VSS-VSC alignment
Similarities and Differences

- **VSS**
  - Data meta model
  - Often implied that there are set/get/subscribe-methods, but not stated as part of VSS
    - This can be seen as a VSC service to set/get/subscribe to VSS datapoints
  - Does not specify error messages, authorization, ...
    - But specified by e.g. VISS
  - Hierarchical order by branches

- **VSC**
  - Methods API –
  - Possibility so specify more detailed type of errors
  - Hierarchical order by namespace
VSS-VSC alignment

Possible touchpoints

- An implementation may want to use the same interface for both
  - Same protocol (e.g. MQTT, Websocket, Some/IP, gRPC)
  - Same authentication model
- Could also be an advantage to have VSS and VSC resources addressing same technical area at the same API/path in e.g. code

```java
Vehicle vehicle = new Vehicle();
// All things available with horn existing in Vehicle.body.horn
// Both VSS
vehicle.body.horn.isActive.set(true);
sleep(2);
vehicle.body.horn.isActive.set(false);
// As well as possible VSC methods
result = vehicle.body.horn.playMelody("La Cucaracha");
```
VSS-VSC alignment

But how?

- Generate "common" files using both VSS and VSC as input
  - E.g. FrancaIDL, GRPC, BAMM, ...
- Define "common" syntax, e.g. syntax similar to VISS also for VSC calls over MQTT/Websocket/HTTPS
VSS Layers

Introduction

- Why?
  - Add your own signals
  - Change instantiation information (e.g. number of seats)
  - Add meta-data (e.g. Accuracy, Authorization or DBUS)
  - Change datatype/scaling/offset

- Possible Use-cases
  - VSS Vspec files to initiate (internal) database
  - VSS Vspec files to configure internal communication channels or internal converters

- Ideas:
  - Need to agree on meta-data for various use-cases, e.g. authorization?

- A bit related
  - (Recommended) Meta-data for observations, like timestamps