Building Bridges with a common Data Middleware

Status of the PoC project after the Porto AMM:

Integration of an **Onboard Sync Server** into the vehicle

Christian Mühlbauer, Data Expert Team BMW AG

COVESA

Accelerating the future of connected vehicles

Data Centric Architecture. Motivation & Goals.

Architecture

- Model and technology agnostic
- Common architecture for data transfer and handling
- Common data domain models to interchange data
- On- & Off-board share common data architecture
- Efficient data transfer

Everywhere & Anytime

- Increasing the availability of data
- Data is available, even if single ECU‘s are shutdown

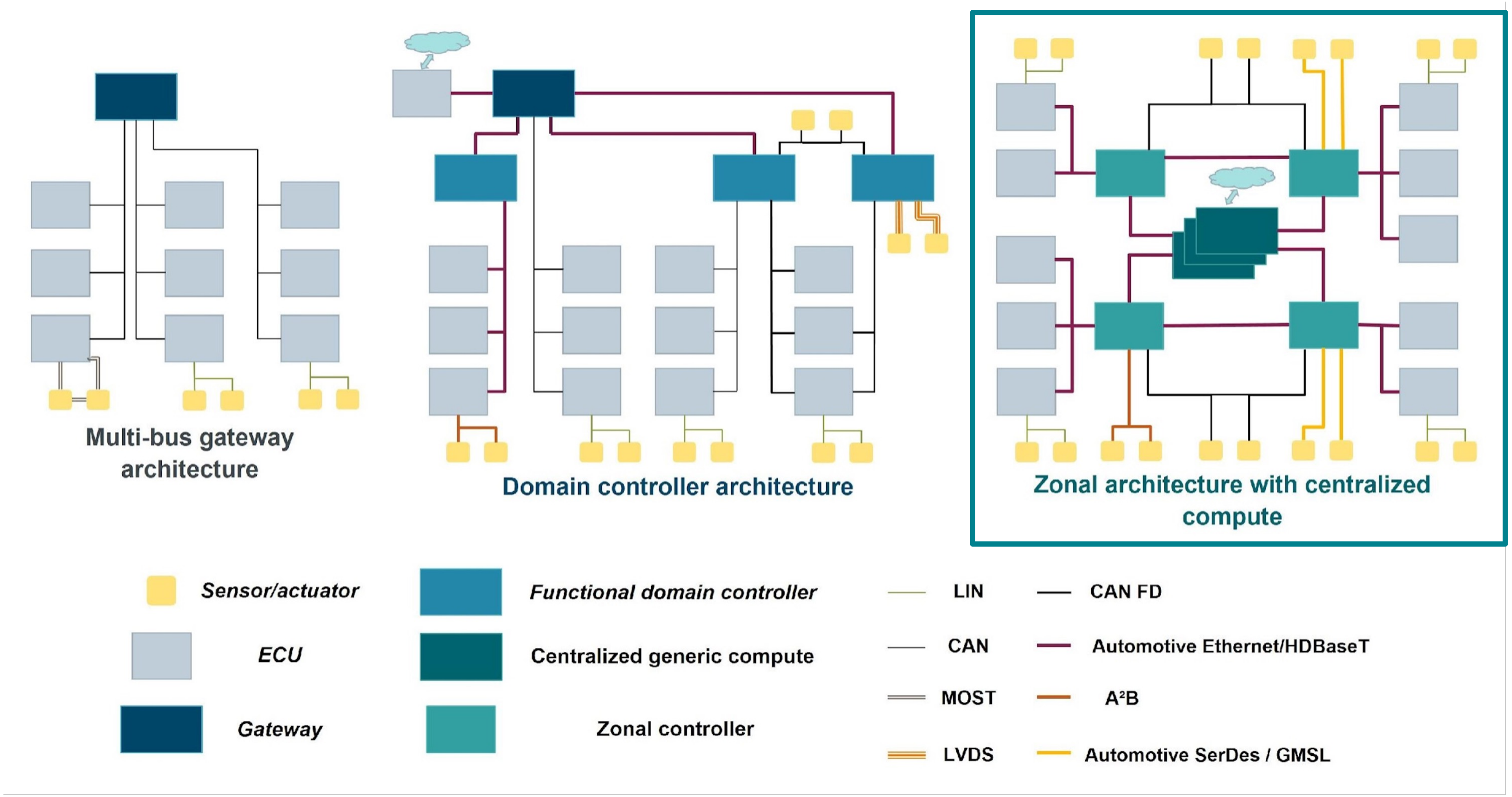
Cost efficient

- Reduction of costs
- Shorter feedback loops, faster reaction to regulations
- Time-to-market for data-centric use cases

Simplicity

- Simplify interaction with vehicle on & off-board
- Data centric use cases on & off-board

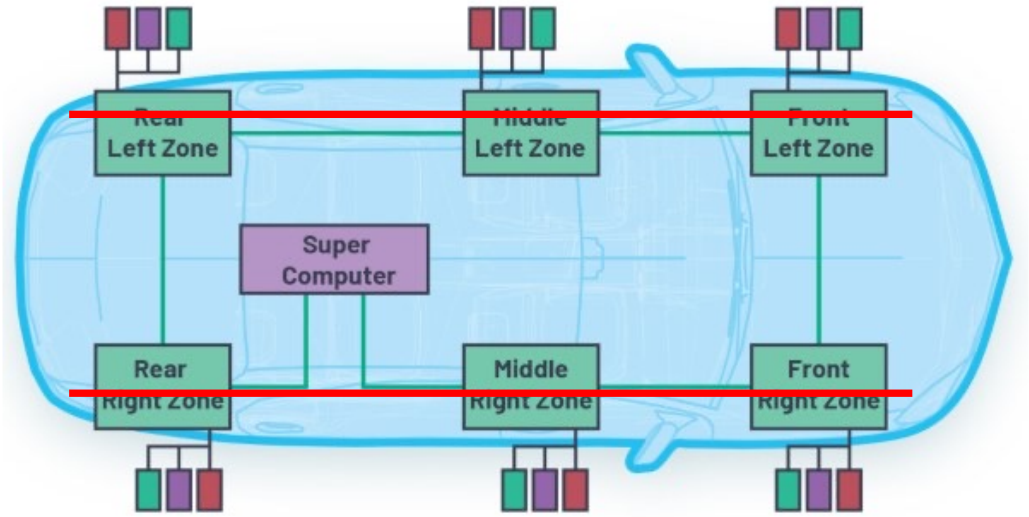
Typical E/E architecture styles in automotive

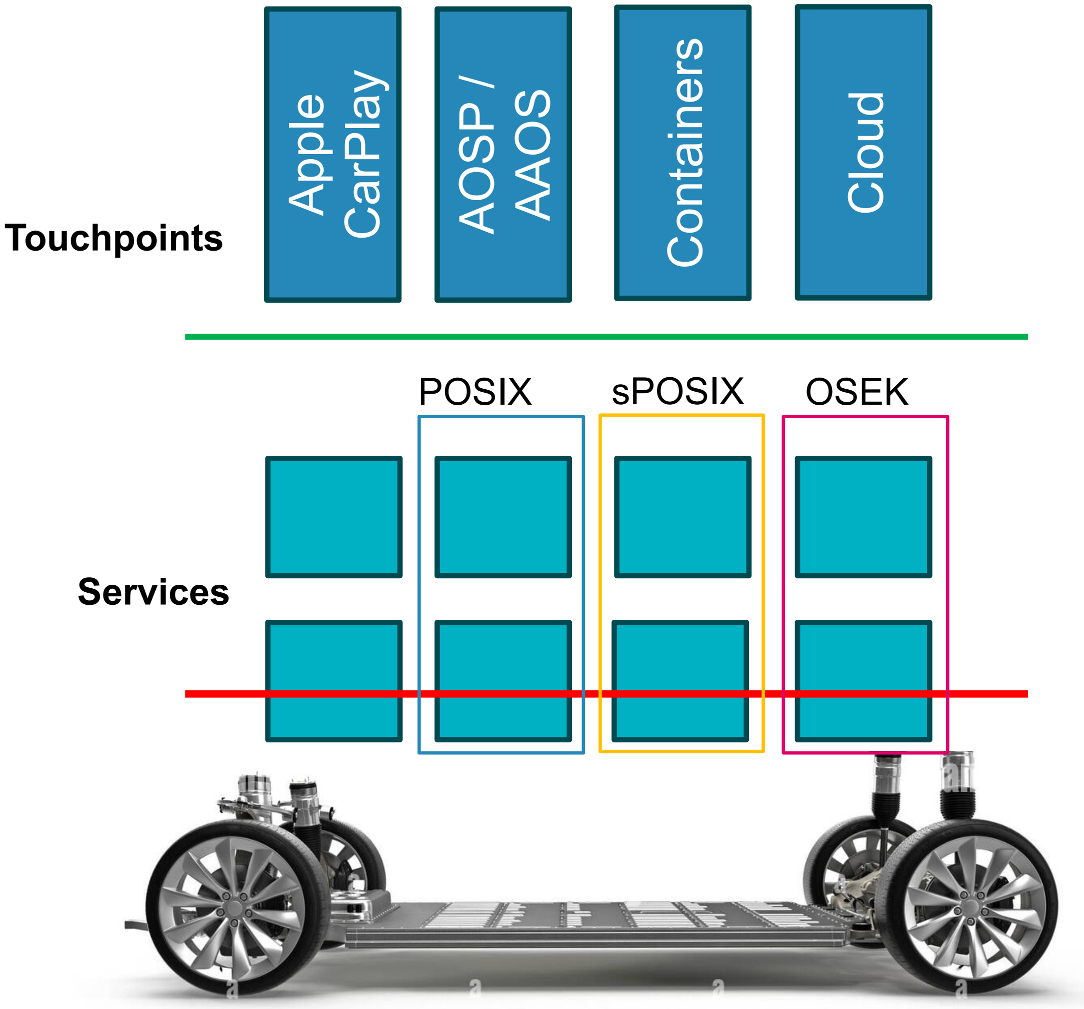

Multi-bus gateway architecture
Domain controller architecture
Zonal architecture with centralized compute
Sensor/actuator
ECU
Gateway
Functional domain controller
Centralized generic compute
Zonal controller
LIN
CAN
MOST
LVDS
CAN FD
Automotive Ethernet/HDBaseT
A²B
Automotive SerDes / GMSL

https://www.eetasia.com/identifying-e-e-architecture-requirements-for-autonomous-vehicle-development/

How can a data-centric architectural approach based on a zonal architecture be implemented?

The basis: Data-Centric Vehicle APIs for Software-Defined Vehicle





PoC Focus

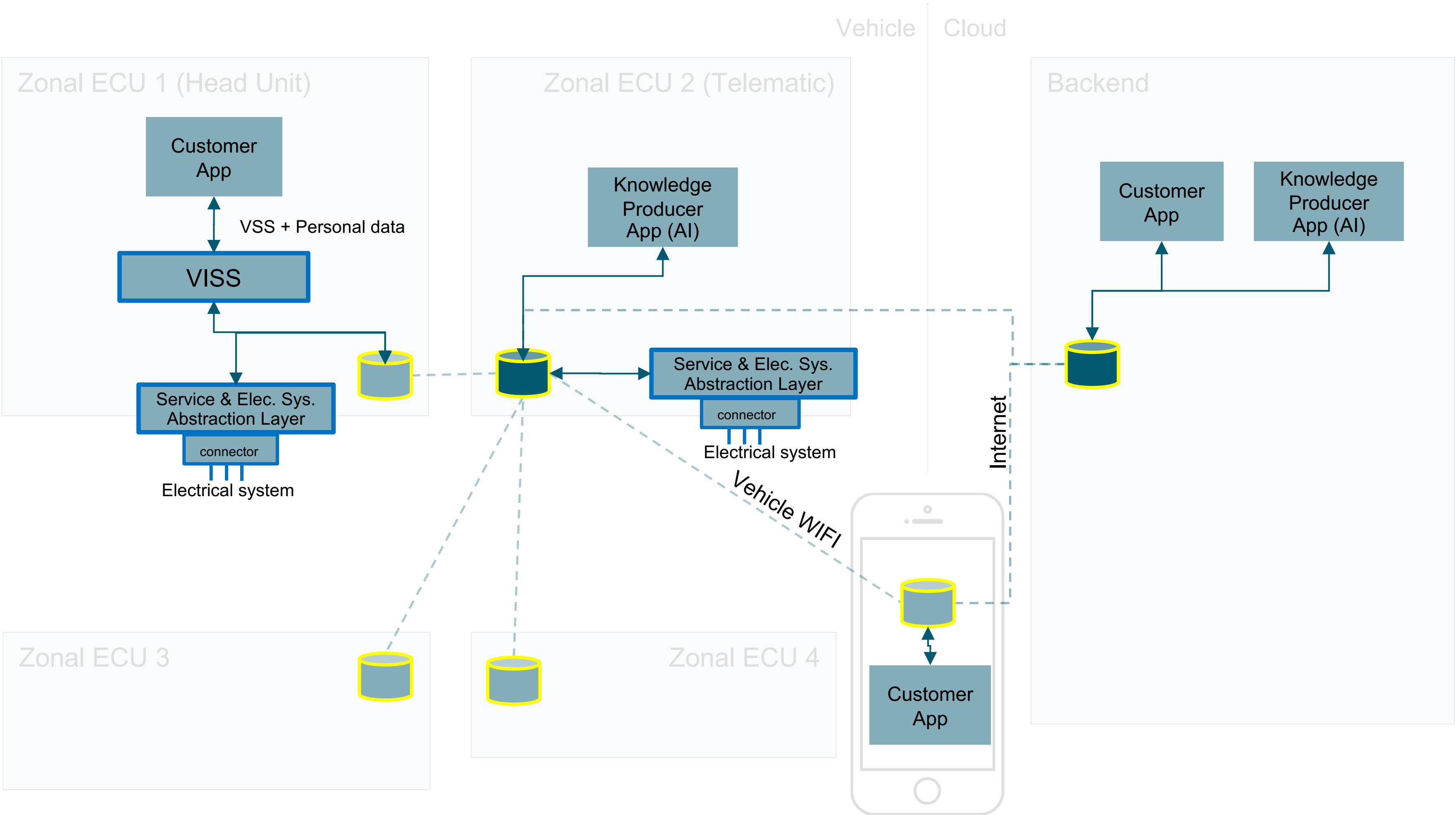
STEP 3
…to benefit
here!

STEP 2
Simplify
here…

STEP 1
Data-Centric
Vehicle API as
Enabler

January 23: Our starting point

https://wiki.covesa.global/download/attachments/46628897/2023_01_18_COVESA_Data_Middleware_V2.0.pptx?version=1&modificationDate=1674836644431&api=v2



Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

January 23: Best way to find answers? PoC!

Minimum PoC goals:

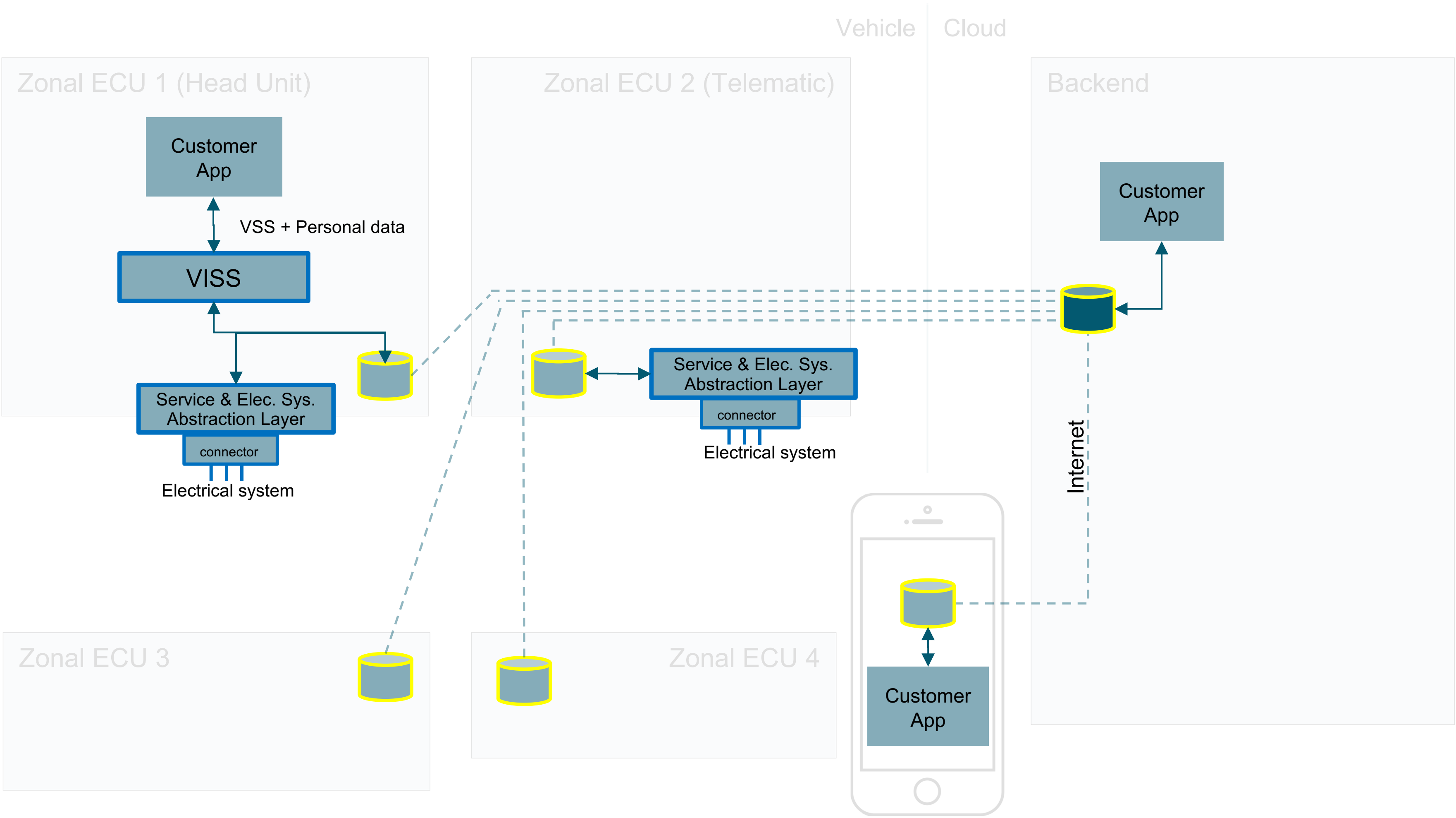
- Amount of automotive requirements which can be covered by 3rd party middleware
- Scalable VSS data model applicability tested in end to end middleware
- Different data domains (different „trees“) can share same end to end infrastructure for sync, access, permissions …
- Standardized data models like VSS(o) can serve as a base for AI / Knowledge generation

Optimum:

- Open source Reference Implementation for 3rd parties to align with Automotive needs

May 23: AMM Porto PoC with BMW i7

https://wiki.covesa.global/download/attachments/64979378/COVESA_Data-Domains_and_Sync.pptx?version=1&modificationDate=1683043726670&api=v2



Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

May 23: AMM Porto PoC with BMW i7

1. **Data Sync** via **3rd Party Data Middleware**

3. **Data Access** via **ViSS**

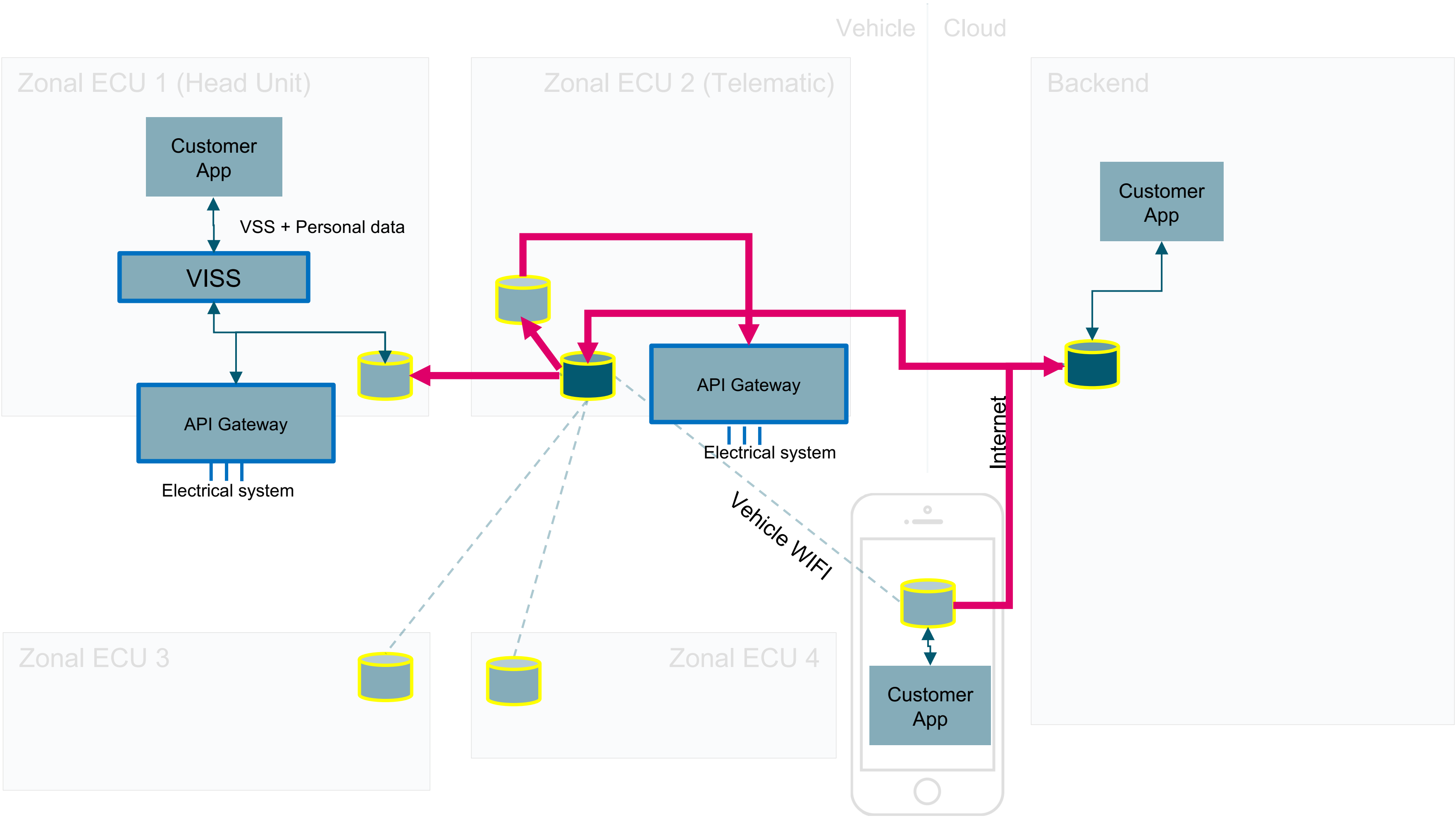
2. **Vehicle Settings** via **VSS** per Passenger

4. Onboard Personalization Domain Logik sends necessary changes to **API Gateway (Pennybacker)**

Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

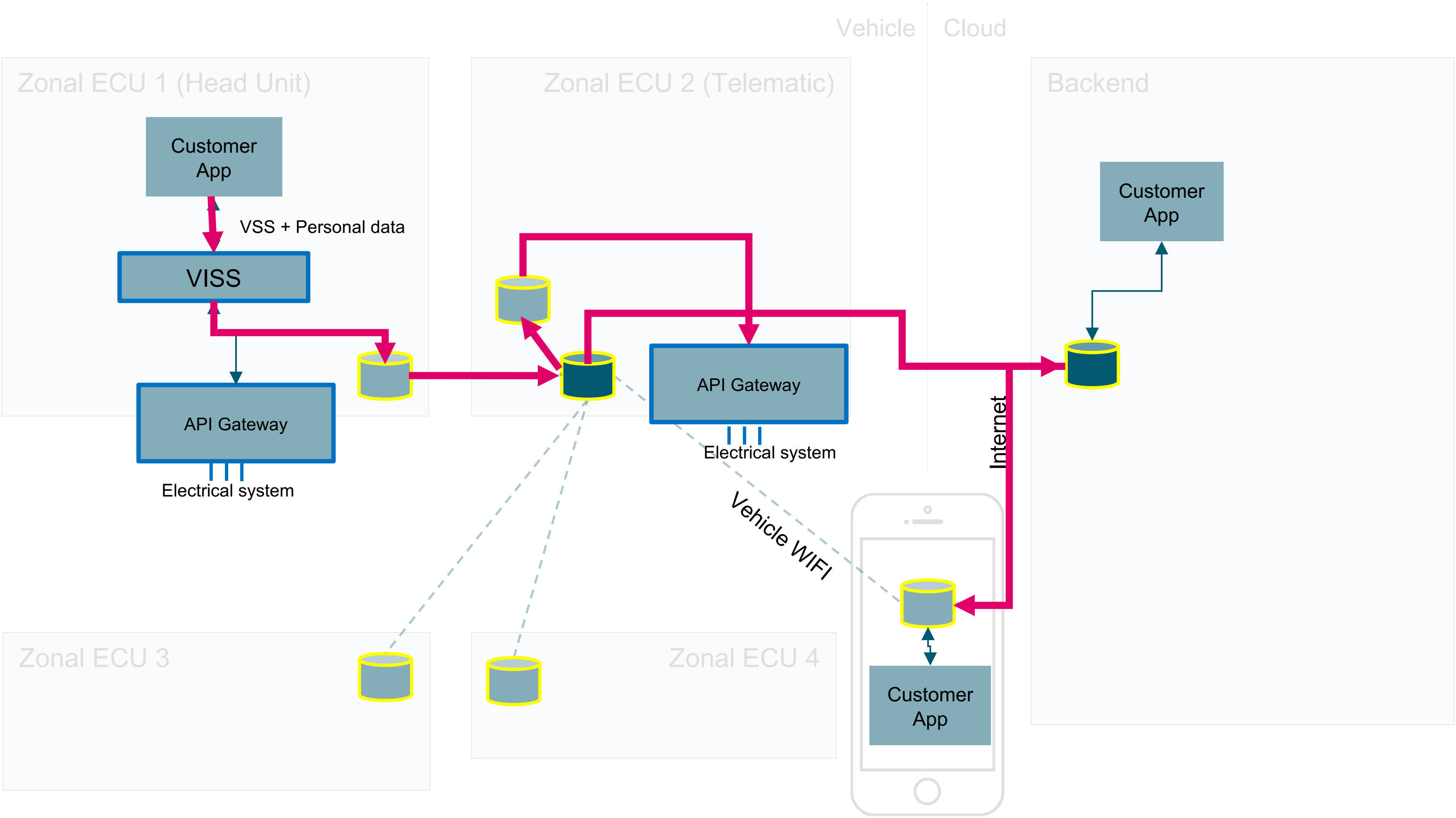
Today: Porto PoC extended by Onboard - Sync Server



Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

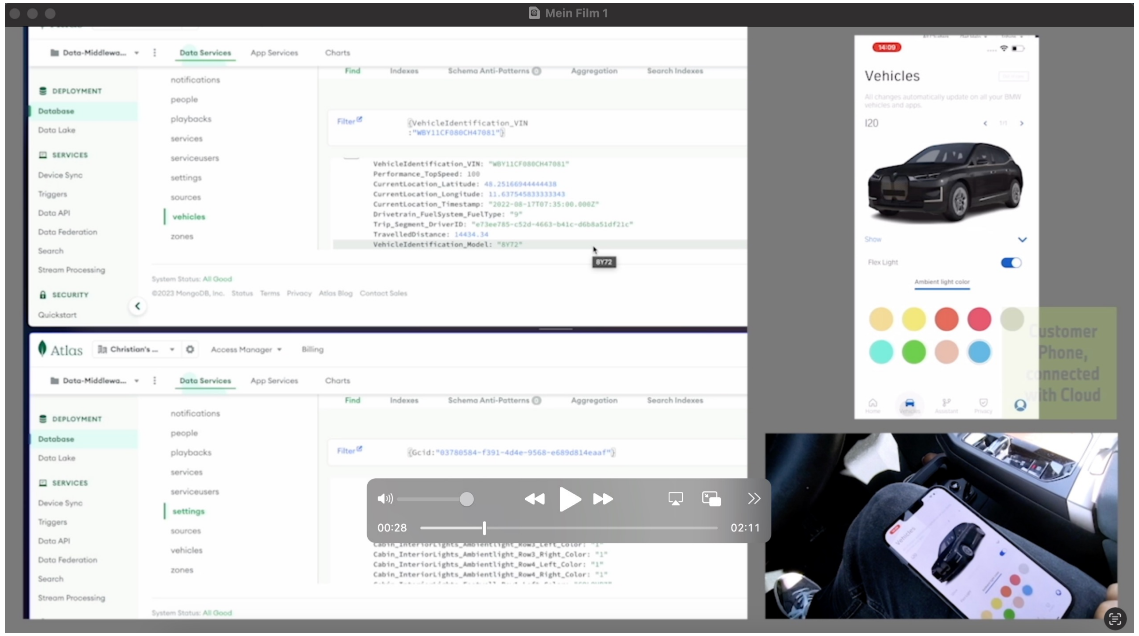
Today: Porto PoC extended by Onboard - Sync Server



Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

Today: Porto PoC extended by Onboard - Sync Server



```
{VehicleIdentification_VIN
:"WBY11CF080CH47081"}

VehicleIdentification_VIN: "WBY11CF080CH47081"
Performance_TopSpeed: 100
CurrentLocation_Latitude: 48.25166944444438
CurrentLocation_Longitude: 11.637545833333343
CurrentLocation_Timestamp: "2022-08-17T07:35:00.000Z"
Drivetrain_FuelSystem_FuelType: "9"
Trip_Segment_DriverID: "e73ee785-c52d-4663-b41c-d6b8a51df21c"
TravelledDistance: 14434.34
VehicleIdentification_Model: "8Y72"
```

```
{Gcid:"03780584-f391-4d4e-9568-e689d814eaaf"}

Cabin_InteriorLights_Ambientlight_Row3_Left_Color: "1"
Cabin_InteriorLights_Ambientlight_Row3_Right_Color: "1"
Cabin_InteriorLights_Ambientlight_Row4_Left_Color: "1"
Cabin_InteriorLights_Ambientlight_Row4_Right_Color: "1"
```

Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

Challenges we faced

General / Legal / Technical Challenges:

- Different touchpoints mean different SDKs (e.g. different functional scope, maturity)
- Different touchpoints mean different architectures (e.g. x86, ARM)
- Handling time-shifted actuator changes vs. sensor value
- In order to meet all country-specific legal requirements regarding data protection, a technical solution is needed that can fulfill all requirements
- Data model changes cause adjustments in many different places. Is there „perfect“ data model governance supported by a toolchain?
- Ensuring compatibility after breaking changes (programmatically not easy)
- Long data point names but also deep nesting can lead to problems (UUIDs?)
- Subscriptions on field level sometimes not easy to handle, e.g. for complex objects
- Local cache compatibility when changing the sync endpoint is challenging
- What degree of modularization can be achieved to remain as technology agnostic as possible

(later Playground discussion!)

Key questions for a End-to-End Data Architecture:

- ⚠ How data can be shared between all touchpoints?
- ✅ How different domains of data share same tech stack
- ✅ How should a bidirectional sync work?
- ⚠ Who is responsible for conflict management?
- ⚠ Who takes care about permissions, roles, rights and privacy?
- ⚠ How the data model can be updated and sycned?
- ⚠ How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- ⚠ How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

Did we achieve already some goals? Yes!

Minimum PoC goals:

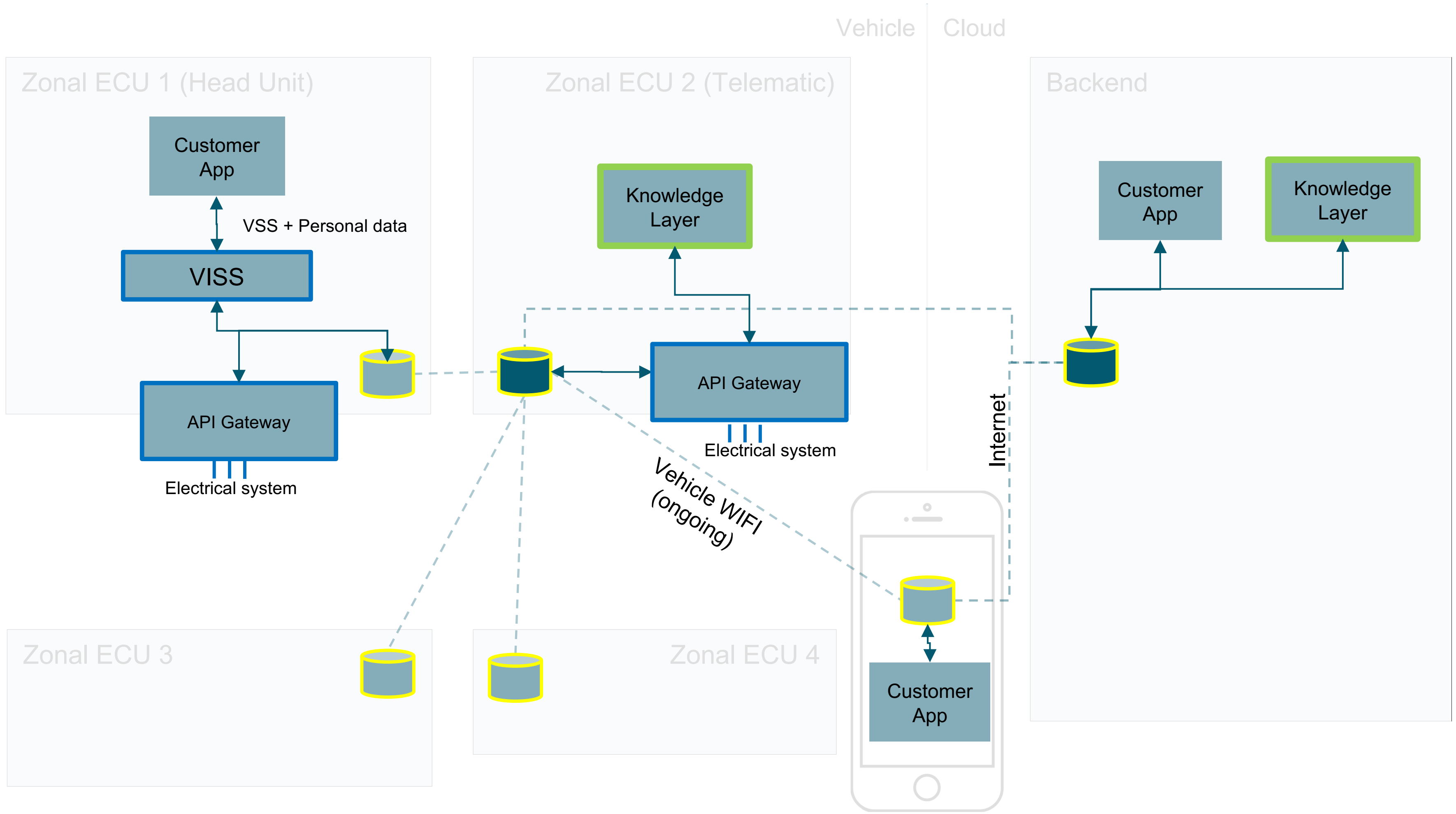
- ✅ Amount of automotive requirements which can be covered by 3rd party middleware
- ✅ Scalable VSS data model applicability tested in end to end middleware.
- ✅ Different data domains (different „trees“) can share same end to end infrastructure for sync, access, permissions …
- ⏳ Standardized data models like VSS(o) can serve as a base for AI / Knowledge generation

Optimum:

- ✅ Open source Reference Implementation for 3rd parties to align with Automotive needs
- ⏳ Performance tests (scaling, vertical and horizontal)

Outlook: next AMM?

https://wiki.covesa.global/pages/viewpage.action?pageId=71074417

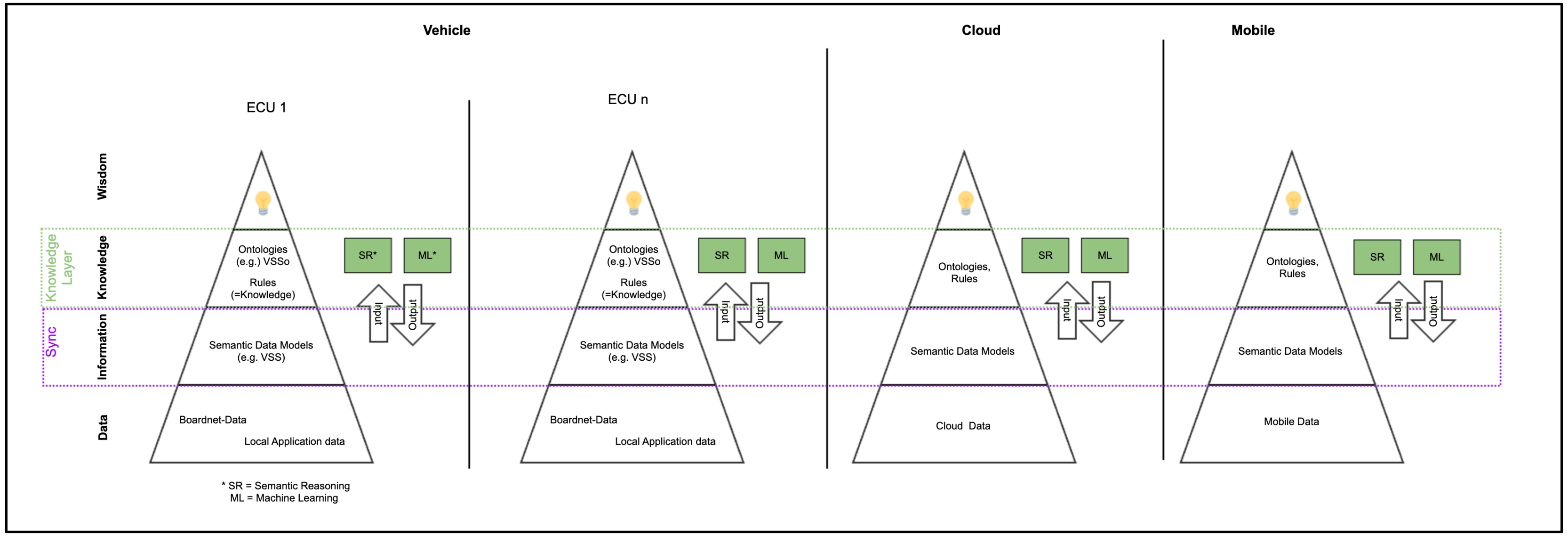


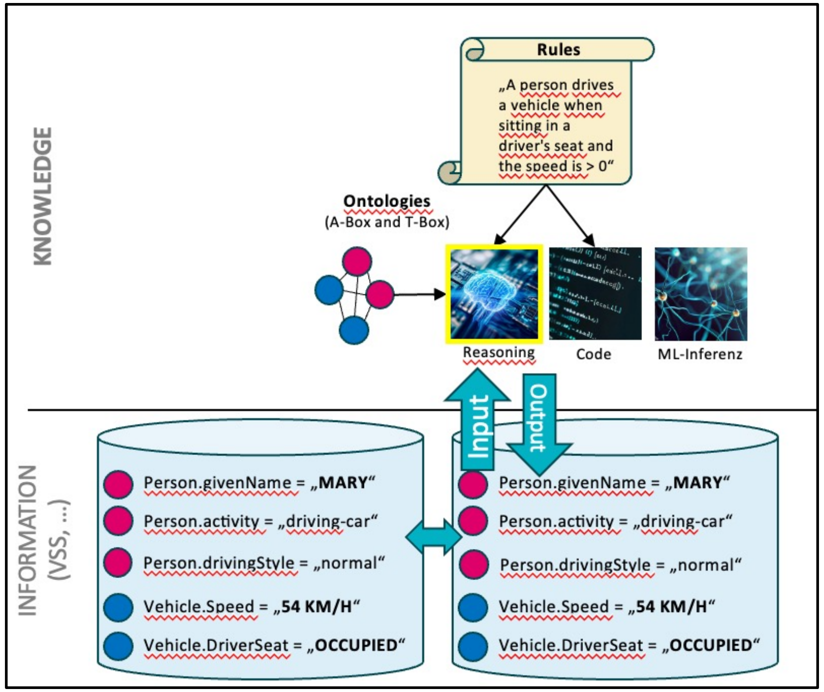
Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How (new) knowledge ...
- ... can be shared with others?

Outlook: Knowledge Layer

https://wiki.covesa.global/pages/viewpage.action?pageId=71074417




KNOWLEDGE
Rules
„A person drives a vehicle when sitting in a driver's seat and the speed is > 0“
Ontologies
(A-Box and T-Box)
Reasoning
Code
ML-Inferenz
Input
Output
INFORMATION (VSS, ...)
Person.givenName = „MARY“
Person.activity = „driving-car“
Person.drivingStyle = „normal“
Vehicle.Speed = „54 KM/H“
Vehicle.DriverSeat = „OCCUPIED“
Person.givenName = „MARY“
Person.activity = „driving-car“
Person.drivingStyle = „normal“
Vehicle.Speed = „54 KM/H“
Vehicle.DriverSeat = „OCCUPIED“

Closing remarks

End-to-end data architecture is individual task.

BUT

- Collaboration helps to understand challenges, find solutions, and avoid redundancies
- Sustainability of infrastructure requires collaborative effort
- Pool knowledge and resources for a robust foundation
- Innovative approaches create sturdy, long-term data architecture
- Transparency and active contribution encouraged for success

COVESA

Accelerating the future of connected vehicles

Thank you for your attention!

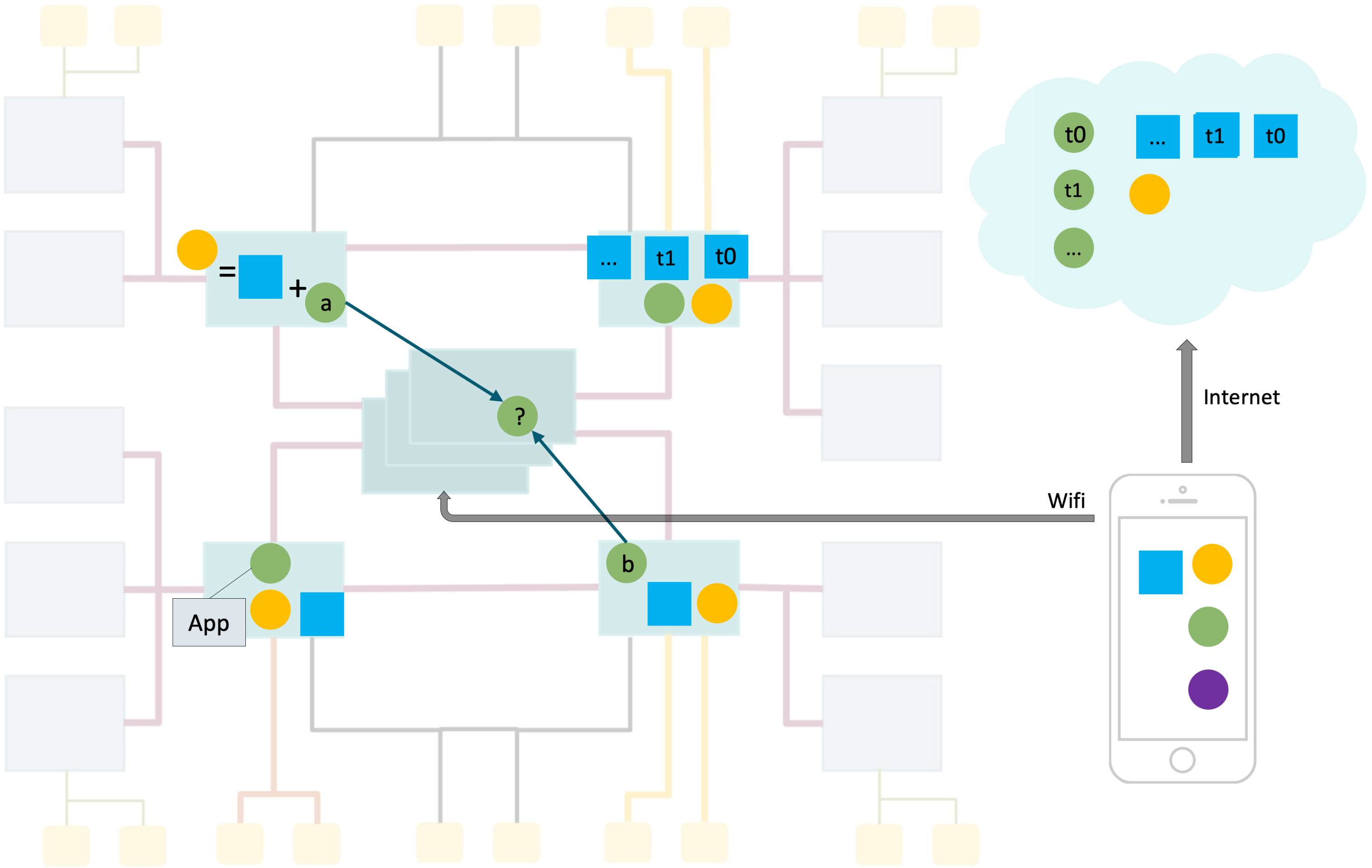
Backup

Where we started

● Vehicle Data

● Personal Data

● Other Domain Data



Key questions for a End-to-End Data Architecture:

- How data can be shared between all touchpoints?
- How different domains of data share same tech stack ?
- How should a bidirectional sync work?
- Who is responsible for conflict management?
- Who takes care about permissions, roles, rights and privacy?
- How the data model can be updated and sycned?
- How subscriptions can be handled?
- How to handle historized and time series data ...
- ...on different touchpoints?
- How to handle multiple sync endpoints?
- How to handle unidirectional data streams?
- How new knowledge ...
- ... can be shared with others?