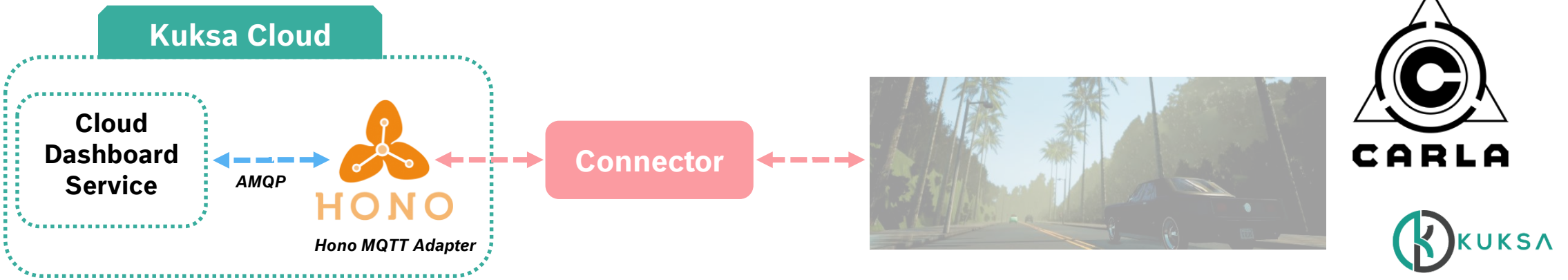


KUKSA

Integrating simulation steps
Testing Kuksa Cloud Services

- Obtain sensor values from simulation and send it to the respective Kuksa Cloud service to assure functionality of service

- Challenges:
  - Providing mass data from simulation based on a vehicle fleet
    - Running co-simulation with SUMO
  - Integration of Command & Control, i.e. sending commands to the simulated vehicle to modify its behavior
  - Metrics to assure the functionality of the deployed Cloud service
    - E.g. measure scalability or detect Microservices Anti-patterns



QUESTIONS???

More Information: https://www.eclipse.org/kuksa/

Mailing List: kuksa-dev@eclipse.org

Bi-weekly Zoom meeting every Thursday on even calendar weeks from 1-2pm (CET/CEST)