Vehicle Signal Specification + Ontology
Status update 04/22
A good data-centric architecture is designed with data sharing in mind.

The data model is permanent and enduring, and applications come and go.

It's not something you BUY, it's something you DO!
VEHICLE SIGNAL SPECIFICATION (VSS)

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

**TAXONOMY**
FOR
ATTRIBUTES,
SENSORS AND
ACTUATORS
OF A VEHICLE.

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

VEHICLE
  - CABIN
  - DRIVETRAIN
    - DOOR
    - ISLOCKED
    - ISOPEN
  - CHASSIS
  - AXLE
    - WHEEL
    - TIRE
    - PRESSURE
  - SEAT
  - HASPASSENGER

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.

Vehicle.Drivetrain.Transmission.Speed
- type: sensor
- datatype: float
- unit: km/h
- description: The vehicle speed as measured by the drivetrain
VEHICLE SIGNAL SPECIFICATION (VSS)

YAML SPECIFICATION

Vehicle.Drivetrain.Transmission.Speed
type: sensor
datatype: float
unit: km/h
description: The vehicle speed as measured by the drivetrain

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

TAXONOMY FOR ATTRIBUTES, SENSORS AND ACTUATORS OF A VEHICLE.

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

TAXONOMY
FOR
ATTRIBUTES,
SENSORS AND
ACTUATORS
OF A VEHICLE.

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

TAXONOMY FOR ATTRIBUTES, SENSORS AND ACTUATORS OF A VEHICLE.

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
VEHICLE SIGNAL SPECIFICATION (VSS)

TAXONOMY
FOR
ATTRIBUTES,
SENSORS AND
ACTUATORS
OF A VEHICLE.

But, where is it used?
VSS in Use: KUKSA.val

- In-vehicle “digital twin” of VSS data
- Written in C++, can run (containerized) on a Vehicle Computer
- App SDK for Python and GO available
- Example data feeders for CAN and GPS data available

- 100 % Open Source
- Secured access to data
- Applications access data using a version of the W3C VISS protocol

https://github.com/eclipse/kuksa.val
MongoDB Application Data Platform for Connected Vehicle

Connected Vehicle

Cloud Data Platform

Integration Layer

Mobile / Web Clients

- Customer (customer Interaction)
  - Customer Website
  - Customer Mobile App

- Workforce (Workforce Enablement)
  - Repair / Maintenance
  - Vehicle Information

- Systems (System Integration)
  - ERP, CRM etc.

Realm DB

Realm Sync

Realm Data API

- Hot Data
  - High Availability & Horizontal Scaling
- Analytical Data
  - Real-Time, ETL free
  - Isolated Workloads
- Time Series
  - Collections and Clustered Index
- Full Text Search
  - Real-Time Search Index
- Cold Data
  - Atlas Data Lake
- Automated Tiering
  - Atlas Online Archive
- Data Locality
  - Global Clusters

Visualization & Connectors

- MongoDB Charts
- MongoDB Drivers
- BI Connector
- Kafka Connector
- Spark Connector

Mobile Data & Synchronization

- Mobile Database
- Mobile Sync

Serverless Services

- GraphQL
- RESTful
- Triggers & Events
- Scheduled Work
- Functions
- Static Hosting

Systems

- Connected Vehicle
  - MongoDB Application Data Platform
  - for Connected Vehicle

Integration Layer

- MongoDB Application Data Platform
  - Static Hosting
  - MongoDB Drivers
  - BI Connector
  - Kafka Connector
  - Spark Connector

Mobile / Web Clients

- Customer (customer Interaction)
  - Customer Website
  - Customer Mobile App

- Workforce (Workforce Enablement)
  - Repair / Maintenance
  - Vehicle Information

- Systems (System Integration)
  - ERP, CRM etc.

Cloud Data Platform

- MongoDB Application Data Platform
  - Static Hosting
  - MongoDB Drivers
  - BI Connector
  - Kafka Connector
  - Spark Connector

Integration Layer

- MongoDB Application Data Platform
  - Static Hosting
  - MongoDB Drivers
  - BI Connector
  - Kafka Connector
  - Spark Connector

Mobile / Web Clients

- Customer (customer Interaction)
  - Customer Website
  - Customer Mobile App

- Workforce (Workforce Enablement)
  - Repair / Maintenance
  - Vehicle Information

- Systems (System Integration)
  - ERP, CRM etc.
**VSS ∞ Document Model**

**Onboard Realm DB(s)**

- **VENUE**
  - **VENUE_ADAS**
    - **VENUE_ADAS_ABSSchema**: (name: 'Vehicle_ADAS_ABSSchema',
      embeds: true,
      properties: {
        Error: 'bool?',
        Inactive: 'bool?',
        IsEngaged: 'bool?',
      },
    );
  - **VENUE_ADAS_CRUISECONTRUCTION**
    - **VENUE_ADAS_CRUISECONTRUCTION**: (name: 'Vehicle_ADAS_CRUISECONTRUCTION',
      embeds: true,
      properties: {
        Error: 'bool?',
        Inactive: 'bool?',
        IsEngaged: 'bool?',
      },
    );

**Realm Sync:**
- Attribute level sync
- Compressed
- Deterministic
- Conflict resolution

**MongoDB Data Platform**

- Time Series
- Documents
- Search
- Graph
- Online Archive

**MongoDB Realm Atlas**

- **AGS Object**: (name: 'AGS',
  Error: 'false',
  Inactive: 'false',
  IsEngaged: 'false',
  SpeedSet: (name: 'SpeedSet',
    ObjectId: 'Object',
    AccelerationObject: 'Object',
    AccelerationSet: 'Object',
    AmbientAirTemperature: 'Object',
    AngularVelocityObject: 'Object',
    AverageSpeed: 'Object',
    Body: 'Object',
    CabinObject: 'Object',
    CurbWeight: 'Object',
    CurrentAcceleration: 'Object',
    CurrentOverallWeight: 'Object',
    DriveTime: 'Object',
    Driver Object: 'Object',
  );

**MongoDB Realm Serverless**

- **GraphQL**
- MQL Data API
- Field lvl. RBAC

**Onboard Realm DB(s)**

- **MongoDB**
- **MongoDB Realm Serverless**
- **MongoDB Atlas**
- **MongoDB Realm Serverless**

**Your Data Ecosystem**

**Mobile SDKs**

- **VENUE**
- **VENUE_ADAS**
- **VENUE_ADAS_ABSSchema**: (name: 'Vehicle_ADAS_ABSSchema',
  embeds: true,
  properties: {
    Error: 'bool?',
    Inactive: 'bool?',
    IsEngaged: 'bool?',
  },
);

**GraphQL Endpoint**

- **GraphQL**
- **MongoDB**
- **MongoDB Realm Serverless**
- **MongoDB Atlas**
- **MongoDB Realm Serverless**
VSS IN VEHICLE SHADOW

TAXONOMY
FOR
ATTRIBUTES,
SENSORS AND
ACTUATORS
OF A VEHICLE.

PORTAL

Unclear master system
Mixed data models
Multiple rule-based computations
New services build new adapters
Privacy hard to manage
VSS IN VEHICLE SHADOW

TAXONOMY FOR
ATTRIBUTES, SENSORS AND ACTUATORS OF A VEHICLE.

Unified Data Model
Single backend API
Improved privacy
VSS VEHICLE DATA SIMULATOR

TAXONOMY FOR ATTRIBUTES, SENSORS AND ACTUATORS OF A VEHICLE.

THURSDAY, 11 am
CVII Working Session - VISS and W3C Alignment
VEHICLE SIGNAL SPECIFICATION (VSS)

Delivering the next generation user experience,
Through an open, standardized platform
And our Mutual Ecosystem of partners:

VISS version 2 - Core
W3C Working Draft 05 April 2022
VEHICLE SIGNAL SPECIFICATION (VSS)

Great! But where are the limits?
VEHICLE SIGNAL SPECIFICATION (VSS)

TAXONOMY FOR ATTRIBUTES, SENSORS AND ACTUATORS OF A VEHICLE.

VSS offers a simple, flexible and protocol agnostic way of describing vehicle signals.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.

One type of relation not enough in many other domains.
One type of relation not enough in many other domains.

Relation to other domains with same technology stack hard to realise. Reference on existing standards not trivial (e.g. schema.org, FOAF).
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.

1. One type of relation not enough in many other domains.
2. Relation to other domains with same technology stack hard to realise. Reference on existing standards not trivial (e.g. schema.org, FOAF)
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.

1. One type of relation not enough in many other domains.
2. Relation to other domains with same technology stack hard to realise. Reference on existing standards not trivial (e.g. schema.org, FOAF).

There are existing standards of the W3C tackling those issues.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.

- One type of relation not enough in many other domains.
- Relation to other domains with same technology stack hard to realise. Reference on existing standards not trivial (e.g. schema.org, FOAF).
- Manual work of integration and mapping.

There are existing standards of the W3C tackling those issues.
PRACTICAL ISSUES USING YAML IN MULTI-DOMAIN SCENARIOS.

1. One type of relation not enough in many other domains.
2. Relation to other domains with same technology stack hard to realise. Reference on existing standards not trivial (e.g. schema.org, FOAF)

There are existing standards of the W3C tackling those issues.
If a part of a vehicle has an issue, who is the customer to inform?
If a part of a vehicle has an issue, who is the customer to inform?
If a part of a vehicle has an issue, who is the customer to inform?
If a part of a vehicle has an issue, who is the customer to inform?
If a part of a vehicle has an issue, who is the customer to inform?
VEHICLE SIGNAL SPECIFICATION ONTOLOGY (VSSo)
**Initial VSSo Development**

Based on VSS v1.0 Benjamin Klotz created within his PhD the first version of VSSo backed by Eurecom and BMW. Creation and mapping mostly manual, extensions partially automatic. Results of his work influenced the development towards VSS v2.0.

---

**VEHICLE SIGNAL SPECIFICATION ONTOLOGY (VSSo)**

![Diagram showing VSS and VSSo versions and timeline from 2019 to 2021]
VEHICLE SIGNAL SPECIFICATION ONTOLOGY (VSSo)

Initial VSSo Development

Based on VSS v1.0 Benjamin Klotz created within his PhD the first version of VSSo backed by Eurecom and BMW. Creation and mapping mostly manual, extensions partially automatic. Results of his work influenced the development towards VSS v2.0.
Initial VSSo Development

Based on VSS v1.0 Benjamin Kloz created within his PhD the first version of VSSo backed by Eurecom and BMW. Creation and mapping mostly manual, extensions partially automatic. Results of his work influenced the development towards VSS v2.0.

The W3C Automotive and Transportation Business Group has begun working on it.

VSSo

VSSo was received as a member submission end of 2020.
**Initial VSSo Development**

Based on VSS v1.0, Benjamin Kloz created within his PhD the first version of VSSo backed by Eurecom and BMW. Creation and mapping mostly manual, extensions partially automatic. Results of his work influenced the development towards VSS v2.0.

**VSSo**

VSSo was received as a member submission end of 2020.

**Proposal**

The W3C Automotive and Transportation Business Group has begun working on it.

**VSSo is listed as a deliverable**

First public working draft released

**VSS**

v1.0

2019

v2.0

2021

[https://github.com/w3c/vss](https://github.com/w3c/vss)
**Current Vehicle Data**

Find out what state a specific vehicle is in and how values of static and dynamic properties define a vehicle fleet.

*An observation is defined by the signals occurring at a certain time.*

**Dynamic Vehicle Data over Time**

Which observations have been made at which point of time? How a data stream develops over time? Can I act on it?

*Static and dynamic properties define a set of vehicles.*

*An observation is defined by the signals occurring at a certain time.*
**Current Vehicle Data**
Find out what state a specific vehicle is in and how values of static and dynamic properties define a vehicle fleet.

**Dynamic Vehicle Data over Time**
Which observations have been made at which point of time? How a data stream develops over time? Can I act on it?

**Interaction with Vehicle Data**
Get specific values using a specific protocol of a specific data provider. Agreement on data exchange with unit type, etc.

---

**A vehicle is defined by its static and dynamic vehicle properties**

**Static and dynamic properties define a set of vehicles.**

**An observation is defined by the signals occurring at a certain time.**

**A requestor needs the contexts, their properties and information how to interact with them.**

---

**VSSo**

**SSN/SOSA + VSSo**

**WoT + VSSo**
VEHICLE SIGNAL SPECIFICATION ONTOLOGY (VSSo)

Define the core **structural** concepts of VSS (e.g. Branch, Attributes, Sensors, etc.)

VSSo Core Ontology
VEHICLE SIGNAL SPECIFICATION ONTOLOGY (VSSo)

Define the core structural concepts of VSS (e.g., Branch, Attributes, Sensors, etc.)

Generate the data definitions from VSS based on the core ontology.
VEHICLE SIGNAL SPECIFICATION ONTOLOGY (VSSo)

Define the core structural concepts of VSS (e.g. Branch, Attributes, Sensors, etc.)

Use VSSo as domain ontology for other, widely adopted standards (SSN/SOSA).

Generate the data definitions from VSS based on the core ontology.

Link to other ontologies

Generated concepts
Define the core *structural* concepts of VSS (e.g. Branch, Attributes, Sensors, etc.)

Use VSSo as domain ontology for other, widely adopted standards (SSN/SOSA).

Generate the data definitions from VSS based on the core ontology.