Blueprint for Vehicle Data Oriented Strategy – Return of Experience

NEUTRAL SERVER

Kevin Valdek, CTO HIGH MOBILITY

Presented at GENIVI AMM on 16.05.2019

CONTENTS

CONTENTS

ACEA | Manufacturers Association

ACEA Position Paper

Access to vehicle data for third-party services

Extended Vehicle & Neutral Server recap

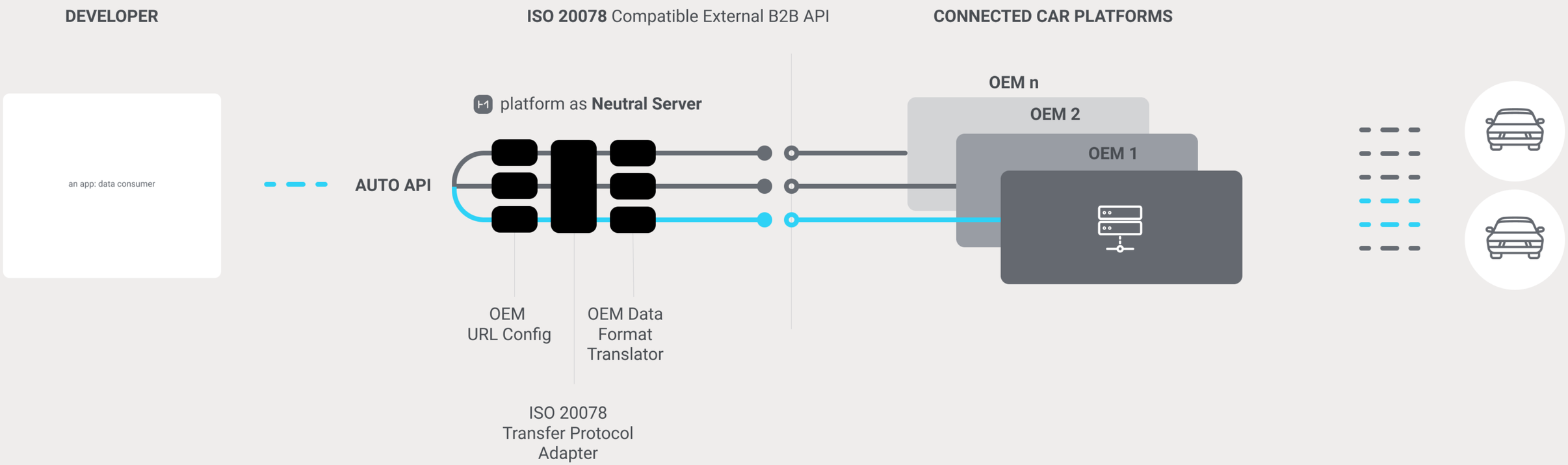
EXTENDED VEHICLE

- Sharing of vehicle telematics data with 3rd parties
- Both anonymous and personalised vehicle data
- Customer consent and customer choice in focus
- Read-only data

NEUTRAL SERVER

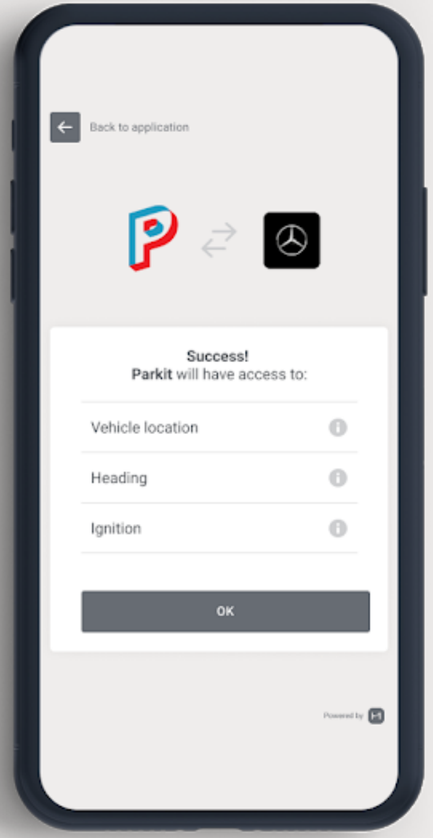
- Independent intermediary engaged by OEMs
- Allowed to broker data to 3rd parties within allowed scope
- Protects direct visibility of 3rd party business models from OEMs

Neutral Server - a cross-OEM trusted entity



Customer consent

- Customer consent a prerequisite
- 3rd party has to provide sufficient value to the customer
- Neutral Server can provide compatibility between OEM tech differences
- Good practices in use



Benefits for all sides

OEM BENEFITS

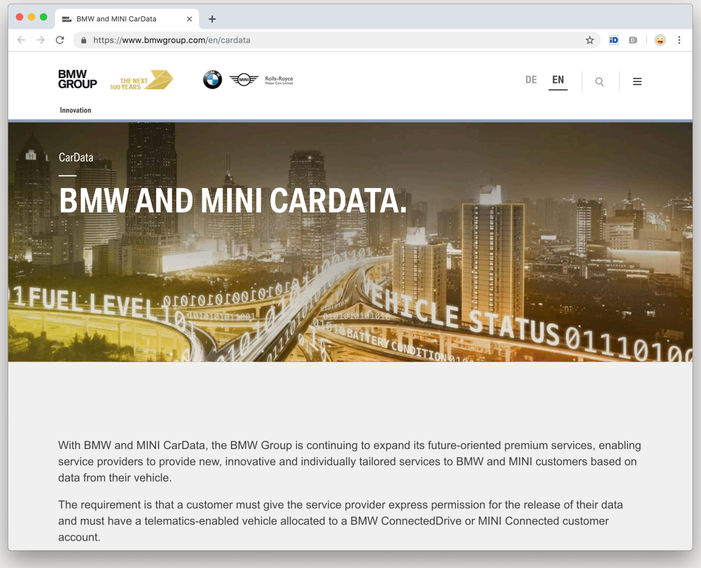
- Neutral Servers multiply the business potential
- Minimised effort from OEMs
- Support team and verification responsibilities shifted
- Possible to cater to new data consumer segments

3RD PARTY BENEFITS

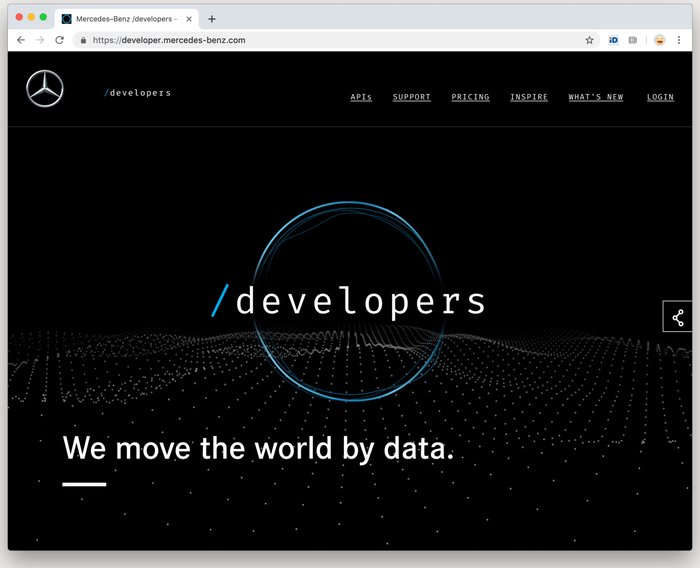
- Lowers the barrier for new mobility services
- Benefiting from added data partnerships
- *If done right*: one data contract, one data integration

CONTENTS

Current vehicle data availability



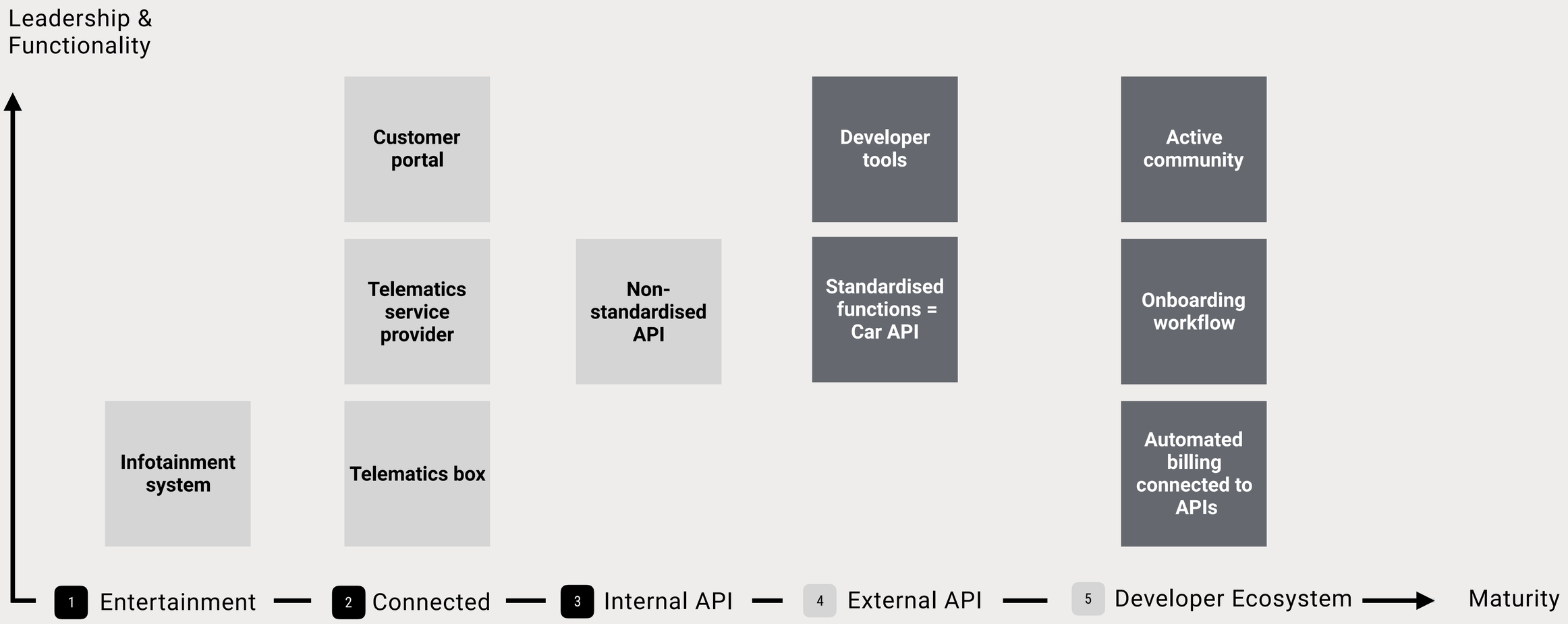
BMW CarData
since May 2017



Mercedes-Benz
since Dec 2018

Availability ~

The journey to a developer ecosystem through APIs



Reception from 3rd parties

3RD PARTIES LOVE

- Finally a scalable model
- No need for to administer hardware or dongles
- Choice in selecting which Neutral Server to integrate with
- Possible to integrate once and continuously support new car brands
- Can avoid individual negotiations with each OEM

3RD PARTY CHALLENGES

- Trade-off when it comes to data update rates
- Pricing and request limits vary significantly
- SLAs of high importance to invest in the effort

What’s possible today: data bundle examples

PAYD Insurance

Starting from €0,39 /mo

Fueling

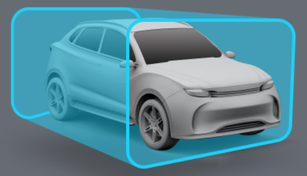
Starting from €0,29 /mo

Charging

Starting from €0,29 /mo

Damage prevention

Starting from €0,29 /mo

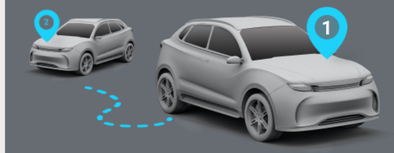


Theft prevention

Starting from €0,29 /mo

Parking

Starting from €1,56 /mo



Fleet management

Starting from €2,34 /mo

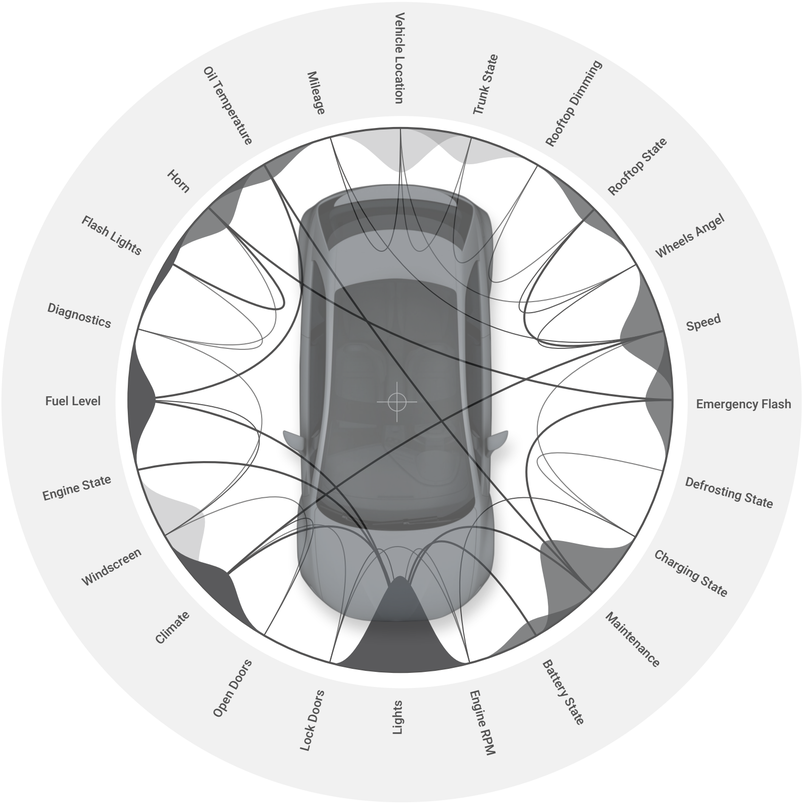
Maintenance

Starting from €0,78 /mo

CONTENTS

What attracts 3rd parties: comprehensive developer tools

1 Standardised API



2 Tooling

- SDKs
- Datasets
- Testing env

3 Resources

- SDKs
- Documentation
- Support

4 Community

- Analytics
- Chat support
- Forums

5 Marketplace (app store)

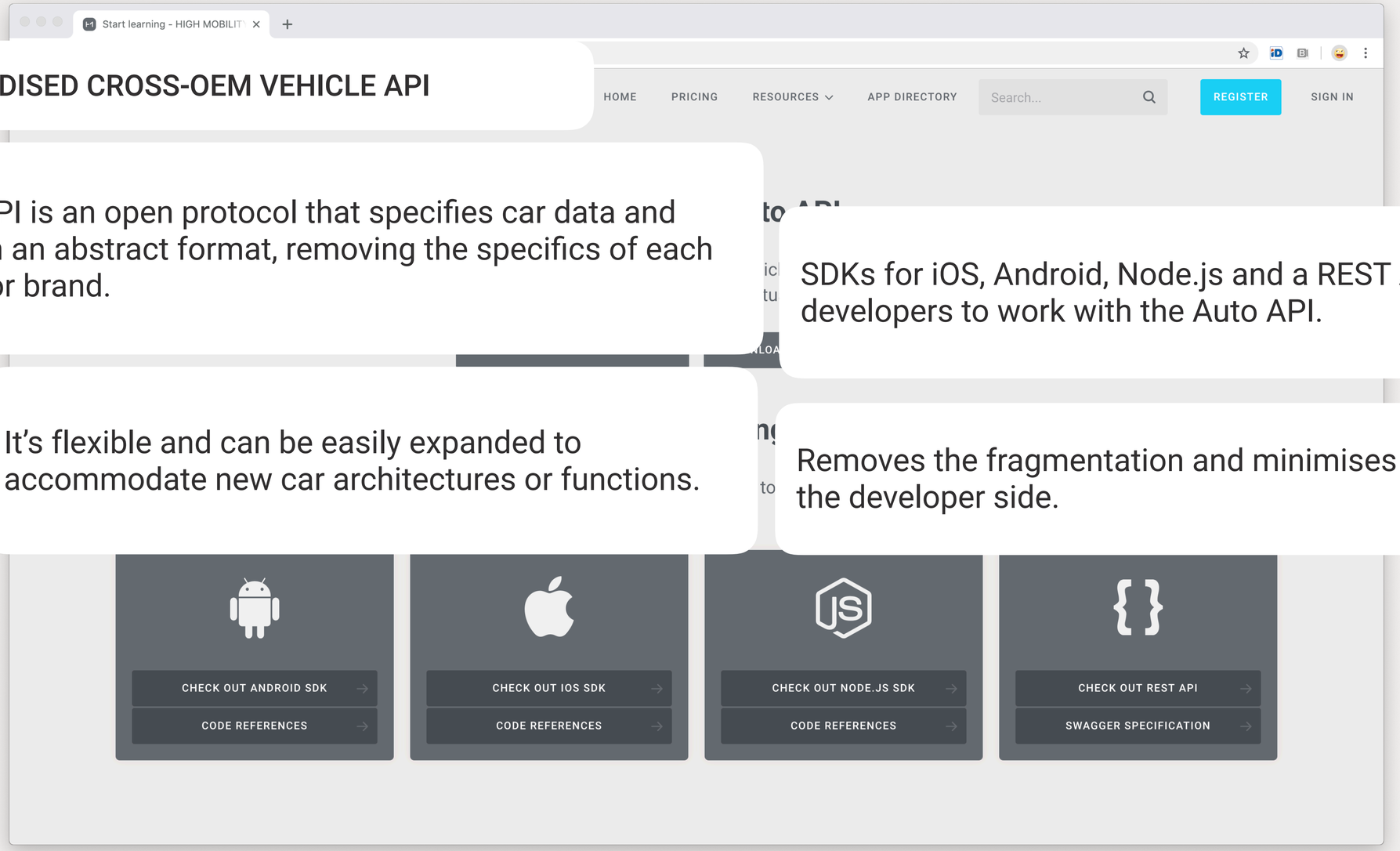
Verification

Publishing

Real car data

CONTENTS

Our approach: standardisation of a vehicle API



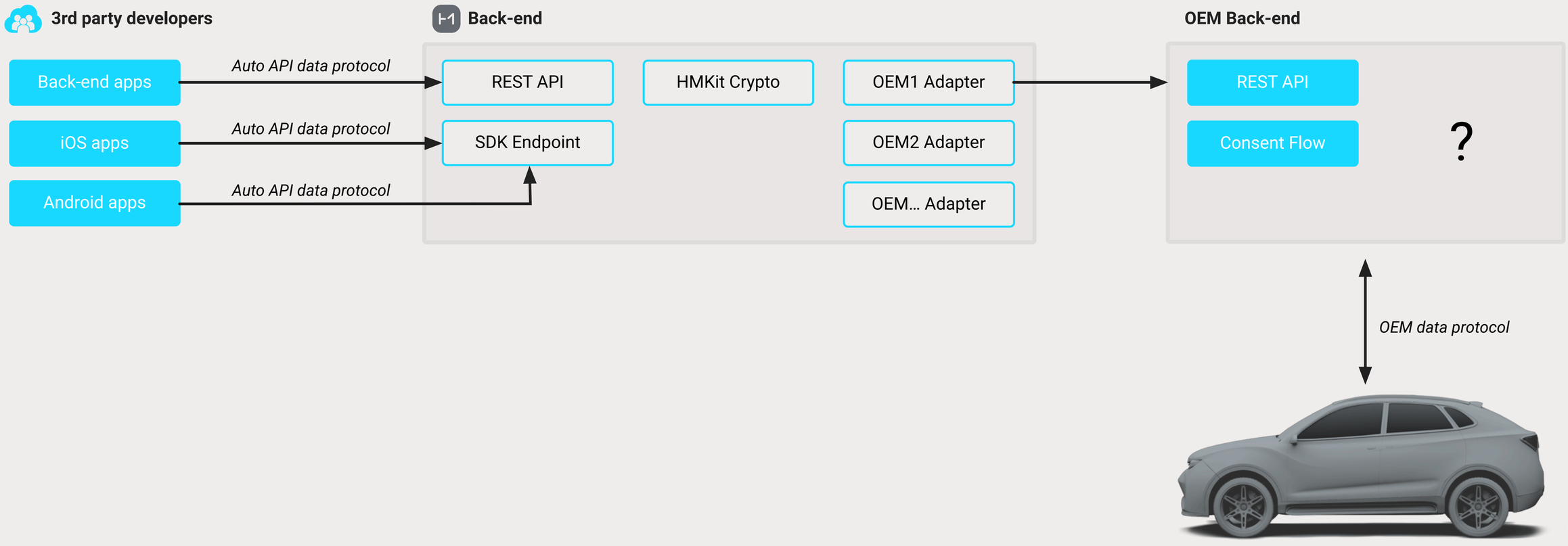
A STANDARDISED CROSS-OEM VEHICLE API

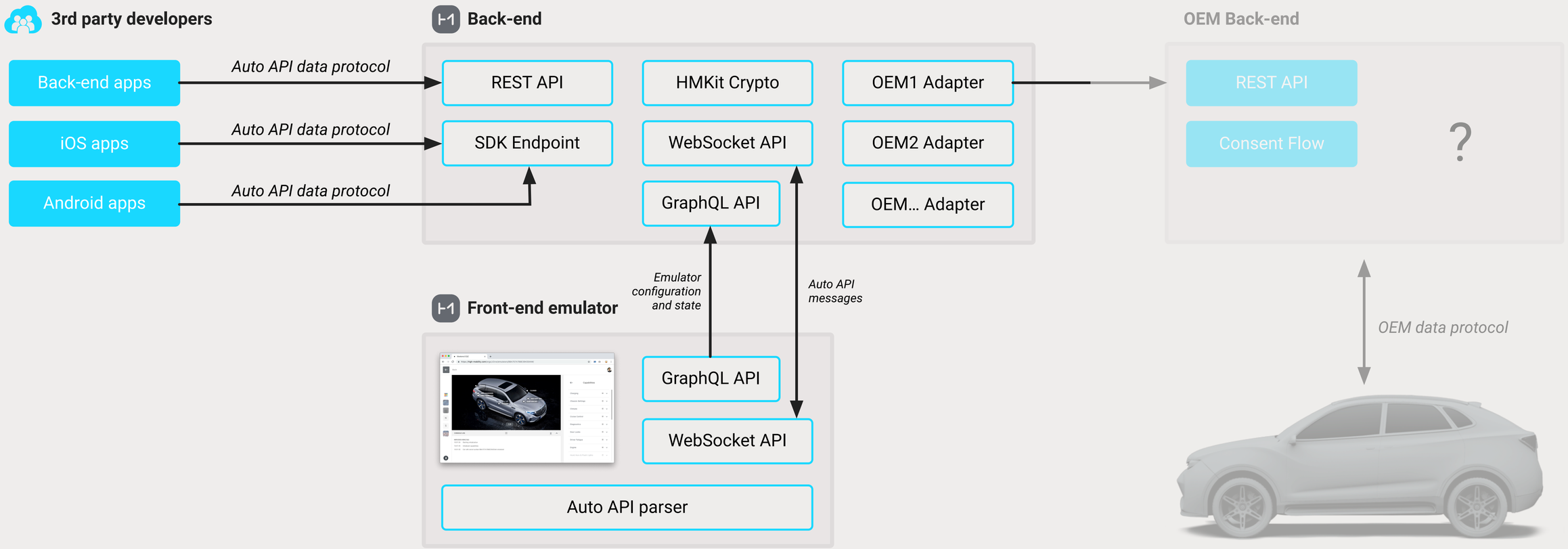
The Auto API is an open protocol that specifies car data and functions in an abstract format, removing the specifics of each car model or brand.

SDKs for iOS, Android, Node.js and a REST API to developers to work with the Auto API.

It's flexible and can be easily expanded to accommodate new car architectures or functions.

Removes the fragmentation and minimises effort on the developer side.





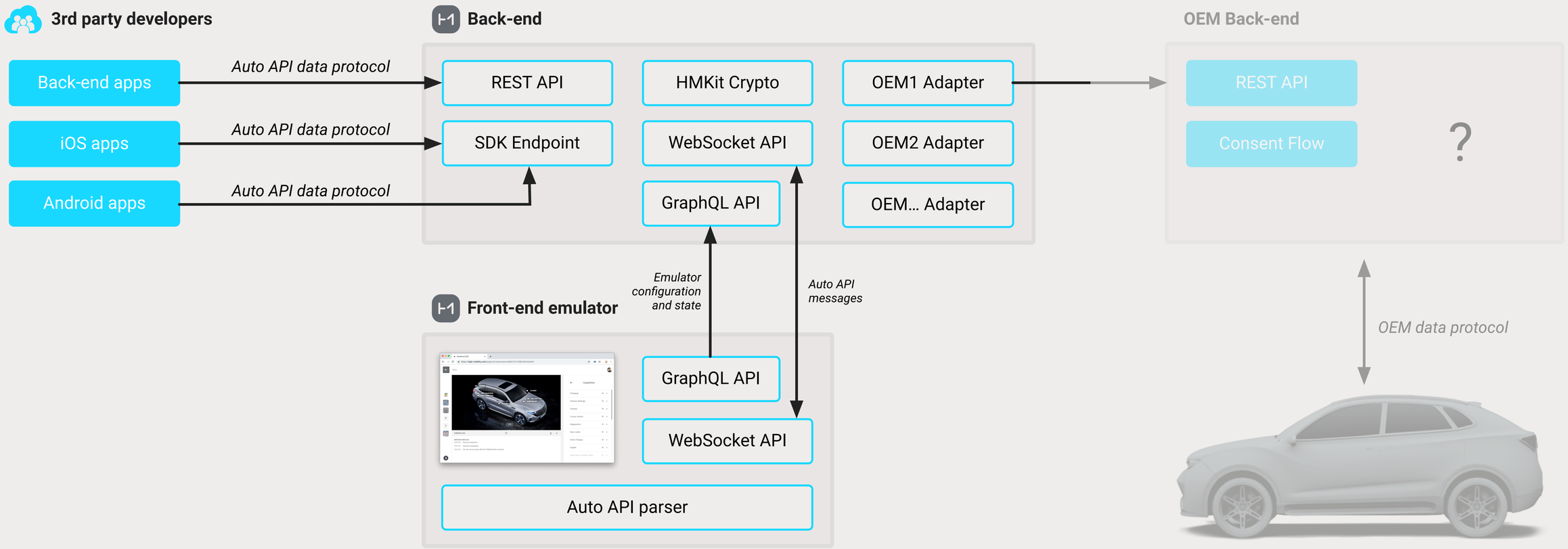
Approach taken to harmonise vehicle data for 3rd parties: Auto API

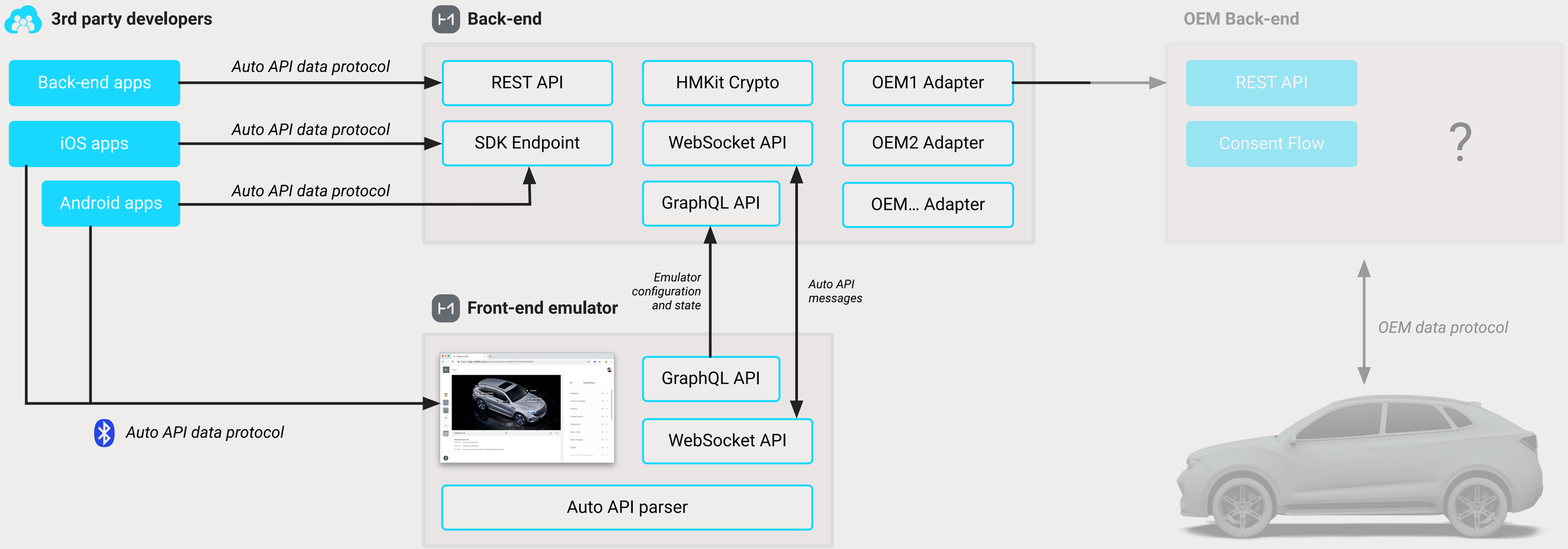
REQUIREMENTS

- Has to work in an IOT environment, embedded solutions
- Supports both polling and pushing of data
- Can efficiently be encrypted/decrypted
- Is specific for automotive use cases

SOLUTION

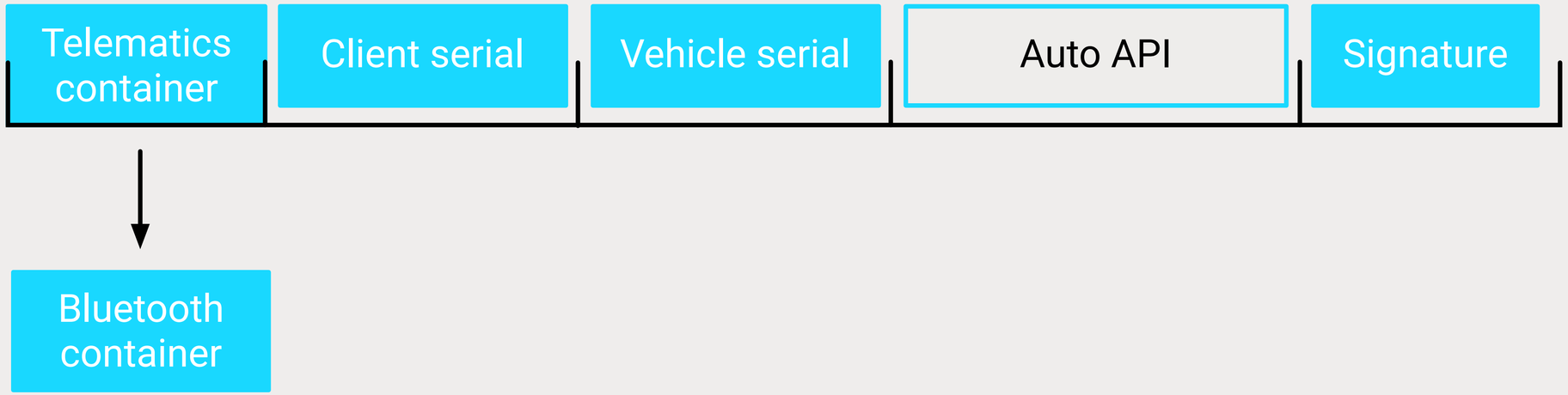
- Went ahead with two different components: HMKit, Auto API
- Binary format at its core — with abstraction layers
- SDKs to handle the security layer and platform specifics
- First prototypes with Bluetooth Low Energy, Telematics added later



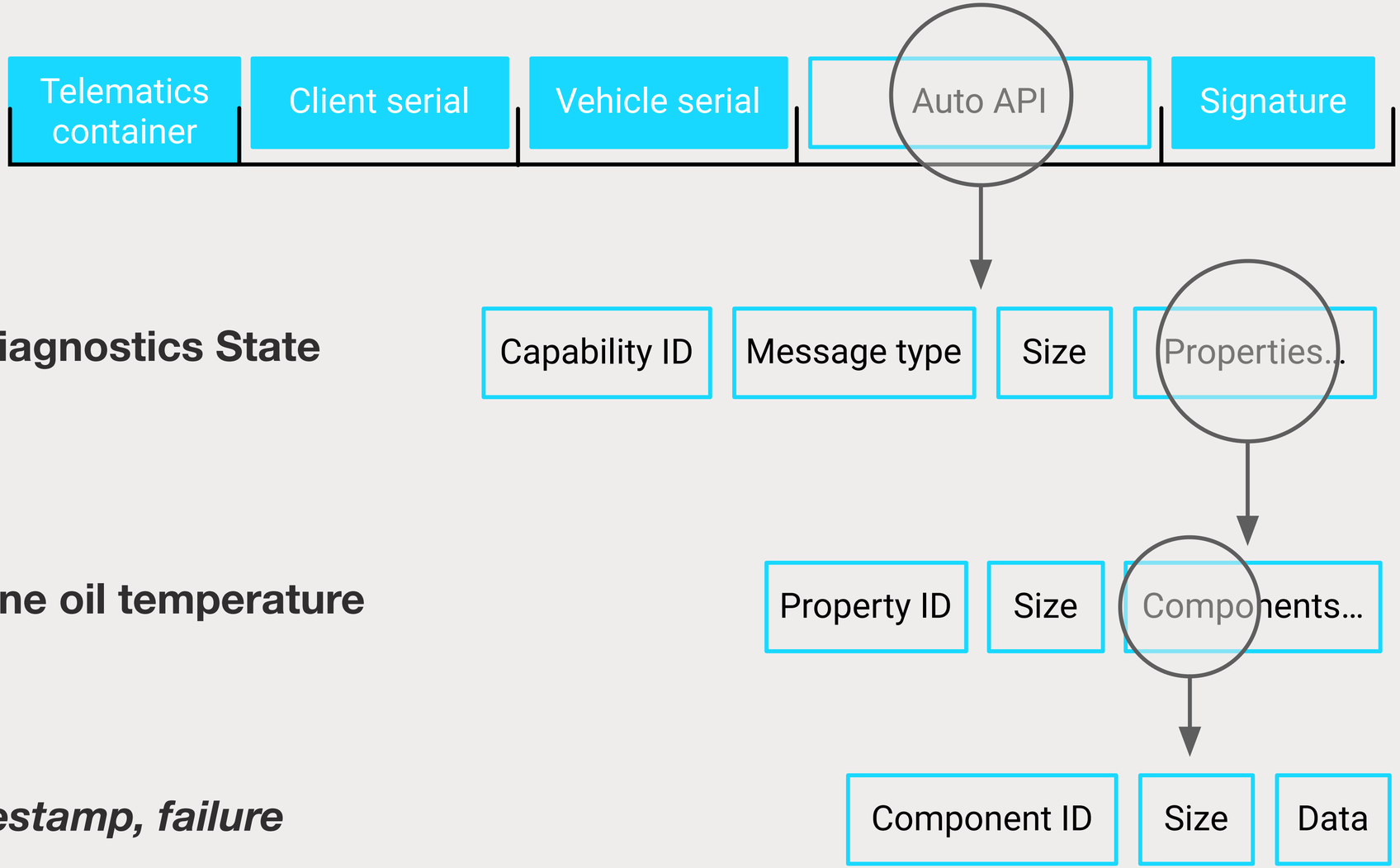


HMKit & Auto API

- HMKit — Transport protocol with security layer and SDKs
- Auto API — Vehicle data specification



HMKit & Auto API



Example: Diagnostics, Diagnostics State

Examples: Mileage, Engine oil temperature

Components: *value, timestamp, failure*

HMKit & Auto API

← Diagnostics State

This message is sent when a Get Diagnostics State message is received by the car. The new state is included in the message payload and may be the result of user, device or car triggered action.

MESSAGE SPECIFICATION

Message Direction	Sent from Car to Smart Device
Data[0..1]	**Message Identifier**
0x00	MSB
0x33	LSB
Data[2]	**Message Type**
0x01	Diagnostics State
Property Data[0]	**Property Identifier**
0x01	Identifier for Mileage

MESSAGE EXAMPLE

```
[
  0x00, 0x33, # MSB, LSB Message Identifier for Diagnostics
  0x01,       # Message Type for Diagnostics State

  0x01, # Property identifier for Mileage
  (uint 16) 7, # Property size is 7 bytes
  0x01, # Data component identifier
  (uint 16) 4, # Data component size is 4 bytes
  0x00, 0x02, 0x49, 0xF0 # Odometer is 150'000 km

  0x02, # Property identifier for Engine oil temperature
  (uint 16) 5, # Property size is 5 bytes
  0x01, # Data component identifier
  (uint 16) 2, # Data component size is 2 bytes
  0x00, 0x63 # Engine oil temperature is 99 degrees Celsius
```

hm-auto-api-swift — -fish

```
[~/D/W/S/hm-auto-api-swift (master|✔) $ ./AutoAPICLT 0011010100114a46325348424443374348343531383639020001010300065479706520580400064d7920]
 436172050006414243313233060008506163 6B6167652B07000207E108000C4573746f72696c20426c617509000200DC0A0001050B00010599000B002101010001000200
 0101990007002701010001 02

 VehicleStatus =
         colourName = Estoril Blau
         licensePlate = ABC123
         modelName = Type X
         modelYear = 2017
         name = My Car
         numberOfDoors = 5
         numberOfSeats = 5
         powerKW = 220
         powerTrain = allElectric
         salesDesignation = Package+
         states =
                 TrunkAccess =
                         lock = unlocked
                         position = opened
                         * properties.count = 2
                 RemoteControl =
                         angle = nil
                         controlMode = started
                         * properties.count = 1
         vin = JF2SHBDC7CH451869
         * properties.count = 13

[~/D/W/S/hm-auto-api-swift (master|✔) $ ./AutoAPICLT                                                                                                ]

   Enter HEX data after the command to parse it.

         Allowed besides HEX:
          - 0x
          - commas
          - spaces
          - new line (i.e. when inside "...", or a var)

         Flags:
          -b64: input is in Base64
          -dc:  input is like in Developer Center (0x00, 0x01 # Comment)
          -ep:  expand properties

   Example: 10938A1 12B9C9 1239 0x1b, 0xc0ca

~/D/W/S/hm-auto-api-swift (master|✔) $
```

Auto API journey from Level 1 to Level 10

Emulator MVP

Neutral Server Live

2014 | 2015 | 2016 | 2017 | 2018 | 2019

Building the SDKs through UX concepts

Auto API Level 1
- Door Locks
- HMKit iOS

Auto API Level 2-4
- Diagnostics
- Trunk Access
- Climate
- Charging
- HMKit Android

Platform open to developers

Auto API Level 5
- Capabilities
- Vehicle Status
- Vehicle Location
- Rooftop
- Windows
- Maintenance
- Engine
- Theft Alarm
- *+ more*
- HMKit Node.js

Auto API Level 6-7
- Race
- Offroad
- Dashboard Lghts
- Chassis Settings
- Seats
- Light Conditions
- *+ more*
- REST API

Car data & App Directory added

Auto API Level 8-10
- Historical
- MultiCommand
- Usage
- *+ more*

 www.high-mobility.com

HMKit & Auto API landscape

Telematics Bluetooth Low Energy

Cloud

Node.js: HMKit

uses

REST API

uses

Mobile

Android: HMKit

uses

iOS: HMKit

uses

Internet of Things

SOON

Python: HMKit

uses

Parse the received command's bytes

```
byte[] bytes = ...
Command command = CommandResolver.resolve(bytes);

VehicleStatus vehicleStatus;
Capabilities capabilities;

if (command instanceof VehicleStatus) {
    vehicleStatus = (VehicleStatus) command;
}
else if (command instanceof Capabilities) {
    capabilities = (Capabilities) command;
}
```

Get a specific state from the vehicle status

```
LockState state = vehicleStatus.getState(LockState.TYPE);
if (state != null) {
    ...
}
```

Inspect whether the capability is supported for the vehicle

```
if (capabilities.isSupported(LockState.TYPE)) {
    ...
}
```

highmobility / **hm-java-auto-api**

Code | Issues 0 | Pull requests 0 | Projects 0 | Insights

☕ Auto API parsing library for Java https://developers.high-mobility.com/

hmkit automotive

highmobility / **hm-auto-api-swift**

Code | Issues 0 | Pull requests 0 | Projects 0 | Insights

AutoAPI.framework - the car API parsing library for HMKit

hmkit automotive

highmobility / **hm-auto-api-elixir**

Code | Issues 0 | Pull requests 0 | Projects 0 | Insights

🚘 Auto API parsing library for Elixir https://high-mobility.com/#/learn/doc...

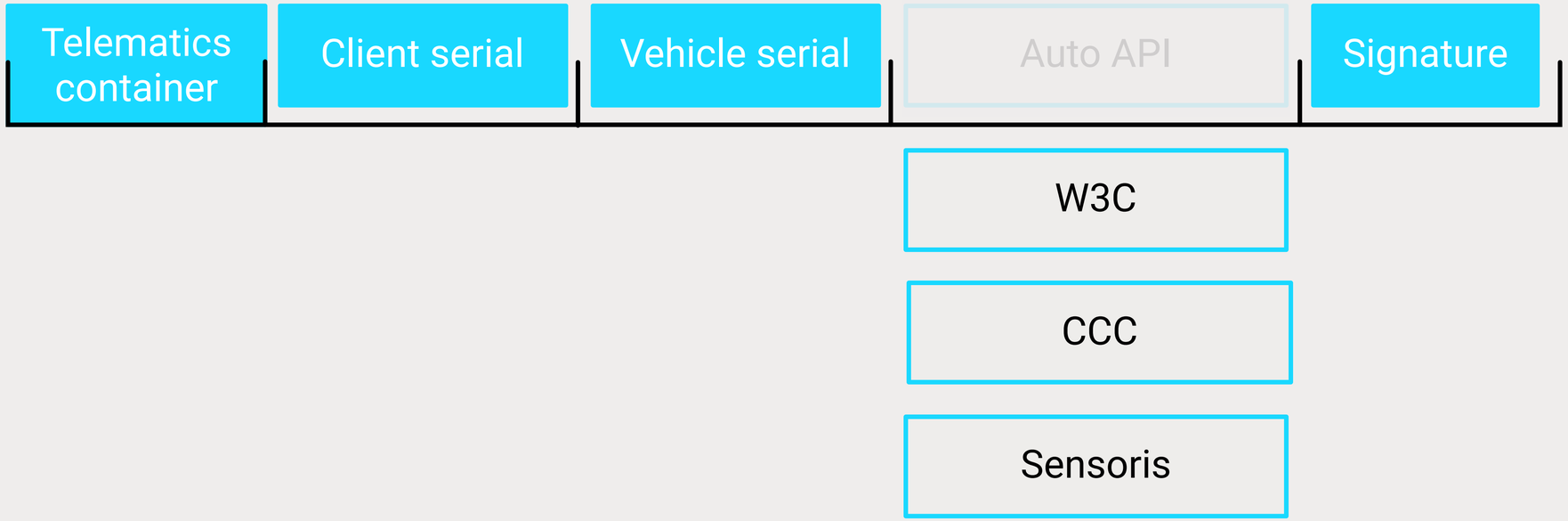
automotive api binary parser hmkit

… and more

CONTENTS

HMKit & vehicle data standardisations

- Positive trend for vehicle data format specifications
- Most helpful if the impact includes external APIs
- Allows for better adoption among 3rd parties



Neutral Server

Thank you

High-Mobility GmbH

Skalitzer Str. 68
10997 Berlin, GERMANY