VSSo: the Vehicle Signal and Attribute Ontology

Benjamin Klotz, Raphaël Troncy, Daniel Wilms and Christian Bonnet

klotz@eurecom.fr
Context

- Front camera
- Radar
- Tire pressure sensor
- Park assistant
- Oil temperature sensor
- Steering angle sensor
- Wheel speed sensor
- Blind spot detection

- Adapt infotainment based on cloud usage, favorite playlists and volume
- Drive your car in front of the building you are leaving
- ...

{"name":"accelerator_pedal_position","value":0,"timestamp":1361454211.483000}
{"name":"fuel_level","value":23.478279,"timestamp":1361454211.485000}
{"name":"torque_at_transmission","value":1,"timestamp":1361454211.488000}
Traditional development approach

Car fleet
- Generation N - 1
- Generation N
- Generation N + 1
- Generation N + 2

Unique interface

OEM backend

Cloud

BMW ConnectedDrive
Requirements [1]

Get information about attributes and signals on connected vehicles:

Telematics/fleet management
- What type of fuel does this car need?
- What is the current gear?

Garage/diagnosis
- What type of transmission does this car have?
- How many different speedometers does this car contain?

E-commerce
- What is the model of this car?
- How old is this car?

Seamless experience
- What are the destination coordinates?
- What is the local temperature on the driver side?

Vehicle signal specification (VSS) [2]

Figures (Apr 2018):
- 451 branches
- 1103 leaves:
  - 43 attributes
  - 1060 signals: including
    - (700 seat-related),
    - 268 with unit

Examples:
- Gearbox-sensed speed: `.Drivetrain.Transmission.Speed`
- Engine speed: `.Drivetrain.Engine.Speed`
- GPS-sensed speed: `.Cabin.Infotainment.Speed`

**SOSA/SSN observation & sensor pattern**

- Definition of a signal
- Definition of a sensor
- Well-defined units
- Geolocation

**BUT**

No formal definition of:
- “speed” or other observable properties
- “speedometer” or other car sensors/actuators
- “car” or vehicle parts
VSSo: a Vehicle Signal Ontology

Map to existing Ontologies
- SSN/SOSA
- QUDT (unit)
- custom datatypes [4]

Generate definition of VSS concepts
Fixing problems
Manually validate and clean the generated ontology

Add sensors and actuators

Modeling issues and resolutions
1. VSS concepts have NOW unique names
2. All signals are attached to (virtual) sensors or actuators
3. All branches are part of the top “vss:Vehicle” branch
4. All position-dependent branches have a property “position”

#1: Uniqueness of names

- Some signals represent the same phenomenon, but sensed by different sensors
  - Ex: Drivetrain.Speed (sensed by the gearbox) and Infotainment.Speed (sensed by the GPS)

"vss:VehicleSpeed" is a unique phenomenon observed by different sensors Producing different signals

Names are clarified to avoid homonymy
#2: Signals are attached to (virtual) sensors or actuators

Some signals are attached to sensors, other to actuators, or both. For instance:

- **Drivetrain.Transmission.Speed** observes a speedometer
- **Mirror.Right.Heating** only acts on a heater
- **Infotainment.Media.Volume** observes/acts on the media volume
#3: Branches are `vss:part of vss:Vehicle`

Current VSS structure

“Signal”/“Attribute” are the name of the top element

VSSo structure

`vss:Vehicle` is the top element containing all branches
#4: position ≠ branch

- Ex: Door.Left.IsLocked, Mirror.Right.Tilt
- Branches vss:Door and vss:Mirror have vss:hasPosition property with limited potential values ("Left", "Right", "Row1", …)
VSS private branch

Private OEM-specific ontology cookbook:

1. Write VSS-compliant specification of private concepts (new signals, attributes and branches)
   - Follow the VSS policies just as when creating a private branch
2. Generate the ontology using the existing tool
3. Validate the ontology
   - Check the unicity of concepts and definitions (in the private branch and if possible with VSSo)
4. Define a private namespace for your ontology integrating VSSo
Evaluation

• VSSo expressivity: requirements can be fulfilled with SPARQL queries
  – *What are the dimension of this car?*
  – *What is the current temperature on the driver side?*

• VSSo extension mechanism is currently under test with real use cases at BMW
  – PoI definitions in the GPS and distance to the destination
  – New sensors for sign recognition (e.g. speed limit)
Applications [5,6]

Interact with a Smart Home

Generate semantic trajectories


http://automotive.eurecom.fr/trajectory
Conclusion

VSSo: a Vehicle Signal and Attribute ontology

- 483 classes (~300 signals); 63 properties (~50 attributes)
- Documentation: http://automotive.eurecom.fr/vsso (v1.1)
- Recommended prefix: vss
- Re-use SSN/SOSA modeling patterns
- Suitable for annotating things in the Web of Things as well as semantic trajectories
Future work

• Potential standard for the W3C automotive WG [7]
• Extensions need more documentation and concrete open examples
• Tools/converters to generate VSSo data from real car sensor data
• The SPARQL endpoint has proven to be inadequate for most cases (needless complexity).

Find what is more adapted for this domain and community

[7] https://www.w3.org/auto/wg/wiki/Main_Page
Thank you for your attention

Do you have questions?

Contact:
Benjamin Klotz
klotz@eurecom.fr
W3C Automotive Working Group [a]

- **Goal**: create specification protocols and APIs to expose vehicle data and information from an automotive network buses to a Web application.

- **Candidate recommendation**: Vehicle Information Server Specification (VISS)
  - Websocket (interest of HTTPS/REST)
  - Based on the Vehicle Signal Specification (VSS)

- **Neutral Vehicle** [b]: platform to provide an end-to-end framework for transferring rich vehicle data from the ground to the cloud and back.

[a] https://www.w3.org/auto/wg/
[b] https://neutralvehicle.com/
A description of a vss:Branch

vss:Branch a rdfs:Class, owl:Class;
rdfs:label "Branch"@en;
rdfs:comment "Branch of the vehicle. Either a component (Body, Chassis...) or the complete vehicle"@en.

vss:ObstacleDetection a rdfs:Class, owl:Class;
rdfs:subClassOf vss:Branch;
rdfs:label "ObstacleDetection"@en;
rdfs:comment "Signal/Attribute.ADAS.ObstacleDetection: Signals form Obstacle Sensor System"@en;
rdfs:subClassOf [  
a owl:Restriction;
  owl:onProperty vss:partOf;
  owl:allValuesFrom vss:ADAS
 ];
rdfs:subClassOf [  
a owl:Restriction;
  owl:onProperty vss:hasSignal;
  owl:allValuesFrom [owl:unionOf vss:ObstacleDetectionIsActive, vss:ObstacleDetectionError] ]
A description of a vss:attribute

vss:attribute a owl:ObjectProperty;
  rdfs:label "Attribute"@en;
  rdfs:comment "Attribute signals that do not change during the power cycle of a vehicle."@en;
  rdfs:domain vss:Branch.

vss:driveType a owl:DatatypeProperty;
  rdfs:subPropertyOf vss:attribute;
  rdfs:label "DriveType"@en;
  rdfs:comment "Attribute.Drivetrain.Transmission.DriveType: Drive type."@en;
  rdfs:domain vss:Transmission;
  rdfs:range [ owl:oneOf("unknown"@en "Front-wheel drive"@en "Rear-wheel drive"@en "All-wheel drive"@en)].
A description of a vss:Signal

vss:ObservableSignal a rdfs:Class, owl:Class;
  rdfs:subClassOf sosa:ObservableProperty;
  rdfs:label "Observable signal"@en;
  rdfs:comment "All observable signals that can dynamically be updated by the vehicle"@en.

vss:AmbientAirTemperature a rdfs:Class, owl:Class;
  rdfs:subClassOf vss:ObservableSignal;
  rdfs:label "AmbientAirTemperature"@en;
  rdfs:comment "Signal.Vehicle.AmbientAirTemperature: Ambient air temperature"@en;
  rdfs:subClassOf [
    a owl:Restriction;
    owl:onProperty sosa:isObservedBy;
    owl:allValuesFrom vss:Thermometer
  ];
  rdfs:subClassOf [
    a owl:Restriction;
    owl:onProperty qudt:unit;
    owl:allValuesFrom vocab:DegreeCelcius
  ].