BlackBerry®

BLACKBERRY IVY AND COVESA

2023-09-19

How IVY manages signals generated in a vehicle and how Software Sensors identify what signals they need in order to implement their Insights

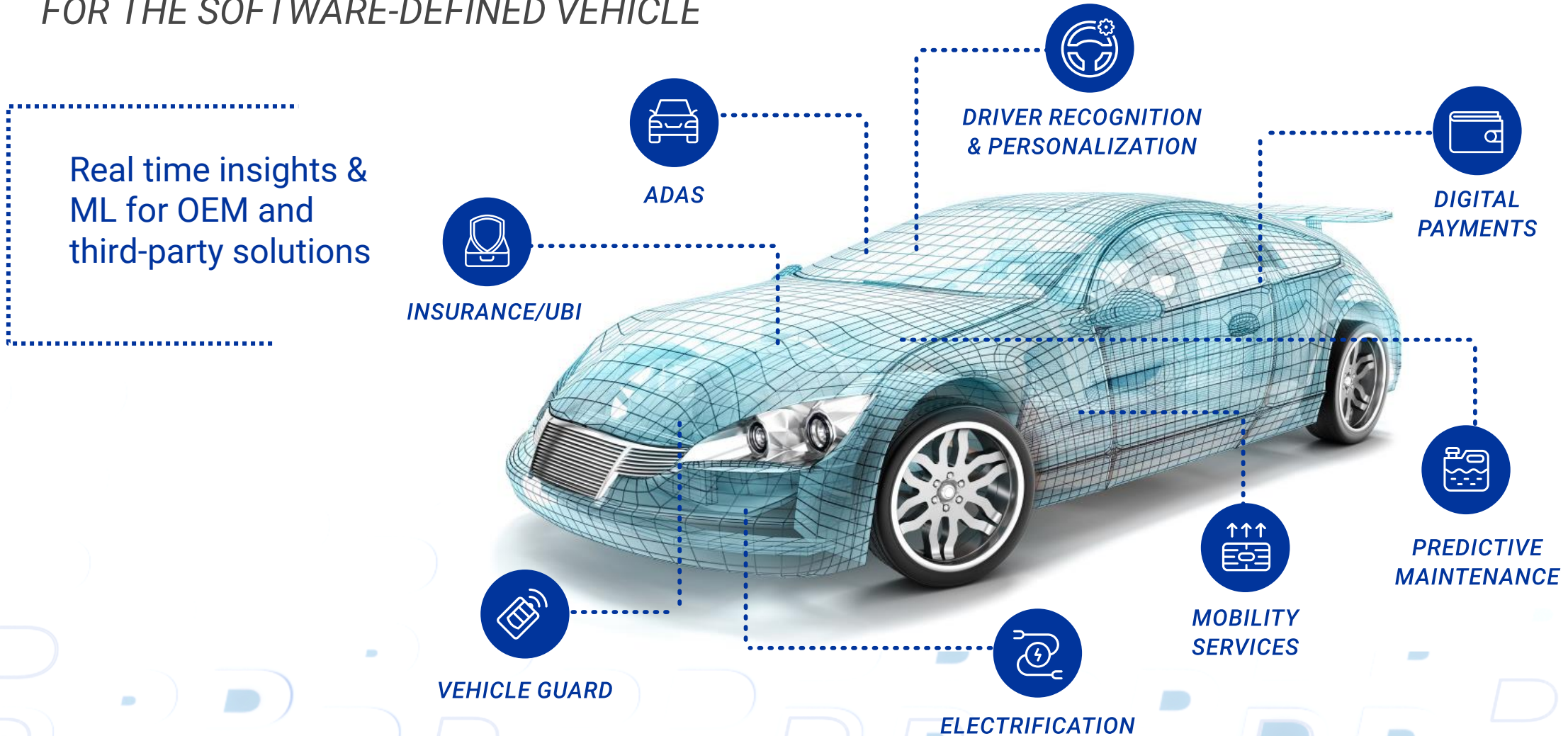Pierre Pierre Blais
Distinguished Software Architect

Jasmin Mulaosmanovic
Senior Director, Product Management

AGENDA

▶ Intro to IVY and QNX

▶ Why COVESA VSS?

▶ IVY VSS Enhancements

▶ Discussion

BlackBerry IVY™

A FOUNDATIONAL DATA INSIGHTS PLATFORM FOR THE SOFTWARE-DEFINED VEHICLE

Real time insights & ML for OEM and third-party solutions

ADAS

INSURANCE/UBI

DRIVER RECOGNITION & PERSONALIZATION

DIGITAL PAYMENTS

VEHICLE GUARD

ELECTRIFICATION

MOBILITY SERVICES

PREDICTIVE MAINTENANCE

BLACKBERRY QNX IS THE WORLD'S #1 AUTOMOTIVE SOFTWARE SUPPLIER

270+
Vehicle Makes & Models

10/10
Top Automakers

24/25
Top EV Automakers

235M+
Vehicles

100%
SOP Deadlines

45+
Automakers

7/7
Top Tier-1s

DRIVING THE SOFTWARE-DEFINED VEHICLE

Increase emphasis on functional safety & security

Implement with velocity

Harness the power of the cloud and the edge

CONTINUALLY EVOLVING AUTOMOTIVE BUSINESS CHALLENGES

Reduce complexity & cost

Enhance user experience

Facilitate new revenue models

IVY SIMPLIFIES LAUNCHING NEW SERVICES BY PROVIDING APPLICATIONS WITH THE VEHICLE DATA & INSIGHTS THEY NEED

IVY Provides automakers:

- Secure access to normalized data across multiple vehicle domains
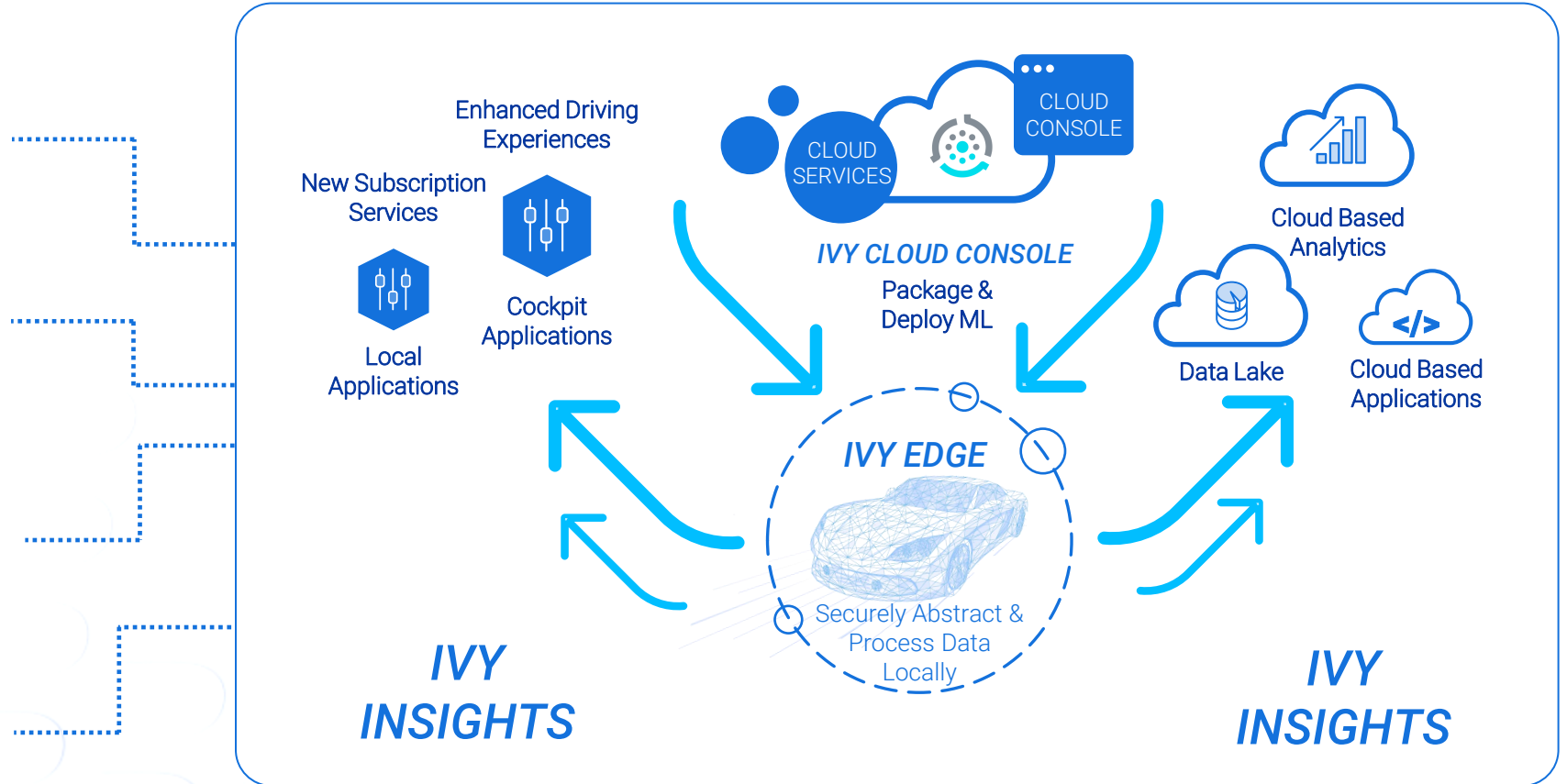
- Cloud deployment of ML software sensors in the vehicle

- The ability to process data on the edge to generate actionable insights

- Enhanced services & applications using rich data insights

Enhanced Driving Experiences

New Subscription Services

CLOUD SERVICES

CLOUD CONSOLE

IVY CLOUD CONSOLE
Package & Deploy ML

Cloud Based Analytics

Cockpit Applications

Local Applications

Data Lake

Cloud Based Applications

IVY EDGE

Securely Abstract & Process Data Locally

IVY INSIGHTS

IVY INSIGHTS

BlackBerry IVY™

FASTER INNOVATION

- Accelerate product development & delivery
- Lower developer friction

IMPROVED ECONOMICS

- Lower operating and data costs
- Turn upfront investment into a variable expense

IN-VEHICLE EXPERIENCES

- Update services throughout the vehicle lifecycle
- Enable monetizable services

CONTROL OF TECHNOLOGY STACK

- OEM control of data and permissions
- Agonistic to underlying cloud infrastructure, OS and hardware
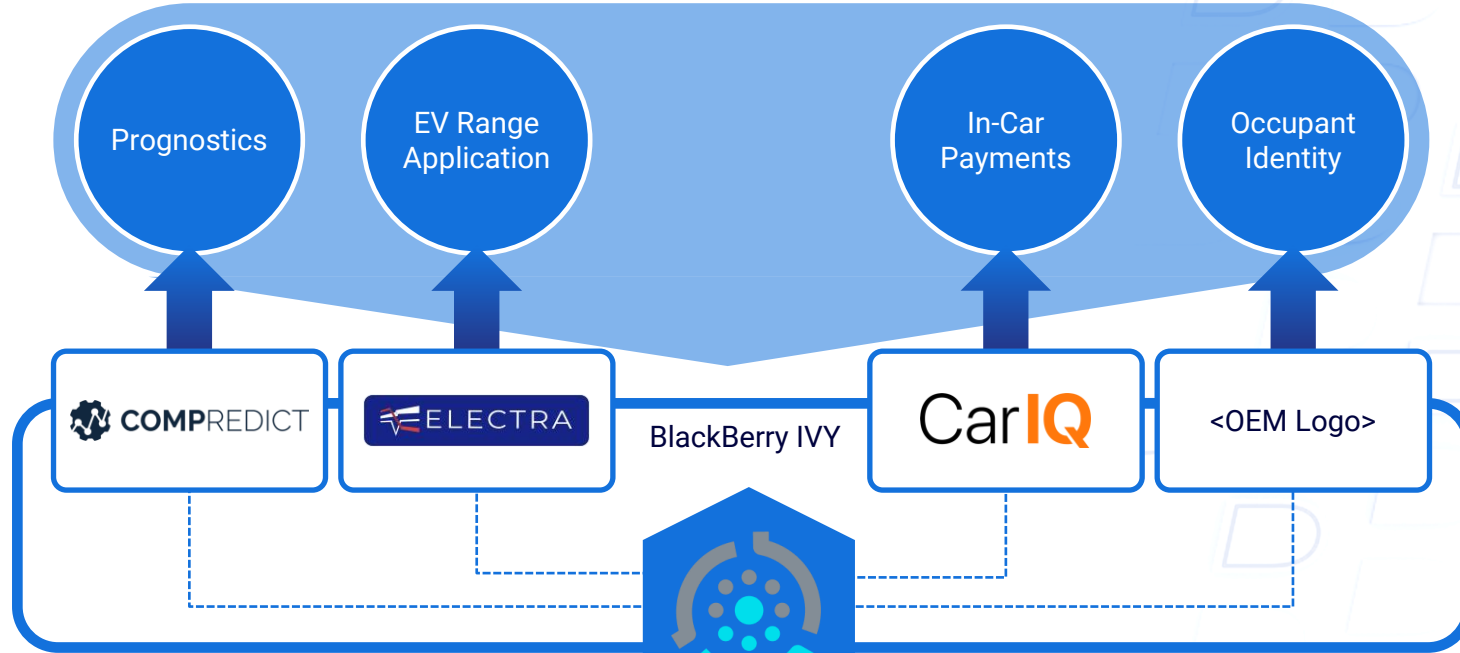
SCALE & ECOSYSTEM

- Single platform for all auto domains
- Access to solution ecosystem

BlackBerry IVY™ — ECOSYSTEM

REDUCE UPFRONT INVESTMENT AND ACCELERATE YOUR ROADMAP

IVY PARTNERS

here · alexa · RIDECELL · cerence · COMPREDICT · ELECTRA · CarIQ · CerebrumX

and more…

Prognostics · EV Range Application · In-Car Payments · Occupant Identity

COMPREDICT · ELECTRA · BlackBerry IVY · CarIQ · <OEM Logo>

Build and deploy your own custom insights

Use IVY as a platform for engaging partners

Leverage the growing list of IVY ecosystem partners

OEM VEHICLE
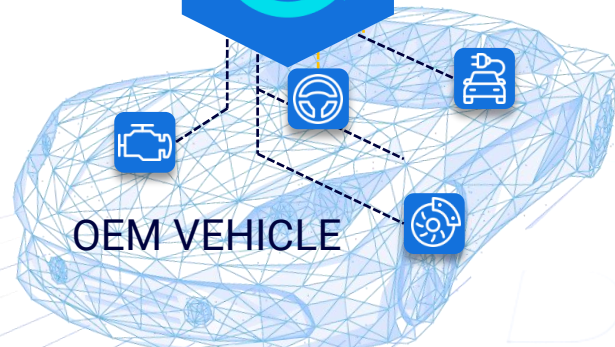
IVY DEMO

WHY COVESA AND VSS? - OPPORTUNITIES

Why COVESA and VSS?

Adoption leapfrogs development of own

Industry interest and adoption by OEMs makes it effectively a standard

Normalized signal data makes higher level applications more portable

Opportunities for development

COVESA VSS still in early stages of development

Right time to influence implementation and industry adoption

Access control (permission definition on a per-node basis)

BLACKBERRY ENHANCEMENTS TO VSS/SIGNAL CATALOG

Per-signal permission assignment →
- Each node has a single read permission
- Actuators have a single write permission
- Permissions are inherited from branch, simplifying permission assignment

Signal semantic uniqueness →
- Across signal catalog provenance (COVESA, BlackBerry, OEM, Tier 1, etc.)
- Across signal catalog versions
- Allows differentiation of signal semantics that have changed across provenance and version
- Allows application and software sensor portability (optionality support)

Additional signals →
- Support for signals requested by customers

Deletion of signals →
- Allows for in-vehicle signals in catalog to match exactly signals generated by vehicle

BLACKBERRY ENHANCEMENTS TO VSS/SIGNAL CATALOG

Signal semantics can change across signal catalog versions and provenances

→

- Some OEMs may prefer to use different units (mph instead of km/h, for example)
- Signal data type may evolve over time and thus versions (change from integer to float, for example)
- Affects application portability

Introspection of each signal's semantic is a burden on applications

→

- We expect most signal definitions will not change over time
- But must still be prepared to handle change
- Main issue is detection of change or differences

Node path name (and thus UUID) is insufficient to determine signal semantics

→

- Other elements, such as type, data type, unit, min, max, allowed values are semantically relevant

EUUID is a 64 bit hash of the node path name and semantical elements of each node
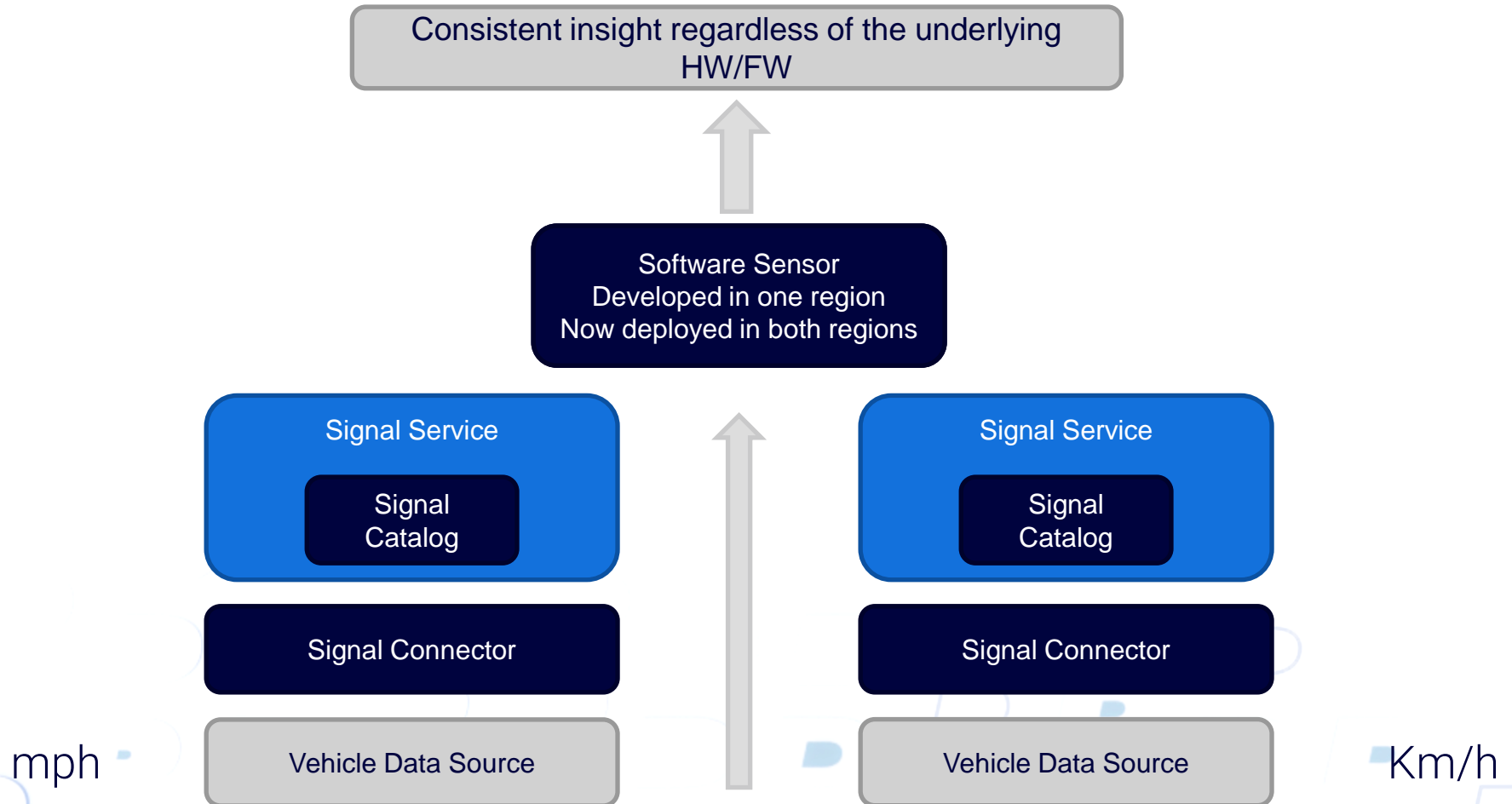
→

- Easier to manipulate than 128-bit UUID
- Can be used as an index in tables
- Guarantees uniqueness (within hashing limitations)

PORTABLE INTELLIGENT INSIGHTS

Two discrete problems:
- Developer coding for two possible inputs
- OEM deploying software sensor in two different regions

mph

Km/h

PERMISSION SUPPORT AND EUUID GENERATION

## Permissions File

```
"x-read-permission": [
    {
        "permission": "Vehicle.READ",
        "nodes": [
            "Vehicle"
        ]
    },
    {
        "permission": "Vehicle.PII.READ",
        "nodes": [
            "Vehicle.VehicleIdentification",
            "Vehicle.Driver.Identifier"
        ]
    }
]
```

Signal Catalog File

```
"Speed": {
    "datatype": "float",
    "description": "Vehicle speed.",
    "type": "sensor",
    "unit": "km/h",
    "uuid": "efe50798638d55fab18ab7d43cc490e9",
    "x-euuid": "86e92e0ee67d30dd",
    "x-read-permission": "Vehicle.READ"
},
```
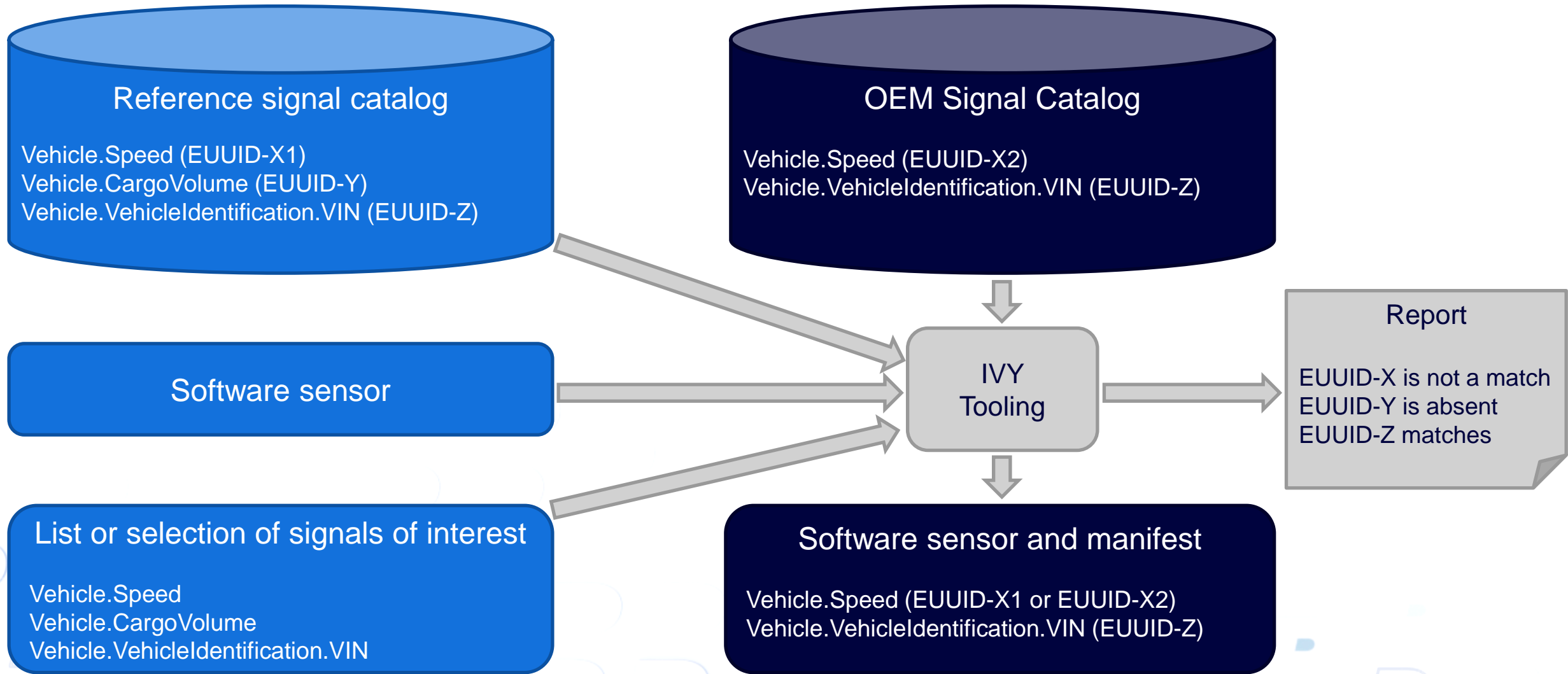
EUUID GENERATION AND PERMISSION ASSIGNMENT
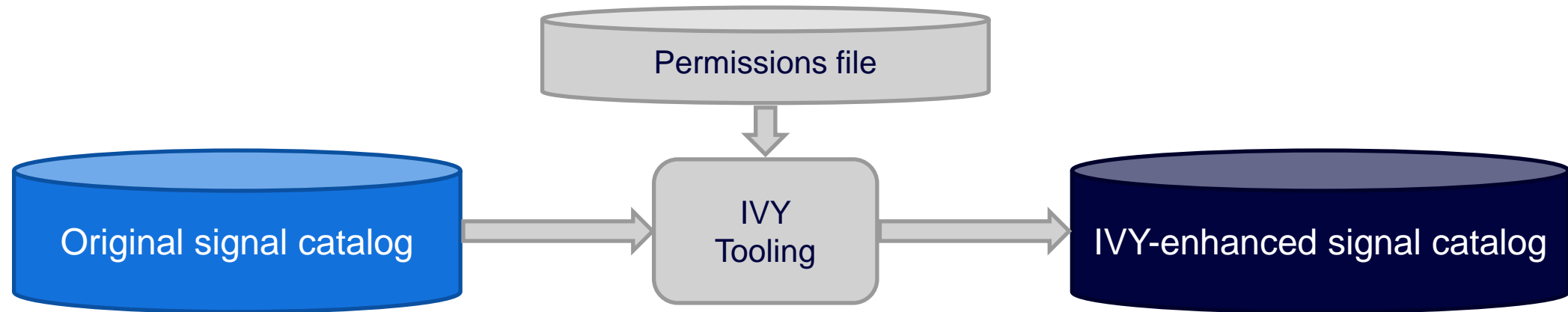

```
"Speed": {
    "datatype": "float",
    "description": "Vehicle speed.",
    "type": "sensor",
    "unit": "km/h",
    "uuid": "efe50798638d55fab18ab7d43cc490e9"
},
```

```
"Speed": {
    "datatype": "float",
    "description": "Vehicle speed.",
    "type": "sensor",
    "unit": "km/h",
    "uuid": "efe50798638d55fab18ab7d43cc490e9",
    "x-euuid": "86e92e0ee67d30dd",
    "x-read-permission": "Vehicle.READ"
},
```

DETECT SOFTWARE SENSOR DEPLOYMENT LIMITATIONS

Reference signal catalog

Vehicle.Speed (EUUID-X1)
Vehicle.CargoVolume (EUUID-Y)
Vehicle.VehicleIdentification.VIN (EUUID-Z)

OEM Signal Catalog

Vehicle.Speed (EUUID-X2)
Vehicle.VehicleIdentification.VIN (EUUID-Z)

Software sensor

List or selection of signals of interest

Vehicle.Speed
Vehicle.CargoVolume
Vehicle.VehicleIdentification.VIN

IVY Tooling

Report

EUUID-X is not a match
EUUID-Y is absent
EUUID-Z matches

Software sensor and manifest

Vehicle.Speed (EUUID-X1 or EUUID-X2)
Vehicle.VehicleIdentification.VIN (EUUID-Z)

EUUIDS IN A MANIFEST

```
{

    "EUUIDs": {
        "allOf": [
            {

                "oneOf": [
                        { "Vehicle.Speed", "EUUID-X1" },
                        { "Vehicle.Speed", "EUUID-X2" }
                ]
            },
            { "Vehicle.VehicleIdentification.VIN", "EUUID-Z" }
        ],
        "anyOf": [
            { "Vehicle.CargoVolume", "EUUID-Y" }
        ]
    }
}
```

PORTABLE SOFTWARE SENSOR CODE EXAMPLE

```
float vehicleSpeed = getSignal("Vehicle.Speed");

auto datatype = getDataType("Vehicle.Speed");

if (datatype == "mph")
{
    vehicleSpeed = vehicleSpeed * 1.60934;
}
```

Write this conversion only when needed, not across the whole signal set

DISCUSSION