Realizing an end-to-end vehicle-to-cloud communication framework

GENIVI Cloud & Connected Services Project

Kevin Valdek – CTO, High Mobility
Gunnar Andersson – Development Lead, GENIVI Alliance

Automotive World | May 4, 2020

GENIVI Alliance

Why is GENIVI Alliance working on a reference architecture for vehicle-data?

GENIVI has promoted Open shared software and standards and created business opportunitites in the automotive industry for over 10 years

- **2009-2015 –** Linux acceptance in automotive. Infotainment platform software and standards.
- **2016-2017 –** Cross-Domain Interaction, adapting to Multi-OS reality, safety demands
- **2018-...** **–** Multi-OS, Connected Cockpit, Virtualization,

  = Integration technologies for diverse, distributed and cloud-connected EE architectures.
- **2020-…** **–** Big-picture, end-to-end integration, adapt to latest industry trends

Example: The Cloud & Connected Services Project

Agenda

- **Project Charter**
- **Data Model**
  - Common Data Model
  - Vehicle Signal Specification — VSS
- **Communication Framework & Architecture**
  - Data Capturing & Cloud Transfer
  - Neutral Servers & 3rd Party Access
- **Proof-of-Concept Implementation**
  - Electric Vehicle Use-case
  - Timeline
- **Contributing**

Cloud & Connected Services – Project Charter

- In just a few years, connected cars will become a predominant form of automotive transportation
- A number of actors now working at breaking down the barriers for mobility services based on vehicle data to create new value
- We are currently seeing a fragmented ecosystem where different actors are using different solutions to access the data of connected cars
- In GENIVI we believe there is a need to join forces and harmonize activities when designing and implementing the full data-oriented connected vehicle architecture in order to:
  - Enable easy interoperability of building blocks, flexibility and choice
  - Develop common solutions and software
  - Enable access to all data we want to exchange
  - Control access to data
  - Enable user privacy and data security
  - Clarify actors, roles and responsibilities
  - Facilitate business opportunities and contractual agreements

Different types of vehicle data

- **Personalised vehicle data**
  - Identifiable to a specific VIN
  - Used by 3rd party services to offer tailored services to users
  - Typical delivery method point-to-point requests e.g. a REST API
- **Pseudonymized vehicle data**
  - Most identifying fields within a database are replaced with artificial identifiers, or pseudonyms
  - Neither fully anonymous nor directly identifying
  - Typically delivered in bulk in dataset from many vehicles
- **Anonymized vehicle data**
  - All identifiers, both direct and indirect are removed

Services and general data needs

- **Personalised services**
  - Retrieval of the latest cached vehicle data from the OEM server
  - Explicit consent required by the vehicle owner or the fleet owner for each individual 3rd party service
  - Event subscriptions and notifications when new data is available
  - Retrieval of historical vehicle states, e.g. the data of the last 12 hours
  - Streaming API with real-time data updates for advanced services

- **Big Data services**
  - Retrieval of bulk historical data
  - Purpose consent required by the car owner or the fleet owner
  - Analytics and histograms
  - Streaming API with configurable parameters (geolocation)

Data Model

Common Data Model

- Everyone we speak to say – **Yes, the industry needs this!**

- Potential for a shared data catalog, for a substantial subset of the vehicle data
- Shared formats, methods, and tools to manipulate the entire data set, including shared data catalog *and* future (some *proprietary*) additions
- Previous projects have come to similar conclusions – so far limited effect
- **Common data model** could enable all data-oriented use cases from **end-to-end**
- **End-to-end** could mean from *vehicle sensor* to *third-party consumer in the cloud*
- Some legacy parts will however not change – requires some translation code.

- Promote the common data model = minimize instances of data translation!

Common Data Model – prior analysis

GENIVI Cloud & Connected services has produced a **Gap-analysis document**

Includes previous major initiatives, some active, and some stopped

- **CVIM** (Automat project, completed)
- **SENSORiS**
- **ISO 20078 Extended Vehicle**
- **Android Automotive** vehicle properties
- **Vehicle Signal Specification (VSS)**

Based on the conclusion, a potential plan forward happens in several projects:

- → **GENIVI Android Automotive SIG**
- → **GENIVI Cloud & Connected services**, reference architecture
- → Continued **collaboration with W3C Automotive & Transportation group**

Vehicle Signal Specification (VSS)

- Proposed Common “Data Model” representation for Automotive (Formally, it’s a **Data Taxonomy**) – Several years in development
- Organizes “all” vehicle data in a **hierarchical tree**
- Spin-off projects work to extend it to a graph-based **Data Ontology**
- Defines **name**, **purpose**, **type**, **unit**, signal quality/reliability/sampling frequency, relationships to other data, **etc**. Layered design & extensible for **additional metadata.**
- Simple, **plain-text** based format. Easy to read, easy to write.
- Extensible tools & conversions
- Defines **standard set of signals** for whole industry. Enables also proprietary extension.
- **Adopted by W3C –** the basis for the Vehicle Information Service Specification (the Web protocol to access car data) in version 1 and upcoming “Gen 2”
- Positive feedback around the industry. We propose it should take off as *the standard way* to describe vehicle data.

Vehicle Signal Specification – format

Plain-text format is the only way to go. Future proof.

The choice must be both of these things!

1) Computer-processable and convertible to any other useful format
2) Easy to read and write

Structured text in the “YAML” format fits the bill!

- Data Analysts find it easy to read and write
- Developers can handle it like “code”
- Trivially convertible to other computer program preferred formats, (e.g. JSON)

Vehicle Signal Specification – structure

Branches (data organization)

- Vehicle.Body.Lights.

Attributes (fixed value)

- Vehicle.Drivetrain.InternalCombustionEngine.**MaxTorque**

Signals (variable)

- Vehicle.Drivetrain.FuelSystem.**AverageConsumption**

Actuators

- Vehicle.Cabin.Sunroof.Shade.**Position**

Vehicle Signal Specification – example

```
Vehicle.Body.Windshield.Front.WasherFluid.Level:
  - datatype: uint8
  - unit: percent
  - type: sensor
  - description: Washer fluid level as a percent.
                 0 = Empty. 100 = Full.
```

File organization is also hierarchical and can be version controlled in separate files:
E.g.: If the file **Vehicle/Body/Windshield.vspec** defines **Front.WasherFluid.Level**
it *can* result (this is also flexible) in: `Vehicle.Body.Windshield.Front.WasherFluid.Level`

Additional metadata can be added (*or substracted or modified*) in any number of separate VSS layers

VSS2 – new features

- The terms VSS and VSS2 are used mostly interchangeably.
- In daily conversation we say “VSS2” to highlight that there have been some format changes and new features accepted for the upcoming version 2.0:
  - More efficient instantiation of similar nodes, reducing redundancy in tree definition
  - Slight modification of the format for more standard usage of YAML
  - “Array” data types (details still under discussion)
  - These changes are significant enough to highlight the new major version number

VSS2 – Summary

- The Vehicle Signal Specification provides two complementary and separately useful things:

1) A potentially industry-wide common data catalog

- A starting point for *widely applicable* data-related development
- Essentially forms a standard “API” for vehicle data access
- Adopted by W3C official automotive communication protocols

2) A shared description format, methods, tools, code-generators and binding libraries to communication technologies

- Note the advantage of this, even for proprietary extensions of the data tree!

Value Exchange Formats

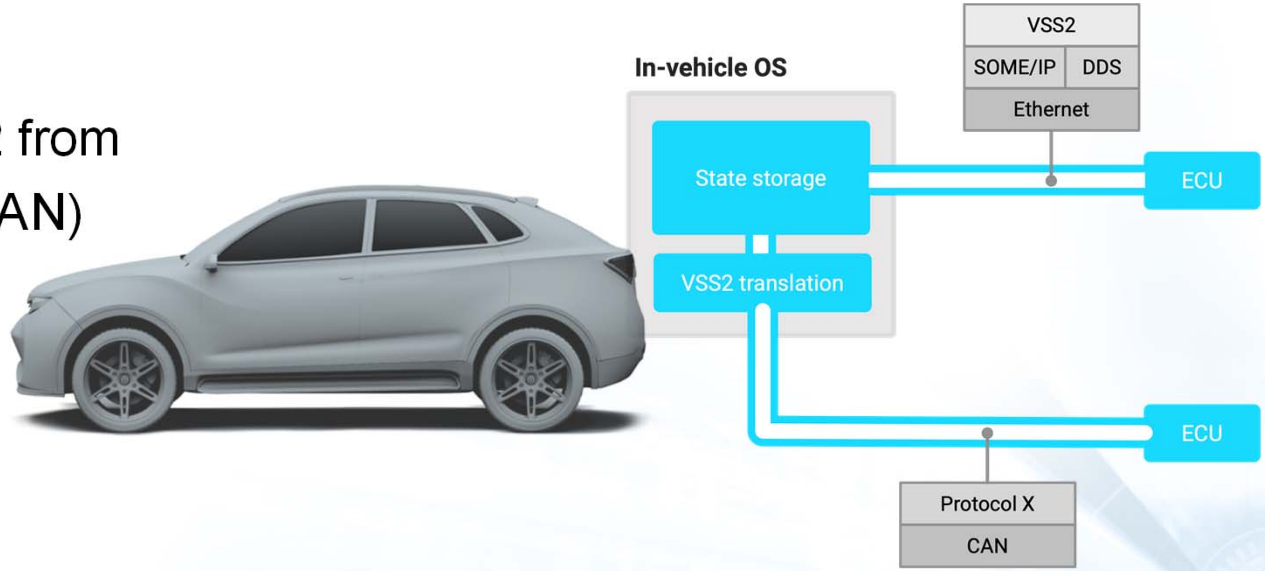
- VSS defined the **Model** and **Taxonomy** of Vehicle Data items
- We must add: data encoding, formats, protocols, bindings
- **Value Exchange Formats** – analyzes message formats for data exchange
- → Define a High-level, *abstract*, “protocol” to exchange data
- Promote alignment of naming, common understanding of terms
- Support on-demand values, bundles of values, time-series, snapshots of related data (freeze-frames), and “edge processed” values such as statistics
- Influence from:
  - SENSORiS, CVIM (Automat)
  - W3C Geospatial, and other standards
- Can act as inspiration to *concrete* protocols, e.g. W3C Gen2

Communication Framework & Architecture

GENIVI®

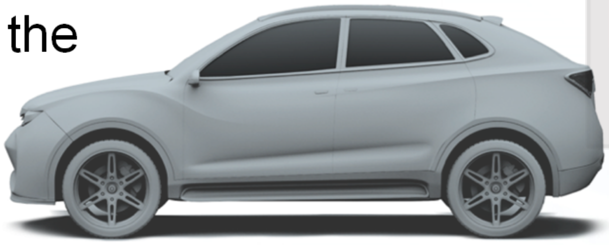
In-vehicle data capturing

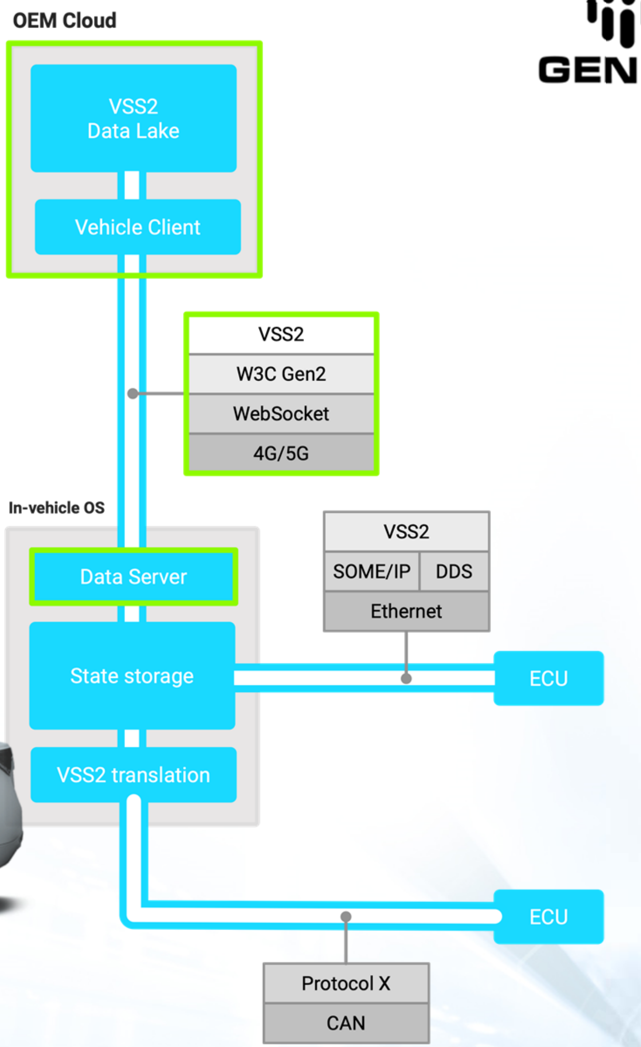
- VSS2 data from ECUs collected through typical in-vehicle protocols e.g.
  - SOME/IP (Scalable service-Oriented MiddlewarE over IP)
  - DDS (Data Distribution Service)
  - Web protocols (HTTP/REST)
- Data protocol translation to VSS2 from proprietary data encoding (e.g. CAN)
- Local vehicle data storage



OEM Cloud transfer and storage

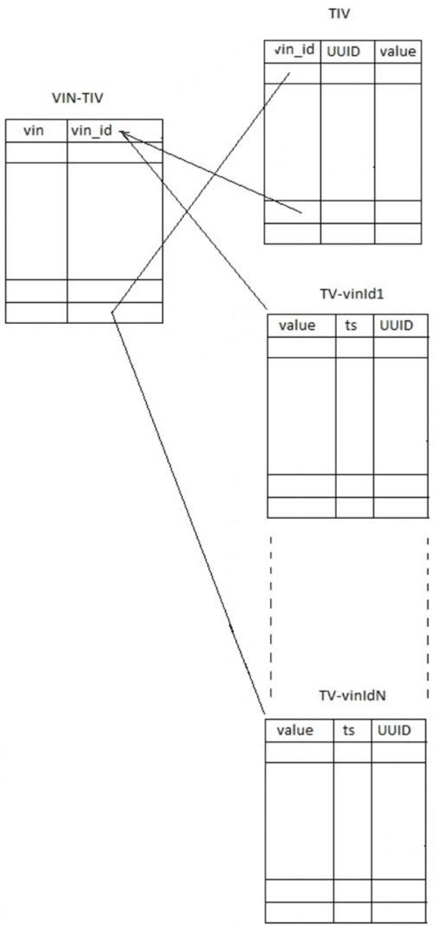
- Gen2 is the second generation of a vehicle API developed by the W3C automotive group
- Data transport over HTTPS (REST API) or secure WebSockets using the VSS2 data model
- The vehicle acts as a data server
- OEM Cloud acts as a data client
- Database schema created out of the VSS2 specification within the project workgroup





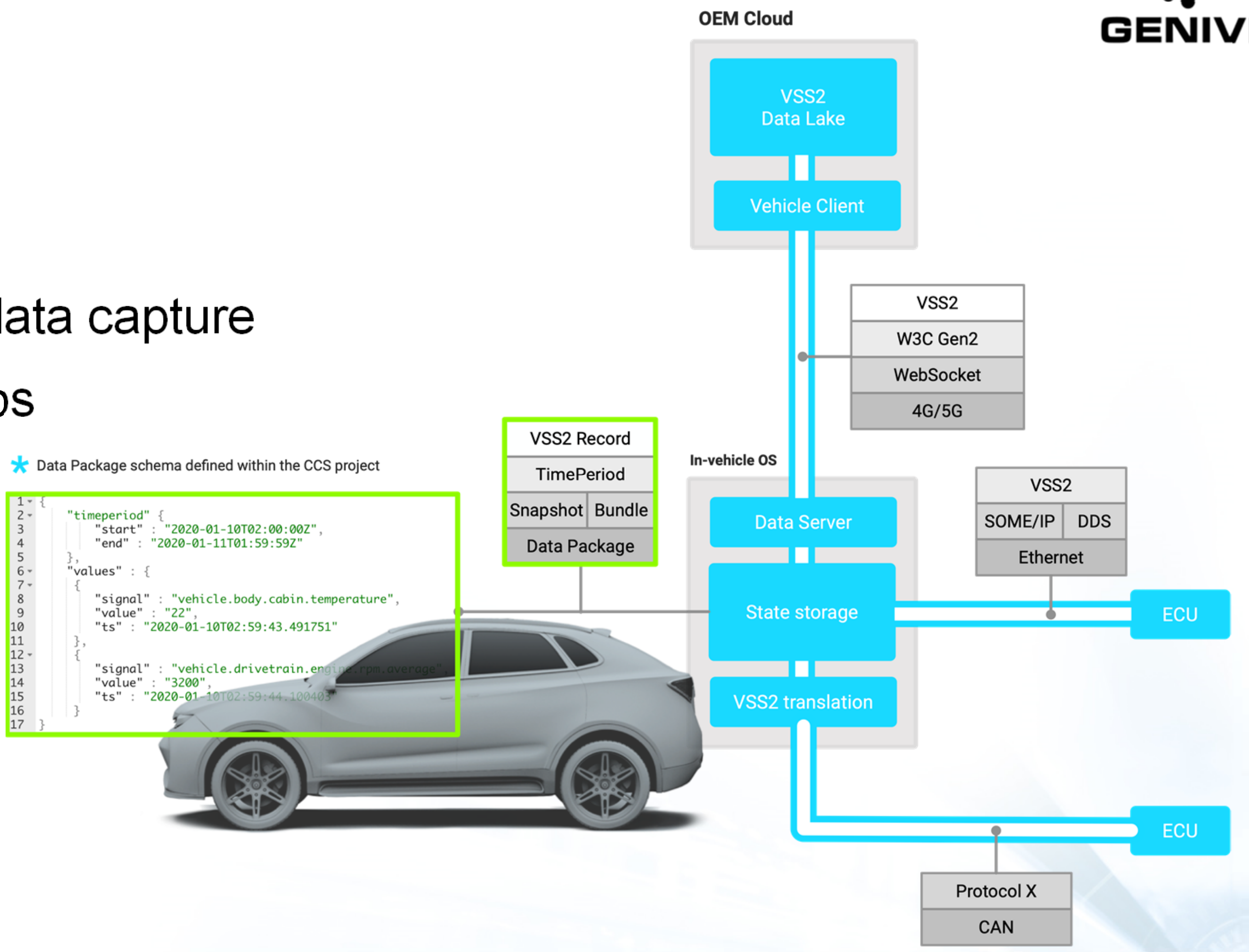
OEM Cloud storage

- Data Lake based on object storage
- Proof-of-Concept scope:
  - VSS-to-DB-adapter
    - Maps from VSS paths to database tables
    - Provides access to VSS metadata
  - Relational database
    - Provides a logical link between VIN and all leaf node data points
    - For **T**ime-**I**n**V**ariant data points (TIV tables)
      - Attribute nodes
    - For **T**ime-**V**ariant data points (TV tables)
      - Sensor/Actuator/.. nodes



Vehicle data buffering

- Necessary to buffer data:
  - Poor connectivity scenarios
  - Scenarios of high-frequency data capture
  - Scenarios of complex data jobs
  - Statistical measurements

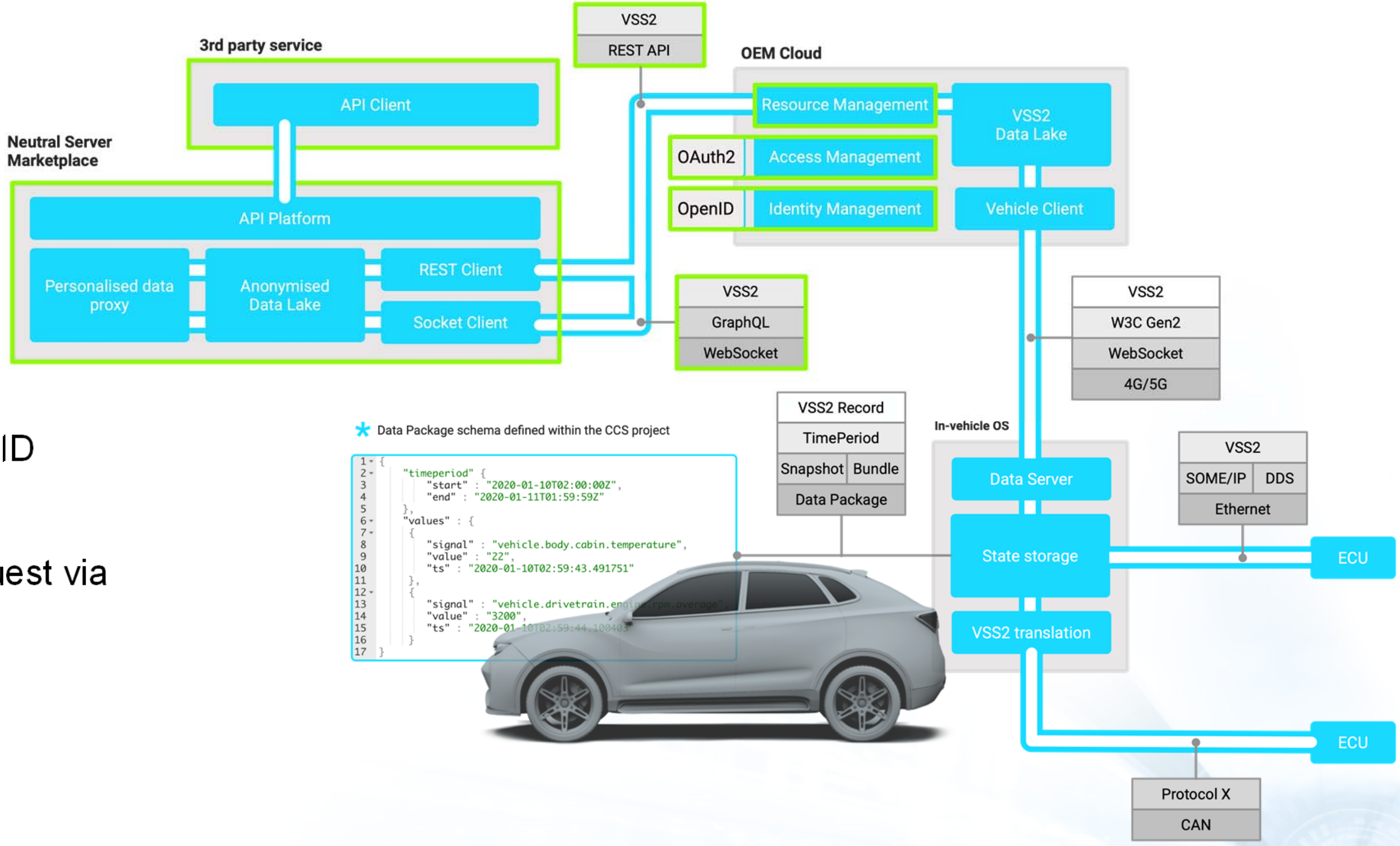


* Data Package schema defined within the CCS project

```
{
    "timeperiod" {
        "start" : "2020-01-10T02:00:00Z",
        "end" : "2020-01-11T01:59:59Z"
    },
    "values" : {
     {
        "signal" : "vehicle.body.cabin.temperature",
        "value" : "22",
        "ts" : "2020-01-10T02:59:43.491751"
     },
     {
        "signal" : "vehicle.drivetrain.engine.rpm.average",
        "value" : "3200",
        "ts" : "2020-01-10T02:59:44.100403"
     }
}
```

Neutral Servers & Data Marketplaces

- VSS2 data delivery from the OEM Cloud to:
  - Neutral Servers
  - Data Marketplaces
- ISO 20078 Extended Vehicle principles taken into account
- Car owner authentication with OpenID
- Car owner consent with OAuth2
- Both poll/push and filtered data request via common web technologies:
  - REST API
  - GraphQL
  - Socket feed

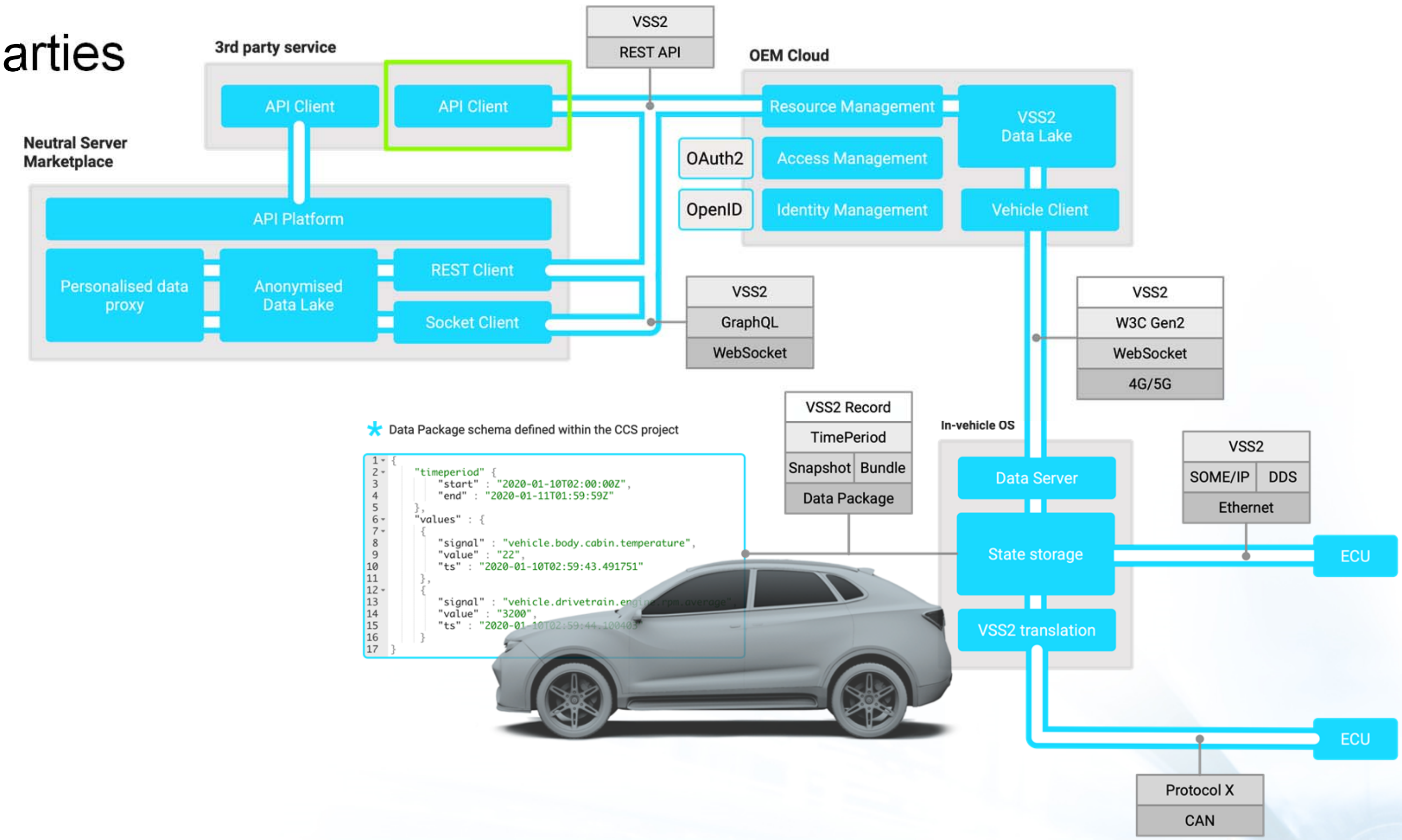


* Data Package schema defined within the CCS project

```
{
    "timeperiod" {
        "start" : "2020-01-10T02:00:00Z",
        "end" : "2020-01-11T01:59:59Z"
    },
    "values" : {
     {
        "signal" : "vehicle.body.cabin.temperature",
        "value" : "22",
        "ts" : "2020-01-10T02:59:43.491751"
     },
     {
        "signal" : "vehicle.drivetrain.engine.rpm.average",
        "value" : "3200",
        "ts" : "2020-01-10T02:59:44.100403"
     }
}
```

3rd party data access

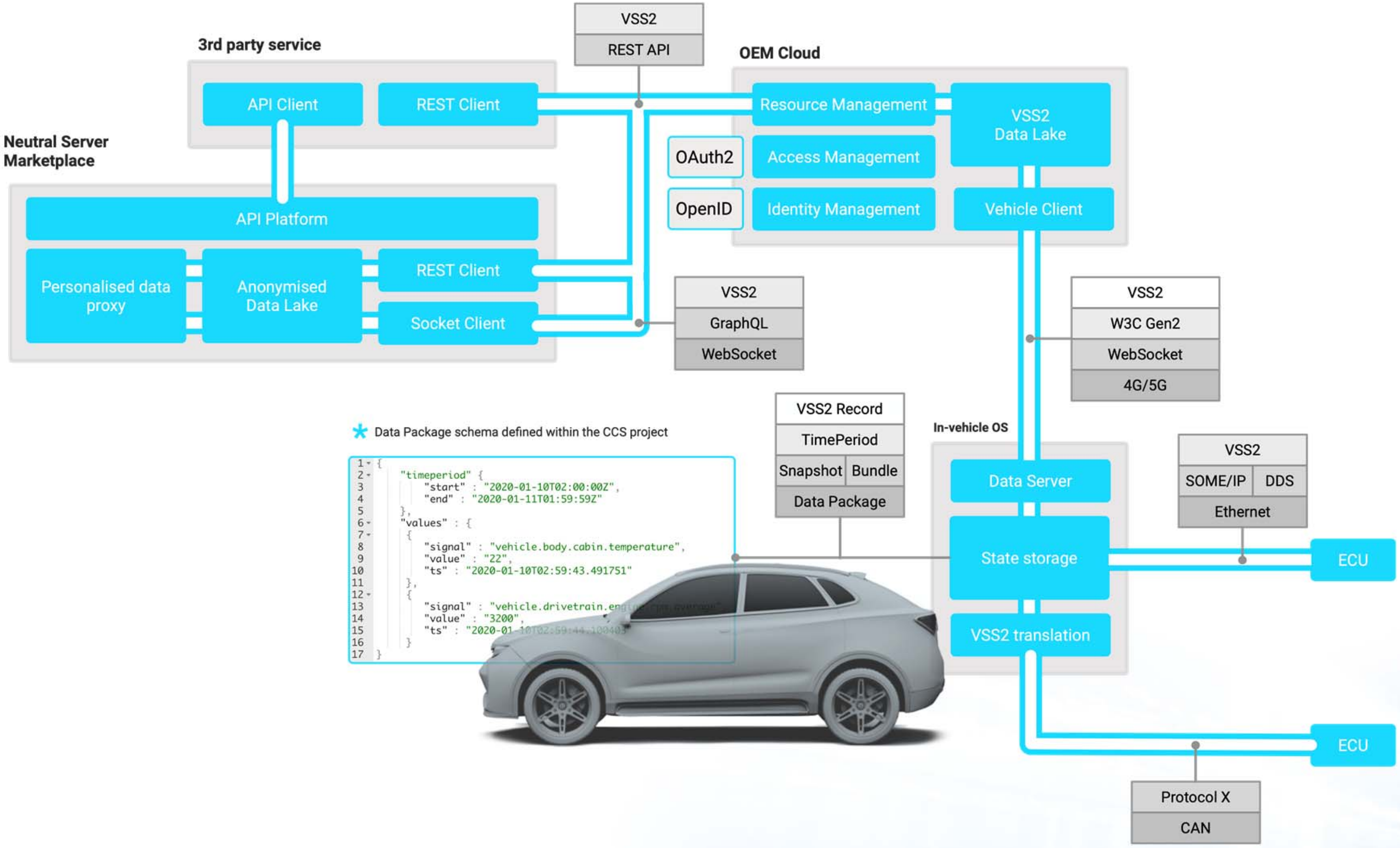
- Direct data access by 3rd parties in certain scenarios:
  - Framework flexibility
  - Safety critical
  - Business critical



* Data Package schema defined within the CCS project

```
{
    "timeperiod" {
        "start" : "2020-01-10T02:00:00Z",
        "end" : "2020-01-11T01:59:59Z"
    },
    "values" : {
     {
        "signal" : "vehicle.body.cabin.temperature",
        "value" : "22",
        "ts" : "2020-01-10T02:59:43.491751"
     },
     {
        "signal" : "vehicle.drivetrain.engine.rpm.average",
        "value" : "3200",
        "ts" : "2020-01-10T02:59:44.100403"
     }
}
```

Communication Framework & Architecture



* Data Package schema defined within the CCS project

```
{
    "timeperiod" {
        "start" : "2020-01-10T02:00:00Z",
        "end" : "2020-01-11T01:59:59Z"
    },
    "values" : {
     {
        "signal" : "vehicle.body.cabin.temperature",
        "value" : "22",
        "ts" : "2020-01-10T02:59:43.491751"
     },
     {
        "signal" : "vehicle.drivetrain.engine.rpm.average"
        "value" : "3200",
        "ts" : "2020-01-10T02:59:44.100405"
     }
}
```

May 2020 |

Proof-of-Concept implementation

GENIVI®

Proof-of-Concept objectives

- To get fast results in evaluating and confirming the communication framework
- To identify the technologies for each component, and which APIs that need to be standardised
- To introduce changes in the framework based on real life experience
- To gather experience that can be used to build a reference architecture as a future step

- Important: software-based architecture only

Demonstrated use-case

- Objective is to show Electric Vehicle (EV) charging data points

Updated set of data points submitted to VSS and W3C Automotive WG for incorporation into the data tree: (Pull request: https://github.com/GENIVI/vehicle_signal_specification/pull/154)

- Example of EV data points

Vehicle.Powertrain.BatteryManagement.**ChargingInlet**
Vehicle.Powertrain.BatteryManagement.Battery.**Temperature**
Vehicle.Powertrain.BatteryManagement.Battery.**NominalVoltage**
Vehicle.Powertrain.BatteryManagement.Battery.**CurrentCapacity**
Vehicle.Powertrain.BatteryManagement.Battery.**ChargeEnergy**
Vehicle.Powertrain.BatteryManagement.Battery.**ConsumedEnergy**
Vehicle.Powertrain.BatteryManagement.Battery.**BrakingEnergy**
Vehicle.Powertrain.BatteryManagement.Battery.**StateOfCharge**

Timeline

- Milestone 1 – GENIVI Virtual Technical Meeting (12-14 May)
- Milestone 2 - Internal milestone (early Q3 - July)
- Milestone 3 - Fall All Member Meeting, Leipzig, Germany (last week of October)
- Milestone 4 - CES 2021, Las Vegas, USA (early January 2021)

- GENIVI Virtual Technical Meeting: register at https://www.eventleaf.com/geniviVTS

Contributing

- **Weekly telcos**
  - Mondays - 11:30 CET (Asia friendly time) – Communication Framework
  - Mondays - 17:00 CET (US friendly time) - Vehicle Data, sprint & backlog review
  - Wednesdays - 17:00 CET (US friendly time) - Communication Framework
  - Mailing list: http://genivi.emwd.com/mailman/listinfo/ccs_lists.genivi.org
  - Cloud & Connected Services Project Wiki : https://at.projects.genivi.org/wiki/x/PIAVAg
  - Cloud & Connected Services Proof-of-Concept Work Breakdown: https://at.projects.genivi.org/wiki/x/84AkAw

Thank you!

Visit GENIVI:

http://www.genivi.org

http://projects.genivi.org

Contact us:

help@genivi.org