Grant Agreement Number: 731993

Project acronym: AUTOPILOT

Project full title: Automated driving Progressed by Internet of Things

D2.7

Report on the Implementation of the IoT Platform

Due delivery date: 31 December 2019

Actual delivery date: 18 November 2019

Organization name of lead participant for this deliverable: IBM Research Ireland

Dissemination level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 731993.

AUTOPILOT

Document Control Sheet

Deliverable number:	D2.7
Deliverable responsible:	Anton Dekusar, IBM Research, Ireland
Workpackage:	WP2
Editor:	Anton Dekusar, IBM Research, Ireland

Author(s) – in alphabetical order		
Name	**Organisation**	**E-mail**
Alexander Velizhev	IBM RE	ave@zurich.ibm.com
Benjamin Hebgen	NEC	benjamin.heben@neclab.eu
Bram van den Ende	TNO	bram.vandenende@tno.nl
Carlos Rosales	CTAG	carlos.rosales@ctag.com
Daan Ravesteijn	TNO	daan.ravesteijn@tno.nl
Daniele Brevi	LINKS Foundation	Daniele.brevi@linksfoudation.com
Ilaria Bosi	LINKS Foundation	Ilaria.bosi@linksfoudation.com
David Martin	Gemalto	Martin.David@gemalto.com
Fabrizio Gatti	Telecom Italia	fabrizio1.gatti@telecomitalia.it
Frans van Dingenen	Technolution	frans.van.dingenen@technolution.nl
Georgios Karagiannis	Huawei	georgios.karagiannis@huawei.com
Gurkan Solmaz	NEC	gurkan.solmaz@neclab.eu
Frans van Dingenen	Technolution	frans.van.dingenen@technolution.nl
Jean-Francois Simeon	Continental	Jean-Francois.Simeon@continental-corporation.com
Johan Scholliers	VTT	Johan.Scholliers@vtt.fi
Lorenzo Viola	Huawei	lorenzo.viola.ext@huawei.com
Louis Touko Tcheumadjeu	DLR	Louis.ToukoTcheumadjeu@dlr.de
Mahdi ben Alaya	Sensinov	benalaya@sensinov.com
Mariano Falcitelli	CNIT	Mariano.Falcitelli@cnit.it
Martin Bauer	NEC	martin.bauer@neclab.eu
Pablo García	CTAG	pablo.garcia@ctag.com
Paolo Scalambro	Telecom Italia	paolo.scalambro@telecomitalia.it
Roberto Gavazzi	Telecom Italia	roberto.gavazzi@telecomitalia.it
Silvia Alén González	CTAG	silvia.alen@ctag.com
Xurxo Legaspi	CTAG	xurxo.legaspi@ctag.com

Document Revision History			
Version	**Date**	**Modifications Introduced**	
		Modification Reason	**Modified by**
V0.1	04/09/2019	Initial draft based on D2.3	All authors
V1.0	07/11/2019	First version of D2.7	All authors
V1.1	18/11/2019	Team review	NEC, DLR
V1.2	16/12/2019	Peer review edits	IBM IE

Abstract
This document represents D2.7, "Report on the implementation of the IoT platform". D2.7 describes the final status of the implementation of the open IoT platform for autonomous driving of the AUTOPILOT project. The report covers all the project pilot sites and use cases. It describes the IoT architecture implemented as part of the AUTOPILOT project.

Legal Disclaimer

Abbreviations and Acronyms

Acronym	Definition
6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
ACP	Access Control Policy
AD	Autonomous Driving
ADN	Application Dedicated Node
AE	Application Entity
API	Application Programming Interface API
ASN	Application Service Node
ASN-CSE	Application Service Node – Common Services Entity
AVP	Automated Valet Parking
BLE	Bluetooth Low Energy
CAM	Cooperative Awareness Message
C-ITS	Cooperative Intelligent Transport Systems
CAN	Controller Area Network
CeH	Connected eHorizon
CEMA	Crowd Estimation and Mobility Analytics
CoAP	Constrained Application Protocol
CSE	Common Services Entity
CSEBase	Common Services Entity Base
DENM	Decentralized Environmental Notification Message
DSRC	Dedicated Short-Range Communications
EC	European Commission
ETSI	European Telecommunications Standards Institute
FMS	Fleet Management System
GE	Generic Enabler
HMI	Human-Machine Interface
IEEE	Institute of Electrical and Electronics Engineers
IN-CSE	Infrastructure Node – Common Services Entity
IoT	Internet of Things
JSON	JavaScript Object Notation
ITS	Intelligent Transport Systems
LAN	Local Area Network
LTE	Long-Term Evolution
LWM2M	Lightweight Machine to Machine
M2M	Machine to Machine
MAP	Map Data
MAV	Micro air vehicle
Mca	Reference Point for M2M Communication with AE
Mcc	Reference Point for M2M Communication with CSE
Mcc’	Reference Point for M2M Communication with CSE of different M2M Service Provider
MEC	Multi-Access Edge Computing

MN-CSE	Infrastructure Node – Common Services Entity
MQTT	OASIS Message Queuing Telemetry Transport
NB-IoT	Narrowband IoT
NGSI	Next Generation Services Interface
OBU	On-Board Unit
OEM	Original Equipment Manufacturer
OSGi	Open Service Gateway Initiative
PaaS	Platform as a Service
PoA	Point of Access
POI	Point of Interest
PS	Pilot Site
RDF	Resource Description Framework
RemoteCSE	Remote Common Service Entity
OWL	Web Ontology Language
REST	Representational State Transfer
RSU	Road Side Unit
SP	Service Provider
SPAT	Signal Phase and Time
SSL	Secure Sockets Layer
TCC	Traffic Control Centre
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TMC	Traffic Management Centre
UD	Urban Driving
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
UWB	Ultra-Wide Band
VDH	Vehicle Data Hub
WP	Work Package
VRU	Vulnerable Road User
Wi-Fi	Wireless Fidelity
WSN	Wireless Sensor Network
XML	Extensible Markup Language

Table of Contents

List of Figures

List of Tables

Executive Summary

This deliverable, D2.7, reports on the final IoT architecture developed as part of the AUTOPILOT task T2.3, "Development of the Open IoT Service Platform".

The AUTOPILOT IoT service platform is a federation of several IoT platforms (referred to as proprietary IoT platforms) provided by the project partners, all being interconnected through an open oneM2M standard IoT platform, referred to as the "oneM2M interoperability platform". The proprietary IoT platforms collect data from connected devices. They exchange IoT data and events with the interoperability platform through oneM2M interworking proxies.

Several IoT platforms have been deployed for the pilot sites:

- FIWARE IoT platform, used in the Dutch pilot site,
- Watson IoT Platform, used in the Dutch and Spanish pilot sites,
- HUAWEI OceanConnect IoT platform, used in the Dutch pilot site,
- Technolution MobiMaestro platform, used in the Dutch pilot site,
- TIM oneM2M IoT platform, used in the Italian pilot site,
- SENSINOV oneM2M platform, used in the French and Dutch pilot sites,
- openMTC oneM2M platform, used in the Finnish pilot site.
- MESIM IoT platform, used in the Korean pilot site.

oneM2M Interworking proxies allow non-oneM2M-compliant platforms to be connected to the oneM2M IoT platform.

Across the pilot sites, devices and vehicles are connected to the various IoT platforms. But their data may flow as required by the use cases from one platform to another allowing them to be shared across the whole IoT ecosystem. The vehicle AD functionality was extended to support IoT data for all the project use cases.

To further facilitate interoperability between the IoT platforms, a Data Modelling Activity Group (DMAG) worked on specifying a common IoT data model for the project use cases and pilot sites. The common IoT data model is based on existing standards: SENSORIS for vehicle messages and detection events, and DATEX II for parking and traffic information.

1 Introduction

1.1 Purpose of the document

This deliverable, D2.7, reports the work being undertaken as part of the AUTOPILOT task T2.3, "Development of the Open IoT Service Platform".

The document provides an overview of the AUTOPILOT open IoT architecture for autonomous driving, developed in six pilot sites: Finland, France, Italy, South Korea, the Netherlands, and Spain.

An overview of the deployed IoT platforms, their capabilities, and statuses of deployment, purposes, and access instructions are provided.

The document also provides an overview of the implementation of the IoT ecosystem in the pilot sites.

The chapters 5 and 6 of the document provide an overview of the work done related to interoperability and the common IoT data models elaborated in the project.

The last chapter includes the final evaluation of the work done in this task during the project.

1.2 Intended audience

This deliverable is a public report intended for both internal use by the AUTOPILOT partners and external use.

2 Overview of the IoT Architecture Implementation

This chapter provides an overview on the IoT architecture of AUTOPILOT, which has been specified as part of task 1.2 “IoT Architecture and Specification”.

2.1 Functional Overview

The AUTOPILOT IoT architecture builds on, and borrows building blocks from, relevant IoT architectures such as AIOTI[1] and IoT-ARM[2]. It aims to provide a global IoT service coverage through features such as **openness**, **flexibility**, **interoperability** between IoT platforms, leveraging of **standards** for communication and interfacing, and **federation** of in-vehicle, road-side unit, and pilot site IoT platforms.

[1] Alliance for Internet of Things Innovation (AIOTI): https://aioti.eu
[2] ARM-IoT: https://developer.arm.com/products/architecture/system-architecture

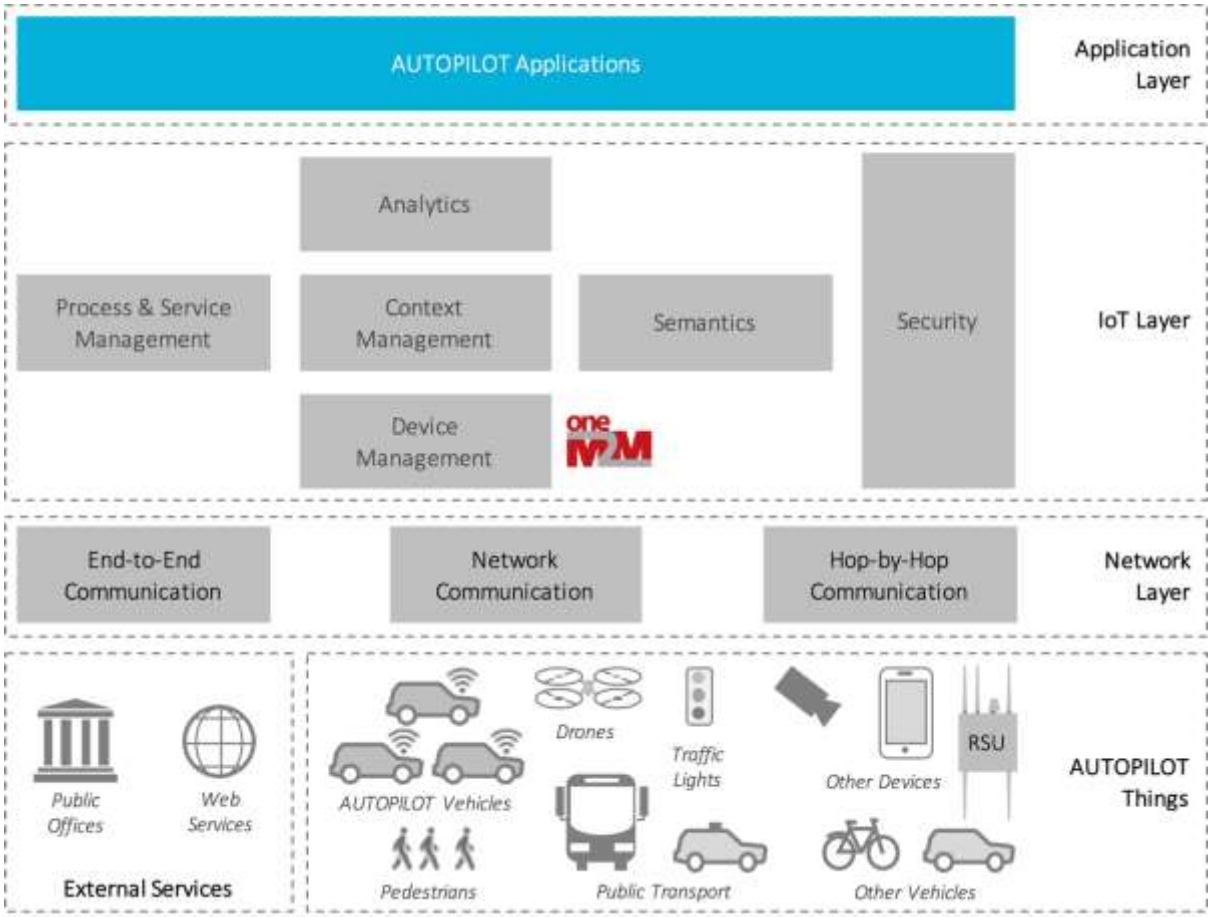

Figure 1. AUTOPILOT IoT Architecture: Functional View

As shown in Figure 1, the AUTOPILOT IoT architecture has four main functional layers:

- **Things Layer:** Includes the AUTOPILOT "things" (vehicles, cameras, drones, road side units, etc.) and external services provided by public offices or private web services.
- **Network Layer:** Enables communication throughout the IoT ecosystem
- **IoT Layer:** Enables the IoT functionality through a set of IoT building blocks: *device management, context management, process & service management, semantics, analytics,* and *security.* Each of these functional building blocks is specified in detail in the AUTOPILOT deliverable D1.3 [D1.3].
- **Application Layer:** Contains services that leverage the AUTOPILOT IoT. In AUTOPILOT, this includes services provided by the use cases to the AD vehicles and users (e.g., drivers, car/ride sharing customers, etc.).

2.2 Component View

Given that AUTOPILOT has several large-scale pilot sites, the architectural components of the open IoT platform (infrastructure, IoT devices, services, etc.) are inherently physically distributed. AD functions themselves have varying requirements in terms of speed of access (latency), availability, and range (covered area). Some localised mission critical functions, such as warning other vehicles in the immediate proximity that a pedestrian is jaywalking, need to be accessible within very low latency. Other functions, such as notification about a parking spot being made available, need to cover wider areas but are less demanding in terms of latency. As a result, the AUTOPILOT IoT platform was designed and implemented as a federation of IoT platforms.

Figure 2 illustrates the AUTOPILOT **federated** IoT platform architecture. The term "federated", in this

context, means that there are several layers of IoT platforms deployed on a variety of physical infrastructures starting from the in-vehicle IoT layer to the top-level internet cloud-based IoT platform.

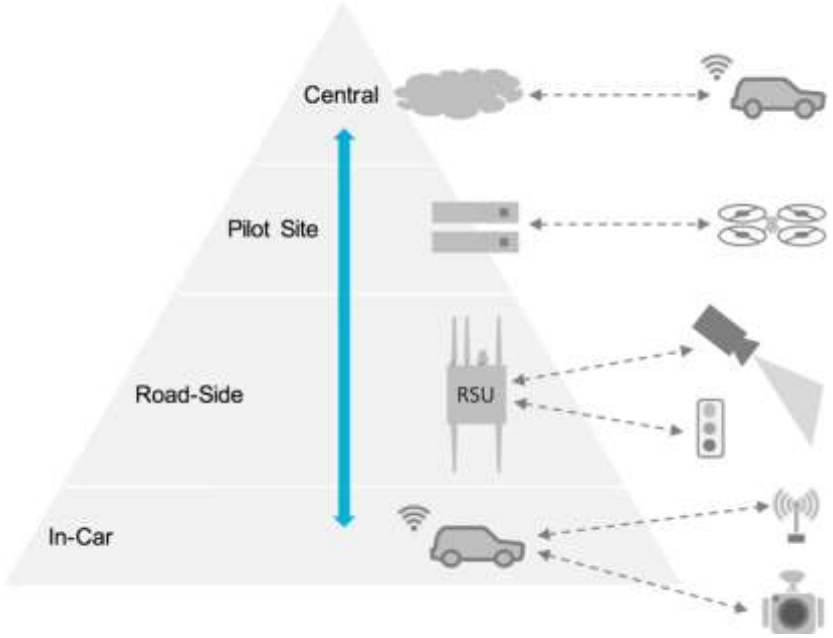

Figure 2. Layers of the Federated IoT Architecture

In this architecture data may flow from any level to any level as required by the use cases. At a given level, data may be processed to generate new information that may be published to an IoT platform at another layer. For example, image data submitted by a dash camera to the in-vehicle IoT platform, may be processed inside the vehicle, and upon detection of an object or hazard on the road a message may be generated and submitted to the road-side and/or cloud IoT platforms.

Figure 3 shows the AUTOPILOT target IoT architecture. As shown in this diagram, devices, gateways, and in-vehicle and road-side IoT platforms exchange information (e.g., about detected objects, hazards, vulnerable road users, traffic lights, vehicle updates, etc.) with several distributed cloud IoT platforms. We distinguish the following two types of cloud IoT platforms.

1. **Proprietary IoT Platforms:** These are used by some applications and use cases to exchange specific data with specific devices or vehicles. For example, the Brainport car/ride sharing service and automated valet parking service use Watson IoT Platform™ to collect data from their vehicles. Several proprietary IoT platforms are used in AUTOPILOT for various purposes, use cases and pilot sites. These are introduced in chapter 3.
2. **oneM2M Interoperability Platform:** This is the central IoT platform for exchanging IoT messages relevant to all autonomous driving (AD) vehicles.

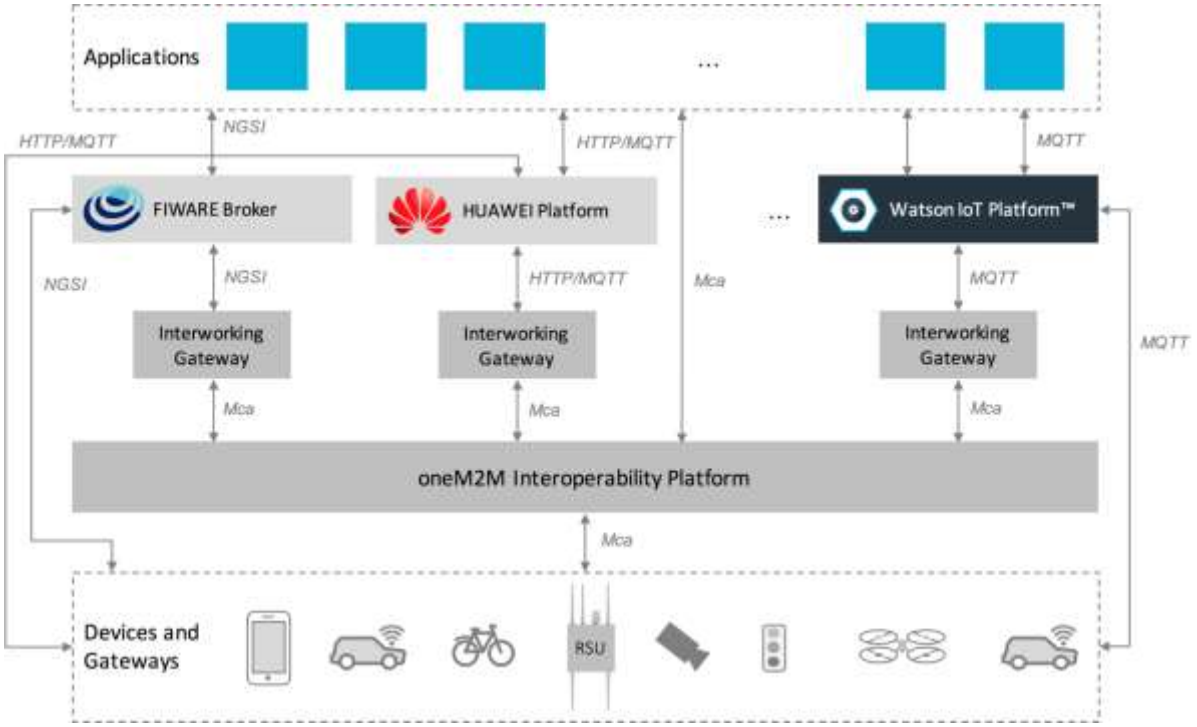

Figure 3. Autopilot Federated IoT Architecture

The proprietary IoT platforms are networked through the oneM2M interoperability platform and are connected to this through oneM2M interworking gateways. The interworking gateway of a given proprietary IoT platform may be configured to share selected data types with the interoperability platform. Such data then becomes accessible to all the connected IoT platforms through the oneM2M interoperability platform. This is particularly useful for sharing data relevant to all the AD vehicles and applications, such as detected hazards, vulnerable road users, objects, etc.

3 Implemented IoT Platforms

This chapter provides an overview of the deployed IoT platforms, their components, features, and intended use. Links to the deployed platform instances, their documentation, source code (if applicable), client-side code, and instructions on how to access the IoT platforms are also provided when applicable.

This chapter reports on the following types of activities:

- Deploying, customising, and configuring the project IoT platforms,
- Connecting devices to the IoT platforms and testing their connectivity,
- Developing and deploying the oneM2M interworking proxies to connect the proprietary pilot site IoT platforms to the central oneM2M interoperability platform.

Other work related to designing and implementing the common IoT exchange data models is reported in Chapter 6.

3.1 Watson IoT Platform™

This section provides an overview on the IBM Watson IoT Platform and its deployment in AUTOPILOT.

3.1.1 Overview of Watson IoT Platform™

This section provides an overview of the IBM Watson IoT Platform™ and its key concepts. Further

details are provided in the official documentation of the platform [IBM18a].

Watson IoT Platform™ (cf. Figure 4) is an IoT broker and device manager, which can be connected to a wide variety of devices, gateways, and applications. It allows devices and applications to publish and subscribe to data. The platform provides secure communication to and from any devices, using MQTT and TLS. It also provides a dashboard for monitoring the devices.

Device events published on Watson IoT platform™ may be pushed to a database for storage. By default, Watson IoT Platform™ supports the IBM Cloudant database [IBM18b].

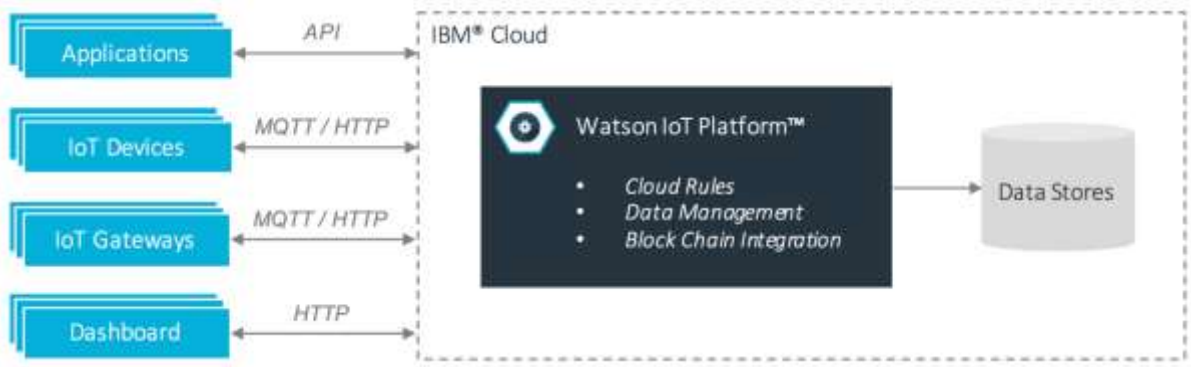

Figure 4. Architecture of the Watson IoT Platform™

The following subsections introduce the key concepts of Watson IoT Platform™.

3.1.1.1 Devices

A *device* can be anything that has a connection to the Internet and that can push data into the cloud. However, devices cannot communicate directly with other devices, instead devices accept commands from applications, and send events to applications. Devices in the Watson IoT Platform™ are identified by a unique authentication token. They **must be registered** before they can connect to the Watson IoT Platform™. The Watson IoT Platform™ recognises two classes of device; *managed devices* and *unmanaged devices*.

Managed devices are defined as devices that contain a device management agent. A device management agent is a set of logics that allows the device to interact with the Watson IoT Platform™ Device Management service by using the Device Management Protocol. Managed devices can perform device management operations, including location updates, firmware downloads and updates, reboots, and factory resets. Management operation may be triggered through the main Watson IoT Platform™ dashboard or the device management API. Device management can also be extended to include custom device management actions.

Unmanaged devices are all devices without a device management agent. They may connect to the Watson IoT Platform™ and send and receive events and commands, but they cannot send device management requests or perform device management operations.

3.1.1.2 Gateways

Gateways are specialised devices that have the combined capabilities of an application and a device, which allows them to serve as access points for other devices. Devices that cannot connect directly to the Internet can access the Watson IoT Platform™ service by first connecting to the gateway device. Gateways have all the functions of a device, but can also publish and subscribe on behalf of the devices connected to them.

Examples of gateways are:

- A software in a vehicle that collects data from the vehicle sensors and publishes them to Watson IoT Platform™;

- A server that receives image data from car park cameras or drones, detects parking spot statuses, and publishes the results to Watson IoT Platform™.

3.1.1.3 Applications

An *application* is anything that has a connection to the Internet and interacts with data from devices and controls the behaviour of those devices. Applications identify themselves with the Watson IoT Platform™ by using an API key and a unique application ID. Unlike devices, individual applications do not need to register before they can connect to the Watson IoT Platform™. However, they **must use a valid API key** that has been previously registered.

Examples of applications are:

- The car/ride sharing service, which listens to traffic and environmental events published by devices and gateway through Watson IoT Platform™, and schedules, accordingly, pick-ups, drop-offs, and vehicle routes;
- The AVP service, which listens to events on the car park and to the positions and statuses of vehicle being parked to assign parking spots and routes to the parking spots;
- The autonomous driving (AD) functionality in an AD vehicle, which listens to external events shared through Watson IoT Platform™.

3.1.1.4 Events

Events are the mechanism by which devices publish data to the Watson IoT Platform™. Devices control the content of their messages and assign a name for each event that is sent. The Watson IoT Platform™ uses the credentials that are attached to each event received to determine which device sent the event. This architecture prevents devices from impersonating one another. Applications can process events in real time and see the source of the event and the data contained in the event. Applications must be configured to define which devices and events they subscribe to.

Examples of events are:

- Car probe data submitted by vehicle gateway;
- Identification of an object, obstacle, vulnerable road user (VRU), or hazard on the road;
- Parking spot and occupancy detection.

3.1.1.5 Commands

Commands are the mechanism by which applications communicate with devices. Only applications can send commands, and the commands are sent to specific devices. The device must determine which action to take on receipt of any given command. Devices can be designed to listen for any command or to subscribe to a specified list of commands.

3.1.2 Communication Protocol

IBM Watson IoT Platform™ supports version 3.1.1 of the MQTT messaging protocol [MQTT]. It accepts any content that is permitted by the MQTT standard. MQTT is data-agnostic, so, in theory, it is possible to send images, texts that are in any encoding, encrypted data, and virtually every type of data in binary format.

The IBM open source Watson Developer Cloud APIs (https://github.com/watson-developer-cloud) provide client APIs (https://github.com/ibm-watson-iot) for Watson IoT Platform™ in various programming languages (Java, NodeJS, and Python). These APIs may be used to facilitate interaction with Watson IoT Platform™.

3.1.3 Watson IoT Platform™ Deployment and Access

IBM is providing two Watson IoT Platform™ instances for AUTOPILOT:

- Development: We recommend that you use this instance as a starting point for learning, development, experimentation, and testing purposes. You may want to use this instance to create virtual (fake) devices and test their connection to the platform.
- Production: This instance should be used for official project demonstrations, and for connecting real devices.

Information about the deployment status and access to the above instances is provided in Table 1 below.

The Watson IoT Platform™ instances will be connected to the oneM2M interoperability platform (see section 5) through oneM2M connectors, called interworking proxies, developed by SENSINOV and IBM IE. A oneM2M connector is already deployed on the TNO servers for connecting the development Watson IoT Platform to the oneM2M platform.

Table 1. Watson IoT Platform – Deployment Status and Access

Platform	Watson IoT Platform™		
Hosting	IBM Cloud		
Deployment Status	***Development***	Deployed	Development instance, for learning, development, experimentation, and testing purposes
	Production	Deployed	Production instance, for official project demonstrations, and for connecting the real devices
	OneM2M Interworking Gateway	Deployed	Bidirectional communication between the oneM2M interoperability platform and Watson IoT Platform. One connector for the development instance is deployed on the TNO servers.
Purpose/Pilot Site	Currently used in the Brainport and Vigo pilot sites for car/ride sharing, AVP, and parking spot detection		
Standard/Protocol	Watson IoT Platform™ oneM2M connector: deployed on TNO servers		
URL	***Development***	https://2j73n2.internetofthings.ibmcloud.com	
	Production	https://btjftu.internetofthings.ibmcloud.com	
Connected Devices and Applications	***Development***	Cars/devices/applications collecting and using parking spot availabilities (DLR), car/ride sharing service, parking spot web service, cameras and corresponding applications for vulnerable road user detection (CTAG)	
	Production	None	
Access Process	The Development and Production Watson IoT Platform™ instances are available to all the pilot sites and use cases. Access to these instances may be requested from IBM IE, through one of the following points of contact: • Anton Dekusar adekusar@ie.ibm.com Any devices and device types to connect to the Watson IoT platforms must be registered in advance by the IBM IE point of contact. You will be requested to provide the device or device type data schemas. Once the devices are registered, you will receive credentials for each device. Device credentials must not be shared with other partners or across devices. Data submitted by one device must comply with the data schema provided for the registration of the device. Any changes to the data schemas must be validated by the IBM IE point of contact before being able to submit data from devices.		
Restrictions	***In addition to the above conditions, please note that personal information, images, or videos, must not be sent to the Watson IBM Platform™ instances.***		

3.1.4 Interfacing with Watson IoT Platform™

Example Java source code to interact with Watson IoT Platform™ is provided in the project GitLab repository **iot-central**, under the folder entitled "**watson**".

3.2 FIWARE

This section provides an overview of FIWARE, its features, and deployment status in AUTOPILOT. For detailed information about FIWARE, please refer to the FIWARE wiki [FW].

FIWARE IoT chapter provides the generic enablers (GE) to allow things to become available, searchable, accessible, and usable. In this context, "things" mean any physical object, living organism, person or concept interesting from the perspective of an application and whose parameters are totally or partially tied to sensors, actuators or combinations of them.

IoT chapter uses the NGSI standards for data exchange.

- The NGSI-10 interface [NGSI10] is used for exchanging information about entities and their attribute, i.e., attribute values and metadata.
- The NGSI-9 interface [NGSI9] is used for availability information about entities and their attributes. Here, instead of exchanging attribute values, information about which provider can provide certain attribute values is exchanged.

3.2.1 FIWARE Architecture

IoT chapter architecture deployments vary from simple scenarios (e.g., connecting few devices using standard IoT communication protocols to the data chapter Context Broker) to more complex scenarios distributed across a large number IoT networks, connecting IoT gateways and IoT nodes, and providing advanced composition and discovery functions.

IoT GEs are spread over two different domains as shown in Figure 5:

- **IoT Backend:** Comprises the set of functions, logical resources, and services hosted in a cloud data centre. Northward, it is connected to the data chapter Context Broker, so IoT resources are translated into NGSI context entities. Southward, the IoT backend is connected to the IoT edge elements, that is all the physical IoT infrastructure.
- **IoT Edge**: Comprises all on-field IoT infrastructure elements needed to connect physical devices to FIWARE Applications. Typically, the IoT edge comprises the IoT end-nodes, IoT gateways and IoT networks (connectivity). The IoT Edge and its related APIs facilitate the integration of new types of gateways and devices, which are under definition in many innovative research projects, and warranty the openness of FIWARE IoT architectures.

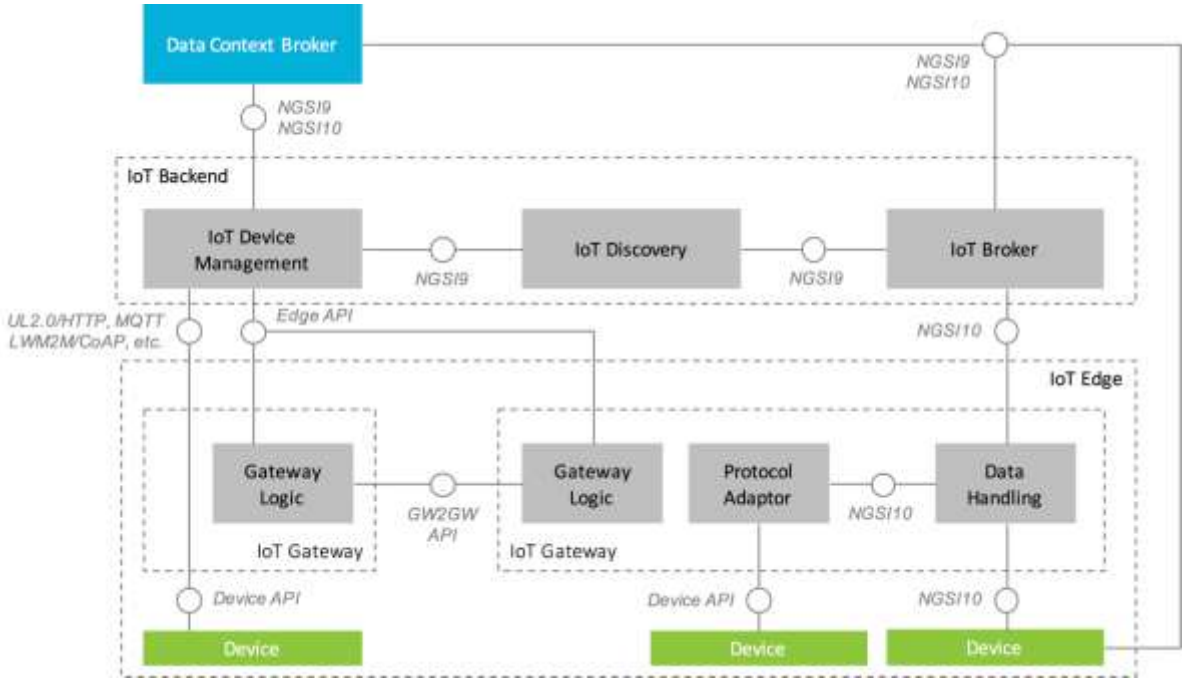

Figure 5. FIWARE IoT Architecture

3.2.2 IoT Broker GE reference implementation: AERON

The IoT Broker is specified as a lightweight and scalable middleware component that separates IoT applications from the underlying device installations. The IoT Broker implementation available through the FIWARE Catalogue is the reference implementation of this Generic Enabler. This implementation satisfies all properties described in the specification of the Generic Enabler. More details about the IoT Broker GE can be found in the FIWARE wiki [FW].

IoT Broker is integrated with three other components: IoT Discovery (ConfMan), IoT Knowledge Server, and FIWARE (Orion) Context Broker. Table 2 provides links to the IoT Broker source code, Docker images and manuals.

Table 2. Web Links to IoT Broker Artifacts

Artifact	Link
Source Code	https://github.com/Aeronbroker/Aeron
Binaries from FIWARE Catalogue	https://catalogue.fiware.org/enablers/iot-broker
Docker Image	https://hub.docker.com/r/fiware/iotbroker/
Documentation	http://fiware-iot-broker.readthedocs.io/en/master/ https://forge.fiware.org/plugins/mediawiki/wiki/fiware
Online Course	https://edu.fiware.org/course/view.php?id=33

3.2.3 IoT Discovery GE alternative implementation: NEC ConfMan

The NEC Configuration Management or NEC ConfMan is an implementation of the FIWARE IoT Discovery Generic Enabler. This implementation is specifically designed to interwork with the IoT Broker GE FIWARE reference implementation, serving as the registry of FIWARE NGSI context providers. The detailed information about the IoT Discovery GE can be found in [FWa].

ConfMan is responsible for discovering the availability of context. The availability of context must be through the registrations made by IoT Agents (i.e., data providers). By registering, data providers make their access endpoints available to the data consumers. Data providers offer information about context entities (i.e., virtual entities) and their attributes (e.g., measurement values,

metadata). The role of IoT Agents can be taken by either the Data Handling GE in IoT Gateways or the Back-end Device Management GE. Other IoT Backend systems may also provide context information.

Typically, context source availability information is forwarded to the FIWARE Context Broker GE. This allows context information (i.e., information generated by or coming from the "things") to be available and accessible to applications. ConfMan is integrated with two other components: IoT Broker and IoT Knowledge Server. The ConfMan source code is accessible at https://github.com/Aeronbroker/NEConfMan.

3.2.4 Crowd Estimation Service

The AUTOPILOT IoT architecture's IoT layer includes the "Analytics" building block. To fulfil the functionality required by this building block, NEC FIWARE-based IoT Platform includes an analytics component which is integrated with FIWARE IoT Broker and IoT Discovery. The name of this component is **Crowd Estimation and Mobility Analytics (CEMA).**

CEMA works on raw data which is received from IoT devices deployed in the field such as Wi-Fi sensors to count the people and estimate crowdedness in certain areas such as road sides. This service can be useful to understand pedestrian traffic and notify the autonomous vehicle beforehand about the expected crowdedness levels of nearby roads. This could be used for better routing planning. Moreover, a Wi-Fi sniffing device with GPS sensors is placed in autonomous cars for mobile sensing.

Application developers can access CEMA analytics results through NGSI subscriptions or queries through the IoT broker. CEMA analytics results are also showcased in a dashboard interface in real time. Furthermore, Semantic Mediation Gateway component sends real-time CEMA analytics results from the FIWARE IoT Broker to the oneM2M using NGSI and MCA interfaces, respectively. Lastly, the format of the CEMA messages follows the data models of the AUTOPILOT. This can enable data consumers of the oneM2M (e.g., autonomous driving vehicles) easily make use this data.

3.2.5 FIWARE Status of Deployment and Access

The deployment status and access procedure of the FIWARE components are summarised in Table 3.

FIWARE Aeron Broker and NEConfMan components are currently deployed in the servers which are provided by NEC in Heidelberg, Germany. These servers are dedicated for the Brainport pilot site and in more general to the central IoT platform. The servers can connect to the oneM2M platform in Brainport pilot site.

Since the server is inside NEC, the components currently reside inside the internal NEC VPN. However, the access to the FIWARE IoT Broker and Discovery can be provided through whitelisting certain IP addresses.

Currently, CEMA service is available through the IoT platform service. CEMA and the IoT platform reside in the NEC server and CEMA outputs can be openly shared with all partners. CEMA is integrated with a lighter version of the IoT Broker we can be referred as the Thin Broker. The CEMA service was tested and access was provided to the use case developers.

Like Aeron, Scorpio instances are currently hosted on an NEC server. Public access can be organized through IP white listing. The source code for Scorpio is publicly available at Github.

Table 3. FIWARE – Deployment Status and Access

Platform	FIWARE
Hosting	NEC Servers, NEC Laboratories Europe, Heidelberg, Germany

Deployment Status	***AERON Broker***	Deployed	Dedicated to the Brainport pilot site
	NEConfMan	Deployed	Dedicated to the Brainport pilot site
	CEMA	Deployed	Dedicated to the Brainport pilot site
	ScorpioBroker	Deployed	Used to test federation of information from multiple pilot sites
	oneM2M Interworking Proxy	Deployed	The FIWARE-oneM2M interworking proxy was successfully tested
Purpose/Pilot Site	Used primarily for the Brainport pilot site		
Standard/Protocol	HTTP (Rest)		
Connected Devices and Applications			
Access Process	Since the server is inside NEC, the components currently reside inside the internal NEC VPN. However, the access to the FIWARE IoT Broker and Discovery can be provided through whitelisting certain IP addresses.		
Restrictions	***In addition to the above conditions, please note that personal information, images, or videos, must not be sent to the NEC FIWARE broker instances.***		

3.2.6 NGSI-LD and Scorpio Broker Implementation

NEC has adapted the new ETSI ISG CIM standard NGSI-LD, which is an evolution of NGSI. The Scorpio Broker implementation is available at https://github.com/ScorpioBroker/ScorpioBroker.

3.2.6.1 NGSI-LD

NGSI-LD defines an API for context interaction as well as an entity based data model to represent context data.

Entity Model

Following its predecessor, the data model of NGSI-LD is centred around the entity concept. An entity is described by its type and a unique identifier. As shown in Figure 6 entities can have properties and relationships. These two types of attributes allow us to have a Digital Twin of a real world object not only clearly representing properties but also relationships to other entities, which can be followed. This enables users of NGSI-LD to represent Digital Twins in a knowledge graph.

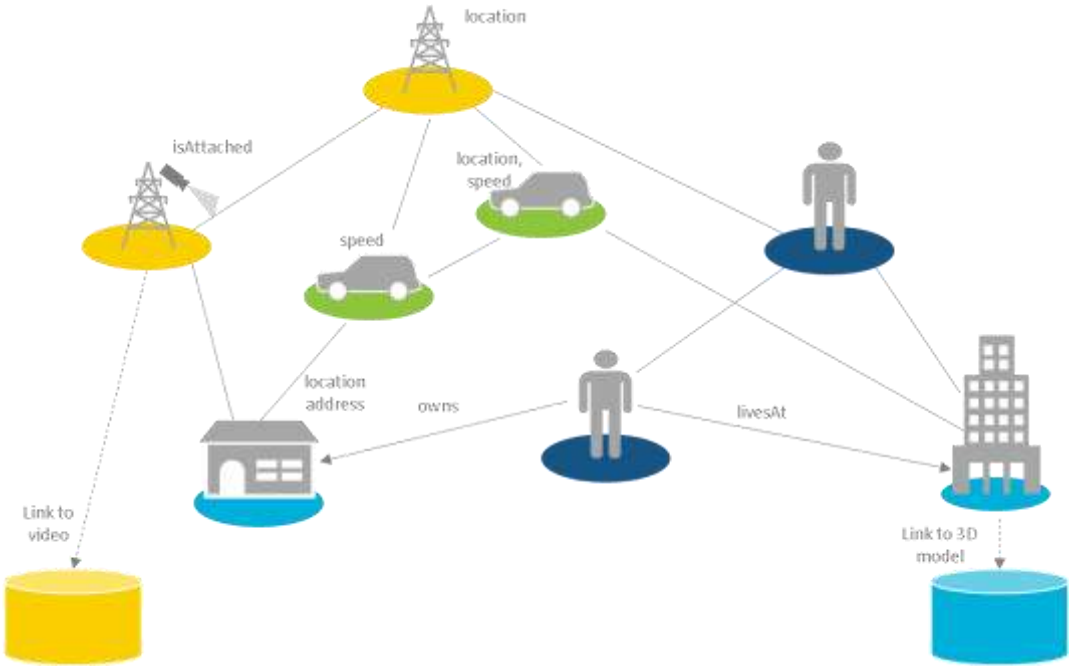

Figure 6. NGSI-LD Knowledge Graph

APIs

NGSI-LD defines a set of APIs which are the successors of NGSI-9 and NGSI-10 interfaces. In the following section we will give a short overview of most important functionalities of the API. The whole spec can be found at the ETSI website. The defined APIs provide methods context information provisioning, i.e. create entity, update an entity, partial update of an entity, append attributes and delete.

For the context information consumption 3 types of access are provided:

- Retrieval of a specific entity by its ID
- Querying entities based on provided filters, e.g. by its type, attribute value or geo location
- Subscription to changes in entities. Filters like in queries can be used here as well

As context data providers defines two types of providers.

- The Context Producer, which will actively push its information towards the broker using the context provisioning methods.
- The Context Source, which is supports the full set of data consumption methods and register the provided entities with the broker. Context data from Context Sources is then pulled by the broker when needed.

For the, above mentioned, registration of available entities NGSI-LD defines the csourceRegistrations interface which is used to provide a set of metadata of the available entities. The registrations of a ContextSource can contain a combination of, available IDs, available types, available attributes, a coverage area, an operation area and additional custom metadata.

@context usage for integration

The @context entry is a fundamental aspect of JSON-LD and hence for NGSI-LD. In JSON-LD all entries must be represented as URI, which can make the payload of a document rather big. Therefore JSON-LD has defined the @context entry which does not only allow to map the URIs to short names but also to map more complex structures in a document into a simplified version. Addition a set of functions to compact, expand and flatten documents based on the @context entry was defined by JSON-LD.

3.2.6.2 Scorpio NGSI-LD implementation, deployment, and integration

NEC has made one of the first implementation of the new NGSI-LD standard called the Scorpio Broker. In order to have flexible deployment options and scalability Scorpio is designed using a microservice architecture. As technology base Scorpio is using Spring Cloud to have individual microservices implementing functional aspects of the NGSI-LD standard.

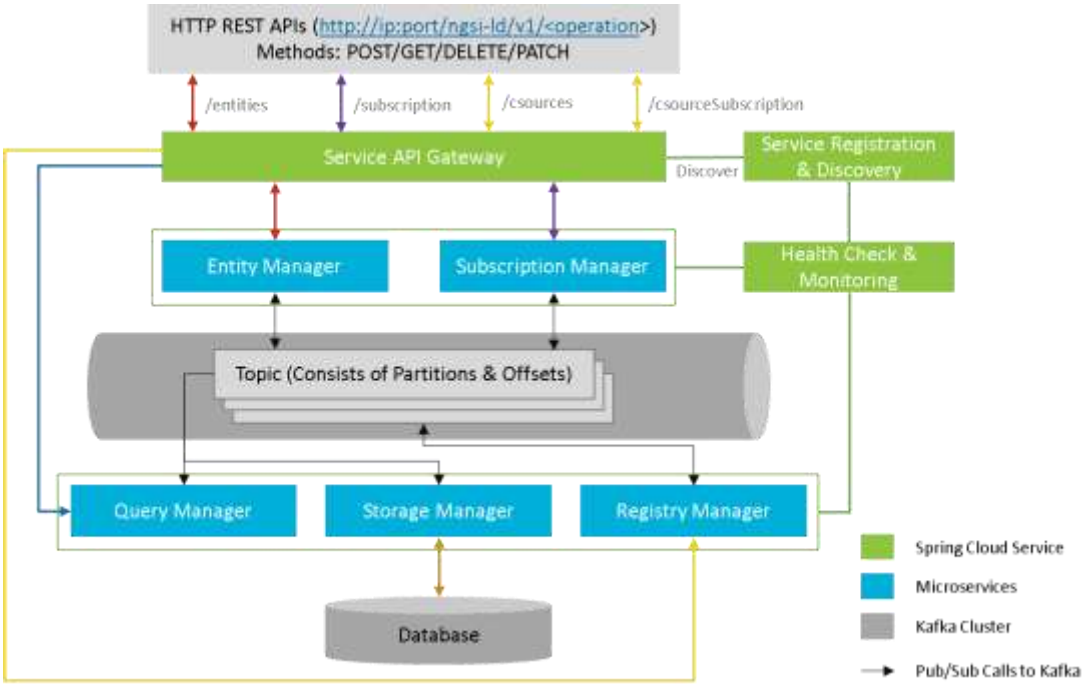

Figure 7. Scorpio Architecture

As shown in Figure 7, there is a micro service for entity creation, updating, appending and deletion request, the EntityManager. Two microservice for entity retrieval, the QueryManager and the SubscriptionManager, which respectively provide a query and subscription functionality for entities. Another microservice is the RegistryManager, which provides access to the CSourceRegistration information of context sources and context providers accessible through the broker. Additionally, there is the Storage Manager, which unlike the other microservices does not provide a REST interface towards the user but is responsible to properly persist entities and is supporting the QueryManager by using the capabilities of the used Postgres database to efficiently perform complex queries. For a fast and reliable communication between the microservices the Apache Kafka message bus is used.

3.2.6.3 Federation setup

Scorpio is supporting a federated setup of multiple instances. The federation setup is realized through the Context Source Registry. For AUTOPILOT the suggested setup is a hierarchical federation.

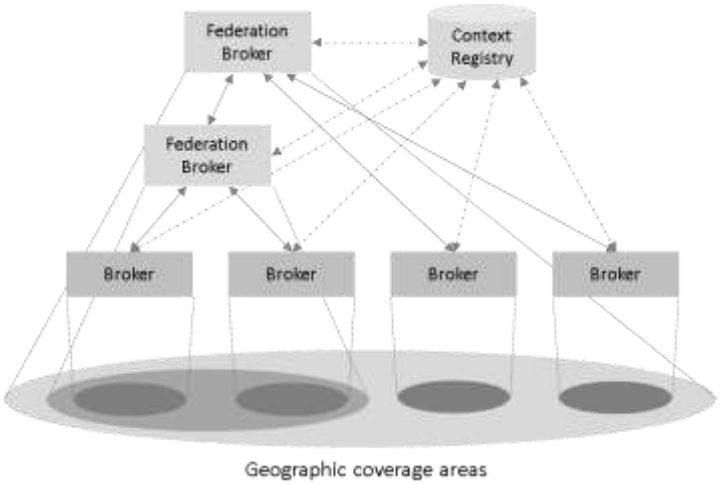

Figure 8. Federation setup based on geo properties

As shown in Figure 8, each broker knows a federation broker, which provides access to all its children. Siblings do not have a direct relationship but communicate through their federation broker. The suggested federation criteria for AUTOPILOT, but also other use cases, are to design a federation based on geographical spatial information.

This means a service relying on NGSI-LD would ask for context information filtered by a geographic area to a European top-level broker. The top-level broker would be based on geo data and additional filters, like type of entity, look up the which of its children to ask for the context information. This behavior is redone by the lower level brokers till the request information is reached.

3.2.6.4 Integration with FIWARE IoT Brokers

The NGSI-LD data format is not directly backward compatible with the NGSI V2 data format. Even though a basic backward compatibility can be provided with a correct usage of the @context entry, some of the existing information, especially the metadata field in NGSI V2, can be lost or not correctly interpreted, e.g. relationships might by indicated by a prefix or an entry is a timestamp and that information is stored in the metadata.

Hence, we developed two approaches to overcome this problem. In our initial approach we adapted an existing NGSI V2 to NGSI-LD translator, provided by FIWARE members, to support the most common NGSI V2 data scenarios, in the AUTOPILOT context. Based on this translator we created a Context Producer for NGSI V2 data sources.

Additionally, we developed a configurable NGSI V2 Input microservice for Scorpio. This microservice has all the existing common scenarios from the translator as well as configuration options to define which NGSI V2 attribute prefix is a relationship and which metadata indicates which type of attribute is provided.

3.3 HUAWEI OceanConnect Platform

This section provides an overview of the Huawei OceanConnect IoT platform based on the official Huawei documentation [HOC].

The Huawei OceanConnect platform is an open ecosystem built on IoT, cloud computing, and Big Data technologies. It provides over 170 open APIs and serial agents that enable application integration, simplify and accelerate device access, guarantee network connection, and realise seamless connection between upstream and downstream products for Huawei partners.

The Huawei OceanConnect IoT platform connects in a secure way IoT devices to the IoT cloud platform, enabling bidirectional communication to easily collect data and deliver commands between devices and the platform.

The key features provided by the Huawei OceanConnect IoT platform are:

- **Agile and Easy-to-Use Device Integration:** OceanConnect supports more than 20 mainstream cable and wireless IoT protocols. It is also pre-integrated with mainstream IoT chips and modules.
- **Complete and Highly Efficient Device Management**: OceanConnect's device management portal provides powerful functions and a user-friendly GUI for managing and configuring devices, visualising their statuses, identifying faults, and performing firmware/software upgrade and maintenance.
- **Flexible and Open Applications**: The platform provides over 170 APIs and functions for device management, data, and rules, allowing users to quickly create network.
- **Highly-Concurrent and Highly-Reliable Cloud Services:** Hundreds of millions of connections are supported with a service reliability of 99.9%. In addition, the Huawei OceanConnect IoT platform provides E2E security protection, device-level certification/authentication, low-power optimisation, and complete application-level access control.

3.3.1 Huawei OceanConnect IoT Platform Architecture and Capabilities

Figure 9 depicts the Huawei OceanConnect IoT platform architecture. The supported functionalities are briefly described in this section. For more further please refer to the platform official documentation [HOC].

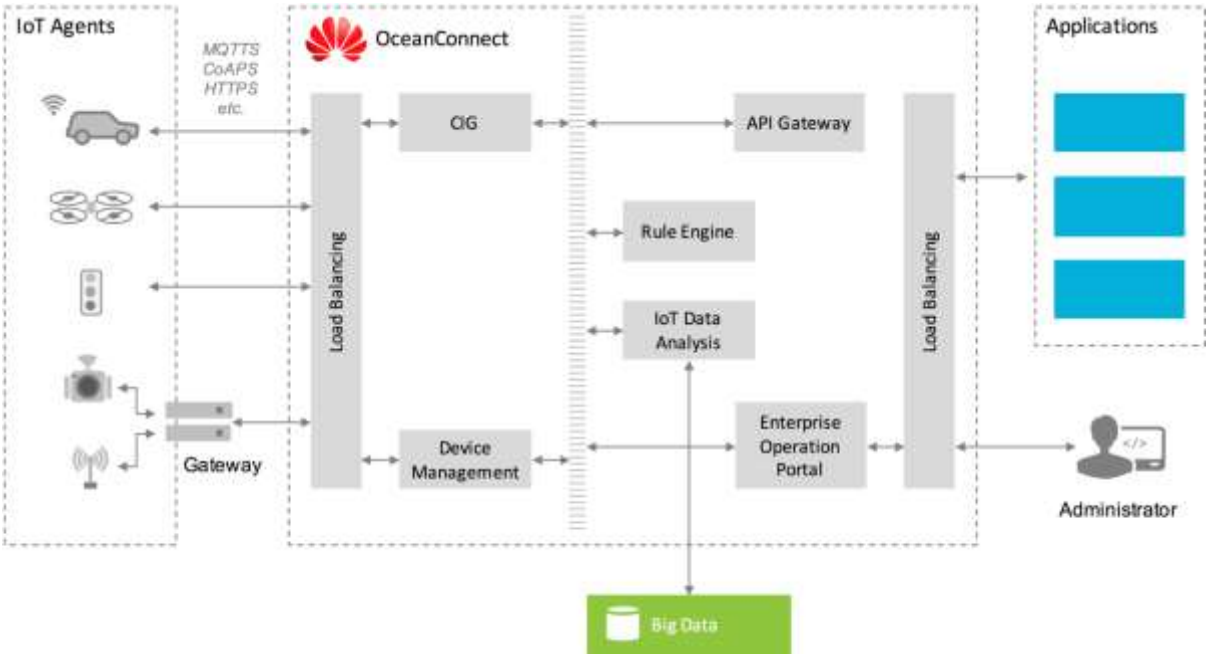

Figure 9. HUAWEI OceanConnect IoT Platform Architecture

- **Cloud Inter-Networking Gateway (CIG):** The CIG is OSGi-based [OSGi] and comprises modules that transmit messages between IoT devices and the IoT platform and supports multiple communication protocols including TCP, UDP, MQTT, CoAP, and LWM2M.
- **IoT Agent:** Serialised IoT Agents (such as Agent Rich, Agent Lite, Agent Tiny, and Agent IPC) are deployed on different types of gateways and IoT devices. Multi-vendor devices can be quickly integrated with the cloud platform. The Agents are pre-integrated with near-field communication protocols such as Z-Wave, ZigBee, Wi-Fi, and Bluetooth, and are used to manage data links.
- **Device Management:** Bidirectional data channels between devices and the IoT platform are used for device data reporting and remote control, including full-lifecycle management functions, such as getting devices online, maintenance, network connections, alarms, report analysis, upgrade, and deregistration.
- **Rule Engine:** Simplified and flexible rules enable linkage and triggering of messages, notifications, and alarms between devices.
- **IoT Data Analysis:** The Huawei OceanConnect IoT platform processes concurrent real-time data, stores mass data, computes data, and exposes data APIs.
- **Service Operation and Management Portal:** The portal consists of the following modules: application management, device management, reports management, rule engine, software management, sub-account management, and service status statistics.
- **API Gateway:** OceanConnect exposes over 170 types of functions, such as device management, rule engine, and data analysis, helping developers quickly create new applications.

3.3.2 Connecting Applications and Devices to OceanConnect

This section provides the high-level principles of connecting applications and devices to OceanConnect. For more details, please refer to section "Open Capabilities of OceanConnect" of the platform's official documentation [HOC].

Applications requiring access to OceanConnect need to be authenticated first. Once an application is successfully authenticated it may perform the following actions:

- Collect device data from the IoT connection management platform using either an active query or data subscription.
- Issue commands to a specified sensor through the IoT connection management platform.
- Issue rules to the IoT connection management platform allowing response events and commands to be triggered based on the rules.
- Subscribe to device information (events) from the platform.

Devices may be connected to OceanConnect in two steps:

1. Connect the device to the IoT connection management platform either directly or indirectly through a gateway. Direct connection may rely on near-end integration (i.e., device integrates the IoT Agent Lite with the system and invokes the interface opened by the Agent Lite) or remote interrogation (i.e., device is interconnected to the CIG, and a plug-in is compiled and sent to the CIG to enable the device to access the platform). Gateways may be connected through an Agent (Rich) or Agent Lite depending on the network capabilities and the communication protocol used between the device and the gateway
2. Based on the selected access mode, determine the Agent to be used, and then use the API provided by the Agent to complete integration development and implement data reporting, and command receiving.

3.3.3 Interacting with OceanConnect: Open APIs

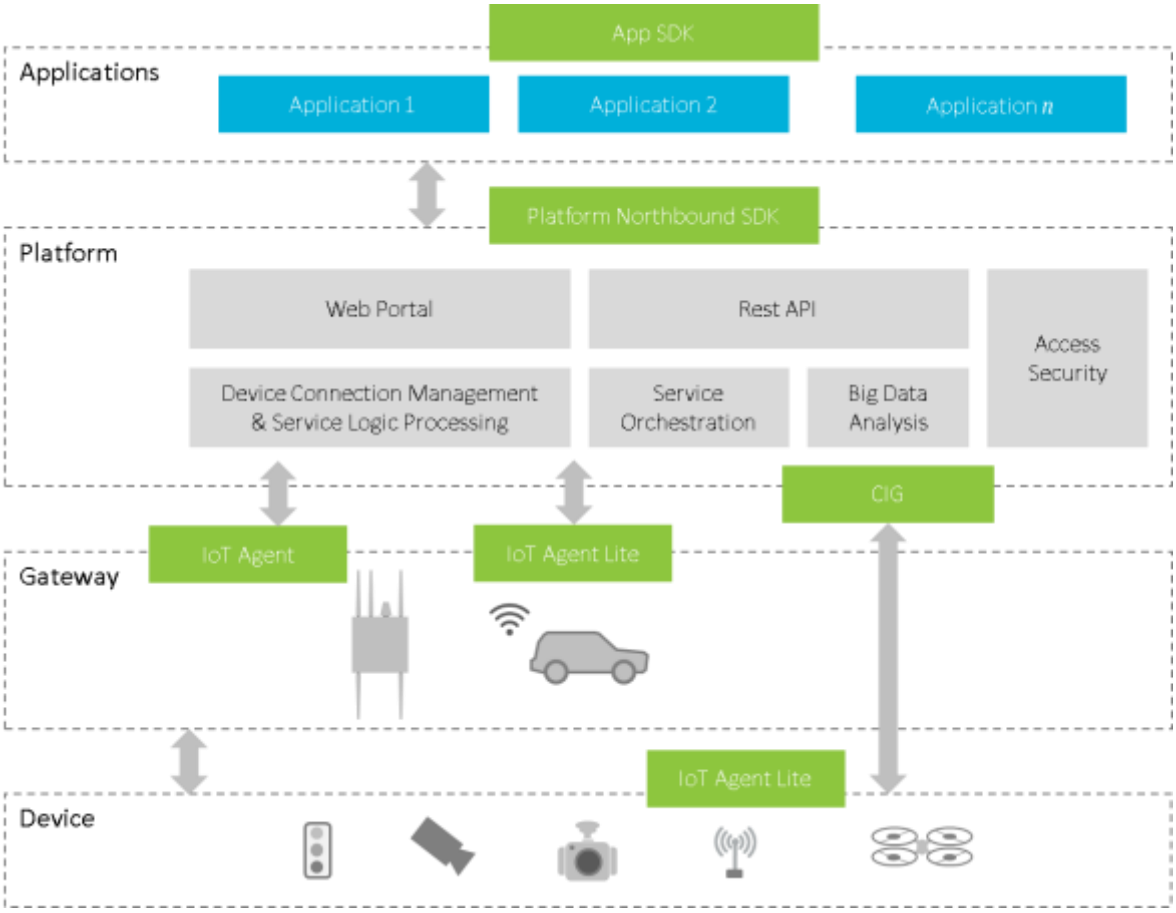

Figure 10. Overview of Huawei OceanConnect IoT Platform open APIs

OceanConnect provides a set of APIs for developers to use for different purposes. The OceanConnect IoT Platform open APIs are depicted in Figure 10 and introduced below. For further details, please refer to section "Open APIs of OceanConnect" of platforms official documentation [HOC].

- **API Software Development Kit (SDK):** Provides developers with interfaces to integrate (or newly develop) open capabilities of the OceanConnect on a mobile APP.
- **Platform Northbound API:** Supports six capabilities opened to application developers, including authentication, device management, data collection, device service invocation (command issuing), rule issuing, and message pushing.
- **Agent:** Provides a gateway integration API, a Z-wave sensor integration API, and a ZigBee sensor integration API.
- **Agent Lite:** Currently, the Agent Lite is provided through the SDK, and the supported development languages are Android and the C programming language. The Agent Lite is applicable to gateway integration and device integration.
- **CIG:** Provides an API for northbound sensor integration. On the CIG, Northbound transport protocol and integration have already been completed (currently, only interconnection with the Huawei Northbound module is supported). Developers only need to develop the data format conversion plug-in on the CIG. The CIG also provides APIs that can be invoked by developers. The CIG API document and development guide are not yet released on the Huawei Developer website.

3.3.4 Huawei OceanConnect - Enterprise Application Integration (EAI) IoT Server

The Huawei OceanConnect platform provides Device Connection Management through Service Orchestration, see Figure 10. The collected IoT data is stored, processed and forwarded to back end systems, e.g., applications.

There might be situations where the Huawei OceanConnect platform needs to support additional requirements and provide the back end system internally, in Huawei OceanConnect, and extend its functionalities. In this case, additional modules, such as the EAI (Enterprise Application Integration) IoT server described in this section, can be used as a middleware or a backend system with tailored functionalities to support these additional requirements.

This Huawei OceanConnect EAI middleware offers:

- Rapid Prototyping
- Efficiency, Robustness and Full Scalability
- Monitoring and Quality of Service
- Can be deployed and distributed also on other hosts
- It complies with all known communication standards.

In particular, the Huawei OcenConnect EAI supports the following key functionalities, see Figure 11:

- Geofencing Engine: used to enable the Geofencing/Geofetching service in the Car Rebalancing use case. It is implemented using Geolocation information received from IoT devices, such as vehicles and smart phones and geofencing windows received from vehicles that need to locate IoT devices located in these geofencing windows. The used communication protocols are MQTT and HTTP.
- Virtual Device Management: used to manage Virtual Devices enabling remote access.

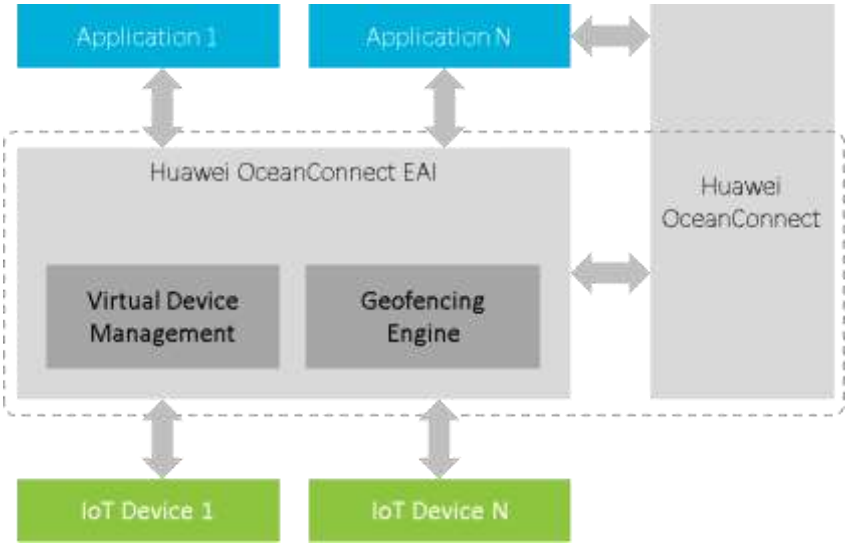

Figure 11. Overview of Huawei OceanConnect EAI IoT Server

Figure 12 shows the possible communication interfaces between the Huawei OceanConnect EAI and the Huawei OceanConnect Ecosystem components.

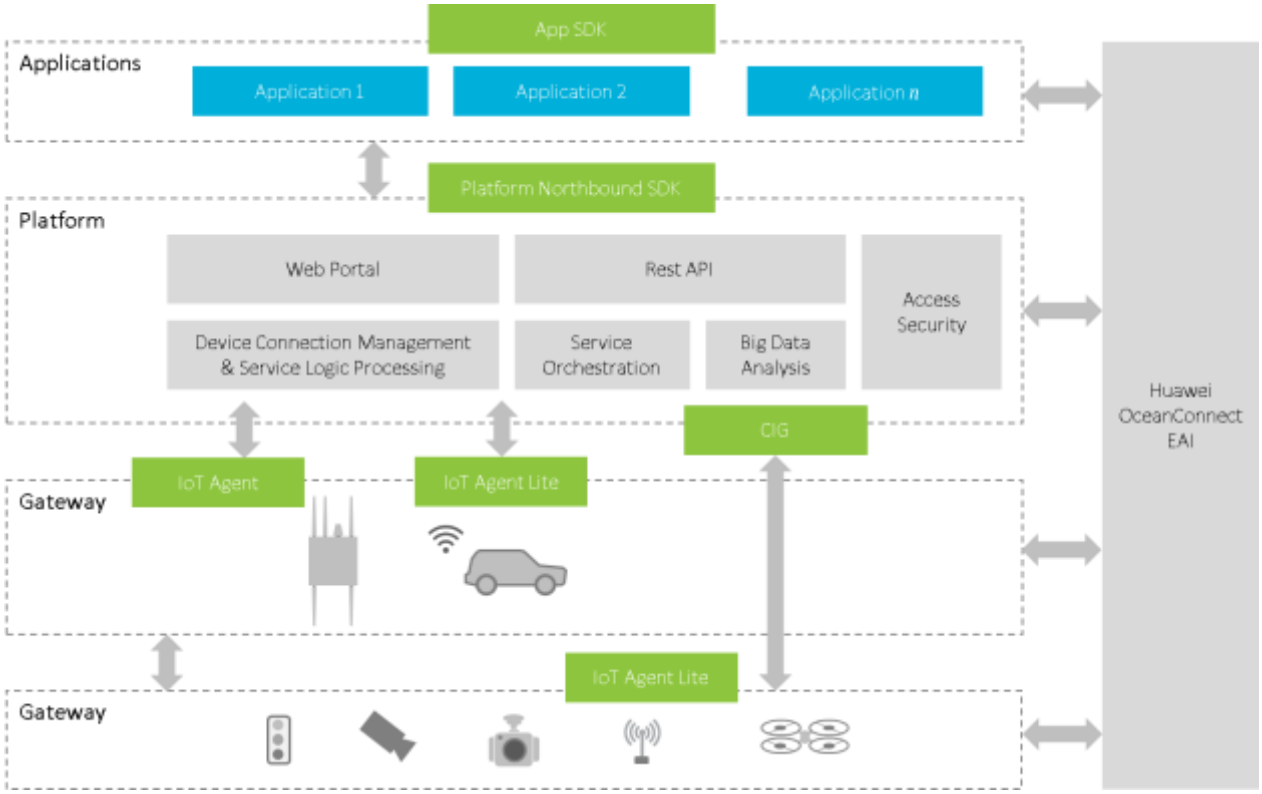

Figure 12. Huawei OceanConnect EAI positioning in the Huawei OceanConnect ecosystem

The Huawei platform is also supporting an oneM2M Interworking Proxy Entity (IPE), which is a specialized oneM2M AE (Application Entity) that allows the oneM2M system to interact with any non-oneM2M system, in a seamless way, through the Mca interface, see [oneM2M-TR-0039]. It has the capability to remap a specific data model to oneM2M resources and maintain bidirectional communication with the non-oneM2M system. In particular in the case of the Huawei IoT platform the IPE is remapping the MQTT interface to a Mca interface and vice-versa.

The IPE is integrated in the EAI and it supports the oneM2M MCA interface (using communication pipes) towards/from the oneM2M platform.

3.3.5 OceanConnect Deployment and Access

HUAWEI provides an instance of OceanConnect for the Brainport pilot site. A summary of the deployment status and access to the platform is provided in Table 4.

Table 4. HUAWEI OceanConnect Platform – Deployment Status and Access

Platform	HUAWEI OceanConnect IoT Platform	
Hosting	OceanConnect Platform	
Deployment Status	IoT Platform	One development instance is deployed and available to use, for learning, development, experimentation, and testing purposes
	oneM2M Interworking Proxy	The oneM2M interworking proxy is implemented and used in the care rebalancing use case
Purpose/Pilot Site	To be used primarily for the Brainport car rebalancing/relocation	
Standard/Protocol	HTTP/MQTT	
URL	http://developer.huawei.com	
Connected Devices and Applications	Solutions cover Smart Home, Internet of Vehicles, Public Utilities	
Access Process	The relevant to the Brainport car rebalancing/relocation development instances are	

	available to the Brainport pilot site. Access to these instances may be requested from Huawei Technologies, through the following points of contact: • Liuxin Walle walle.liuxin@huawei.com • Lorenzo Viola lorenzo.viola@huawei.com Further instructions and conditions regarding access will be provided in due time.
Restrictions	***In addition to the above conditions, please note that personal information, images, or videos, must not be sent to the Huawei platform for Automotive service instances.***

3.4 TIM IoT Platform

3.4.1 Overview of the TIM IoT Platform

The TIM IoT platform is based on the oneM2M standard [ONE17]. TIM provides the oneM2M platform as PaaS (Platform as a Service). Figure 13 shows a high-level architecture of the platform.

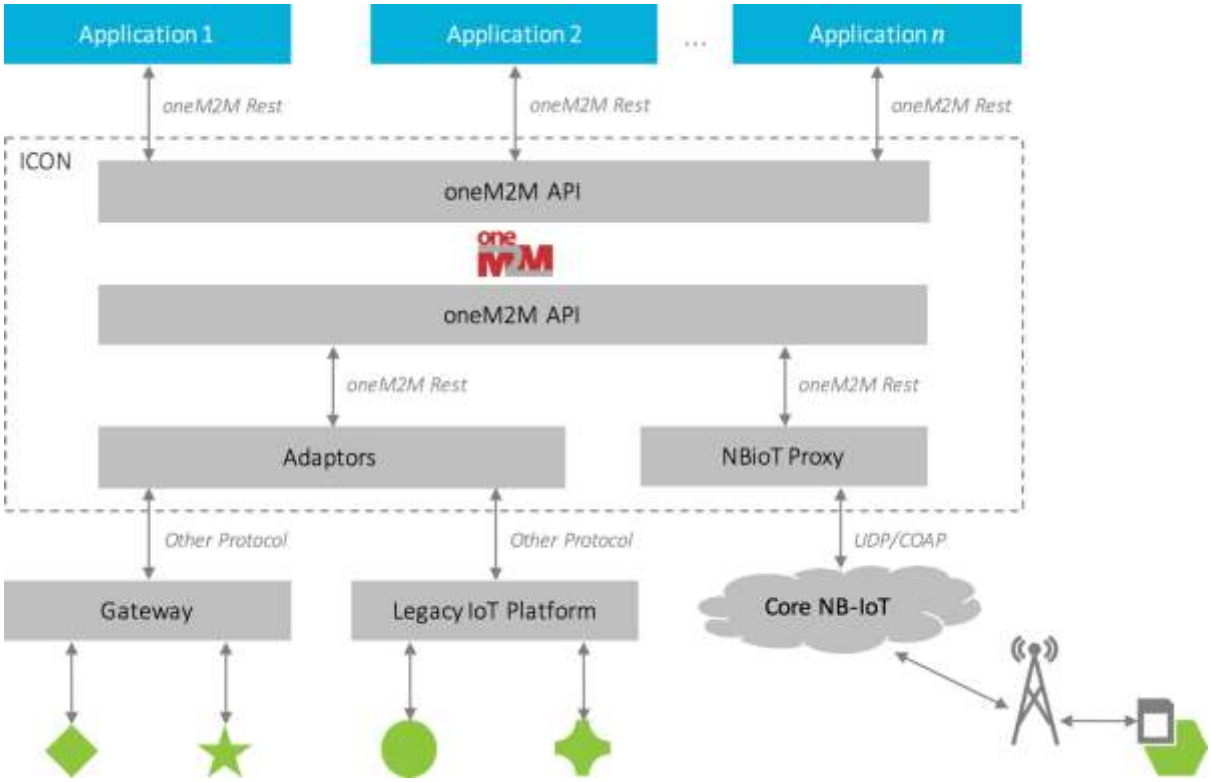

Figure 13. High-Level Architecture of the TIM oneM2M Platform

The platform is based on the Ocean [OIP17] open source project. Its main features are:

- Compliance with the oneM2M standard,
- Southbound and northbound Rest APIs for data storage and sharing,
- Data sharing by means of pull and push (subscription/notification),
- URIs for identifying resources,
- Web console for resource management and provisioning,
- Web console for administrators.

Security in the TIM oneM2M platform is based on the following features.

- oneM2M services and APIs are exposed through SSL (HTTPS).
- Authorisation is based on credentials (username/password) associated with a specific user (tenant).
- An Access Control Policy (ACP) needs to be created for each Application Entity. ACP is defined as a set of conditions that determine whether entities are permitted to access a

protected resource.

3.4.2 TIM oneM2M Platform Deployment and Access

TIM provides an instance of its oneM2M IoT platform on TIM Self Data Center, a commercial platform for hosting, managed by TIM. The platform is exposed on public Internet at https://icon-lab.tim.it. The TIM oneM2M status of deployment and access is summarised in Table 5.

Table 5. TIM oneM2M Platform – Deployment Status and Access

Platform	TIM oneM2M Platform
Hosting	TIM Self Data Center
Deployment Status	One instance is deployed and available to use

Purpose/Pilot Site	Collect data from all the Livorno pilot site use cases:
	• Highway use cases:
	  ○ Hazard (puddles) on the roadway
	  ○ Roadway works with traffic control centre in the loop
	• Urban Driving use cases:
	  ○ Pedestrian detection with camera
	  ○ Connected bicycle
	  ○ Potholes detection

Standard/Protocol	oneM2M
URL	https://icon-lab.tim.it
Connected Devices and Applications	Environment sensors (temperature, humidity, pollution, rain, sound level pressure), parking status, traffic flows, smart waste, smart green, WiFi scanner, metering (power consumption)
Access Process	The instances are available to all the pilot sites and use cases. Access to these instances may be requested from TIM; any devices and device types to connect to the oneM2M platform must be registered and provisioned in advance by TIM. Once the devices are registered, you will receive credentials for each device/application.
Restrictions	***Please note that personal information, images, videos, documents must not be sent to the oneM2M platform***

3.4.3 Interfacing with the TIM oneM2M Platform

The oneM2M platform is accessible through HTTP(S) APIs, using GET and POST to read/write data. Table 6 provides the main HTTP methods needed to interact with the platform.

Table 6. Interfacing with the TIM oneM2M Platform

Posting Data to the TIM oneM2M Platform
The body of the message contains the data that to post to the TIM oneM2M platform. **Method:** HTTP POST **URL Pattern:** https://icon-lab.tim.it/onem2m/<APPLICATION_ENTITY>/<CONTAINER>
Getting Dara from the TIM oneM2M Platform
The body of the response message contains the data retrieved from the TIM oneM2M platform. **Method:** HTTP GET: **URL Pattern:** https://icon-lab.tim.it/onem2m/<APPLICATION_ENTITY>/<CONTAINER>
Subscribing to Notifications from the TIM oneM2M Platform
It is possible to subscribe to notifications of data being received by a container. The mechanism involves creating a Subscription resource within the container in question, indicating the target endpoint where you will notification should be posted (HTTP POST).

Method: HTTP GET: **URL Template:** https://icon-lab.tim.it/onem2m/<APPLICATION_ENTITY>/<CONTAINER>

HTTP is the most used protocol, but the platform also supports MQTT(S) [TS10] and CoAP [TS08] oneM2M bindings.

3.5 SENSINOV oneM2M Platform

The SENSINOV oneM2M platform is used as the interoperability bridge for linking the AUTOPILOT IoT platforms. Its main features are listed below.

- **Supported Nodes:** IN-CSE, MN-CSE, ASN-CSE, and ADN
- **Reference Points:** Mca and Mcc interfaces
- **Resource Types:** CSEBase, RemoteCSE, ACP, ASAR, AE, Container, ContentInstance, Subscription, Group, Node, Request, PoA, Discovery, Notification, etc.
- **Request Primitives:** Retrieve, Create, Update, Delete, Discovery, and Notify
- **Interworking Proxies:** Watson IoT Platform, FIWARE Context Broker OceanConnect IoT platform
- **Addressing Formats:** Structured and unstructured
- **Addressing Modes:** Absolute, SP-relative, and CSE-Relative
- **Protocol Bindings:** HTTP, CoAP, MQTT, and Websocket.
- **Content Formats:** XML and JSON
- **Communication Modes:** blocking, non-blocking synchronous, and non-blocking asynchronous
- **Multi-Hop:** Retargeting via PoA.
- **Storage:** SQL and NoSQL (SQL H2 by default)
- **Security:** SSL/TLS
- **User Interface:** Web interface for browsing oneM2M Resources and command line OSGi console

3.5.1 SENSINOV oneM2M Platform Deployment and Access

The SENSINOV oneM2M status of deployment and access is summarised in Table 7.

Table 7. SENSINOV oneM2M Platform – Deployment Status and Access

Platform	SENSINOV oneM2M Platform
Hosting	TNO Contabo SENSINOV Digital Ocean cloud
Deployment Status	Two instances are deployed and available to use for Brainport and Versailles pilot sites.
Purpose/Pilot Site	Interoperability between IoT devices and IoT backend applications
Standard/Protocol	oneM2M, HTTP, MQTT, CoAP and Websocket
URL	Brainport: • oneM2M dashboard: https://vmi137365.contaboserver.net • oneM2M API: https://vmi137365.contaboserver.net:844 • Resource monitoring: http://vmi137365.contaboserver.net:19999 Versailles • oneM2M dashboard: https://dev1.sensinov.com • oneM2M API: https://dev1.sensinov.com:8443 Resource monitoring: http://dev1.sensinov.com:19999
Connected Devices and Applications	Vehicle, stations and pedestrian sensors
Access Process	The instances are available to all the pilot sites and use cases.

	• Access to Versailles instance should be requested from SENSINOV; any devices and device types to connect to the oneM2M platform must be registered and provisioned in advance by SENSINOV. (Please contact Mahdi Ben Alaya benalaya@sensinov.com) • Access to Brainport instance should be requested from TNO; any devices and device types to connect to the oneM2M platform must be registered and provisioned in advance by TNO. (Please contact Daan Ravesteijn daan.ravesteijn@tno.nl) • Once the devices are registered, you will receive credentials for each device/application.
Restrictions	Please note that personal information, images, videos, documents must not be sent to the oneM2M platform

3.5.2 Interacting with the SENSINOV oneM2M Platform

The oneM2M platform is accessible through HTTPS APIs to read, update, post and delete resources. Section 8.1 in the Annex describes main HTTP methods required to interact with the platform.

3.6 Technolution MobiMaestro Platform

MobiMaestro is a Technolution proprietary IoT platform, used by traffic management centres (TMC) in the Netherlands and Denmark.

3.6.1 Overview of the MobiMaestro Architecture

Figure 14 shows a high-level architecture of the MobiMaestro architecture.

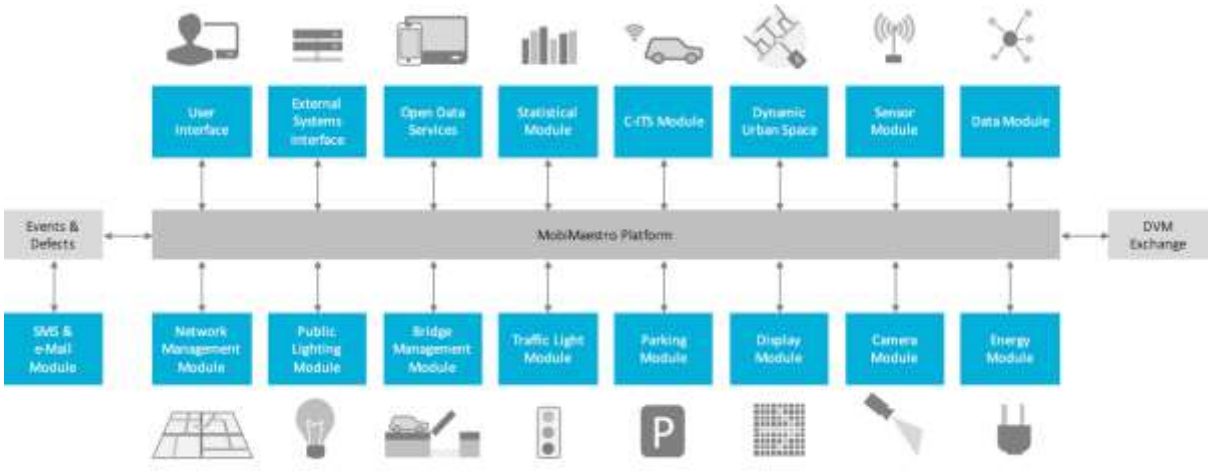

Figure 14. High-Level Architecture of the Technolution MobiMaestro Platform

MobiMaestro is a platform for coordinated network-wide mobility management. It adjusts traffic strategy to meet policy goals, integrates traffic tools and works in an open architecture in order to get everything connected. The platform can be constructed modularly as depicted in Figure 14. For the AUTOPILOT project, MobiMaestro has been adapted as a true IoT-compliant platform considering the target architecture illustrated in Figure 3. To achieve federated architecture Service Bus component of the current MobiMaestro platform has been reused as a proprietary IoT platform and connected to the oneM2M interoperability platform using oneM2M interworking gateway.

The MobiMaestro IoT platform contributed TMC information to the Brainport pilot. It provided, through the oneM2M interoperability platform, lane-information (shoulder access in the platooning use case), traffic light information (urban driving and platooning use cases), and parking information

(valet parking use case).

3.6.2 MobiMaestro Platform Deployment and Access

Technolution provides an instance of its IoT MobiMaestro in the (KPN) data centre, a commercial platform for hosting, managed by Technolution. The platform is exposed on public Internet. The MobiMaestro status of deployment and access are summarised in Table 8.

Table 8. MobiMaestro – Deployment Status and Access

Platform	Technolution MobilMaestro Platform
Hosting	KPN Data Centre managed by Technolution
Deployment Status	One instance is deployed for use
Purpose/Pilot Site	Exchange TMC data with the Brainport pilot site use cases: ○ Floating Car Data ○ General data on road status ○ Hazardous events ○ Actual speeds ○ Traffic Light information ○ Parking information
Standard/Protocol	Datex II / ETSI / SPDP via a MQTT broker and SDK
URL	by request
Connected Devices and Applications	Datasources for roadstatus and FCD.
Access Process	The relevant developments for car platooning and car rebalancing/relocation instances are available to the Brainport pilot site. Main users for the instances are TNO and TU Eindhoven. Access to these companies is taken care of. Other access may be requested from Technolution. All access is user ID and password protected.
Restrictions	***Please note that personal information, images, or videos, must not be sent to the MobiMaestro instance.***

3.7 OpenMTC

OpenMTC is a reference implementation of the oneM2M standard, for conducting applied research and developing innovative M2M and IoT applications. Most of the oneM2M Rel. 2 core features are implemented in OpenMTC. OpenMTC provides protocol bindings for HTTP and MQTT.

OpenMTC is used in the Finnish pilot by VTT. Figure 15 shows the architecture of the implementation of the platform at the Finnish pilot site. In the Finnish site, devices exchange information through the MQTT broker to ensure low latency, and the information is also published in the IoT platform.

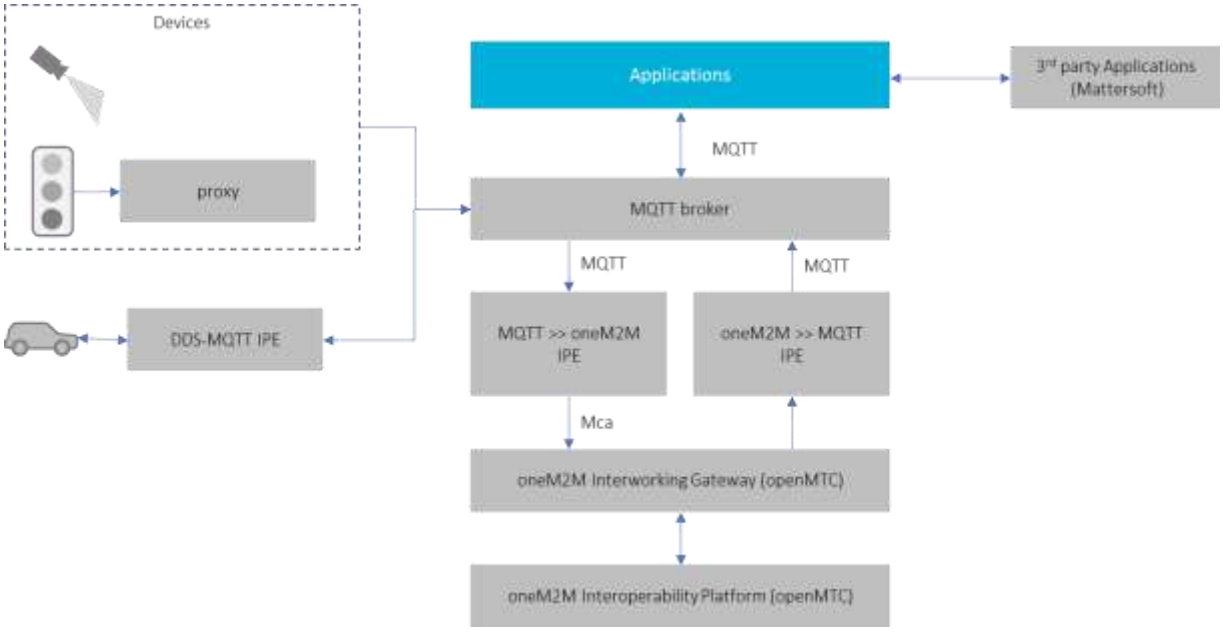

Figure 15. Implementation of oneM2M in the Finnish pilot

3.7.1 openMTC Platform Deployment and Access

The openMTC status of deployment and access in the Finnish pilot site is summarised in Table 9.

Table 9. OpenMTC platform implementation – Deployment Status and Access

Platform	OpenMTC oneM2M Platform
Hosting	VTT Automated Vehicles laboratory
Deployment Status	One instance is deployed at the Tampere pilot site.
Purpose/Pilot Site	Main IoT platform in the Tempere pilot site
Standard/Protocol	HTTP/MQTT, oneM2M standard
URL	By request
Connected Devices and Applications	Vehicle, Road side unit
Access Process	Access is on demand. Access details can be requested from VTT, Tero Peippola, tero.peippola@vtt.fi
Restrictions	***Please note that personal information, images, or videos, must not be sent to the OpenMTC instance.***

3.8 MESIM IoT Platform

MESIM IoT Platform supports service-based operations, hierarchical real-time monitoring, easy creation of services through the service development tool. It supports such IoT standards for messaging as CoAP and MQTT. The platform also provides support for SOAP and HTTP based protocols. Figure 16 depicts architecture of the platform.

Core features and functions provided by the platform:

- Service based operations including SOA support, service repository for devices, and service routing and orchestration.
- GUI based IDE for fast service developing leveraged by packaged templates and components.
- Semantic based information management and query processing for IoT resources
- Broad support for IoT messaging infrastructure based on IETF CoAP, MQTT, TCP/IP SOAP-over-UDP, Binary XML non-IP and others.

- Sophisticated tools for monitoring resources, services, and devices.

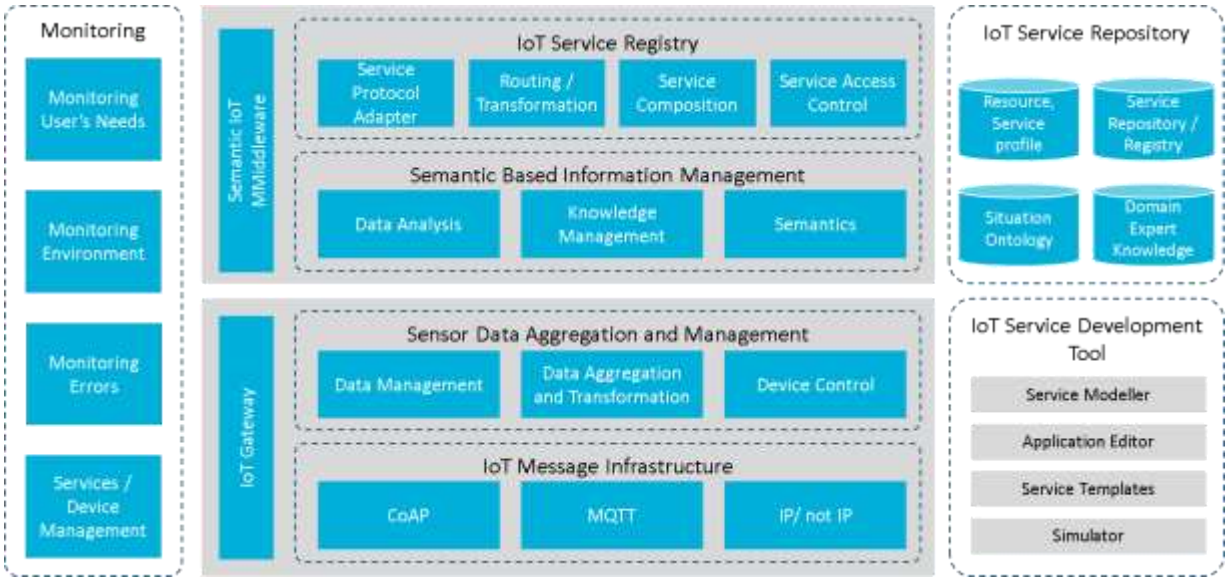

Figure 16. MESIM IoT platform architecture

3.8.1 MESIM IoT Platform Deployment and Access

The MESIM IoT platform status of deployment and access in the Korean pilot size is summarised in Table 10.

Table 10. MESIM IoT platform implementation – Deployment Status and Access

Platform	MESIM IoT Platform
Hosting	At the Korean pilot site premises
Deployment Status	One instance is deployed at the Korean pilot site.
Purpose/Pilot Site	Main IoT platform in the Korean pilot site
Standard/Protocol	HTTP/MQTT, HTTP/SOAP
URL	By request
Connected Devices and Applications	Vehicle, Road side unit, Road radar
Access Process	Access is on demand. Access details can be requested from ETRI, Hyunseo Oh, hsoh5@etri.re.kr
Restrictions	***Please note contact the owner of the platform for the set of restrictions.***

3.9 Conclusion

Use case developers in all the pilot site used an IoT platform of their choice out of the listed platforms. Some partners exchanged data through the oneM2M IoT platform when it is required by their use case. So, here we may say that interoperability and federated IoT platform are key features allowing implementation of use cases with various partners and vendors involved. The Watson IoT Platform and FIWARE, Sensinov and Watson IoT Platform interworking gateways have been tested. All the required interworking gateways are deployed and tested with data. Note that the TIM oneM2M platform is already oneM2M-compliant. The interworking gateways facilitate interoperability between the platforms, which help make the use cases uniform across all the pilot sites.

All deployed IoT platforms are available and accessible for the partners who are responsible for developing the use cases. Moreover, the IoT platform providers provide necessary technical support for the easy usage of the IoT platforms and maintain the server components.

4 Pilot Site IoT Ecosystems

This chapter provides an overview of the implementation of the pilot site IoT ecosystems for the pertaining use cases. This includes the deployed and planned IoT platforms and devices.

4.1 Finland (Tampere)

4.1.1 Overall IoT Architecture of the Finnish Pilot Site

The Finnish pilot site ecosystem involves AD vehicles with a oneM2M IoT platform (openMTC), a mobile road-side unit on which a traffic camera is installed, and connections to a traffic light server, a user interface for the parking maneuver and a system for parking reservations. An MQTT broker allows real-time exchange of messages between the different components. A parking management application has been developed, for managing and monitoring the parking maneuver.

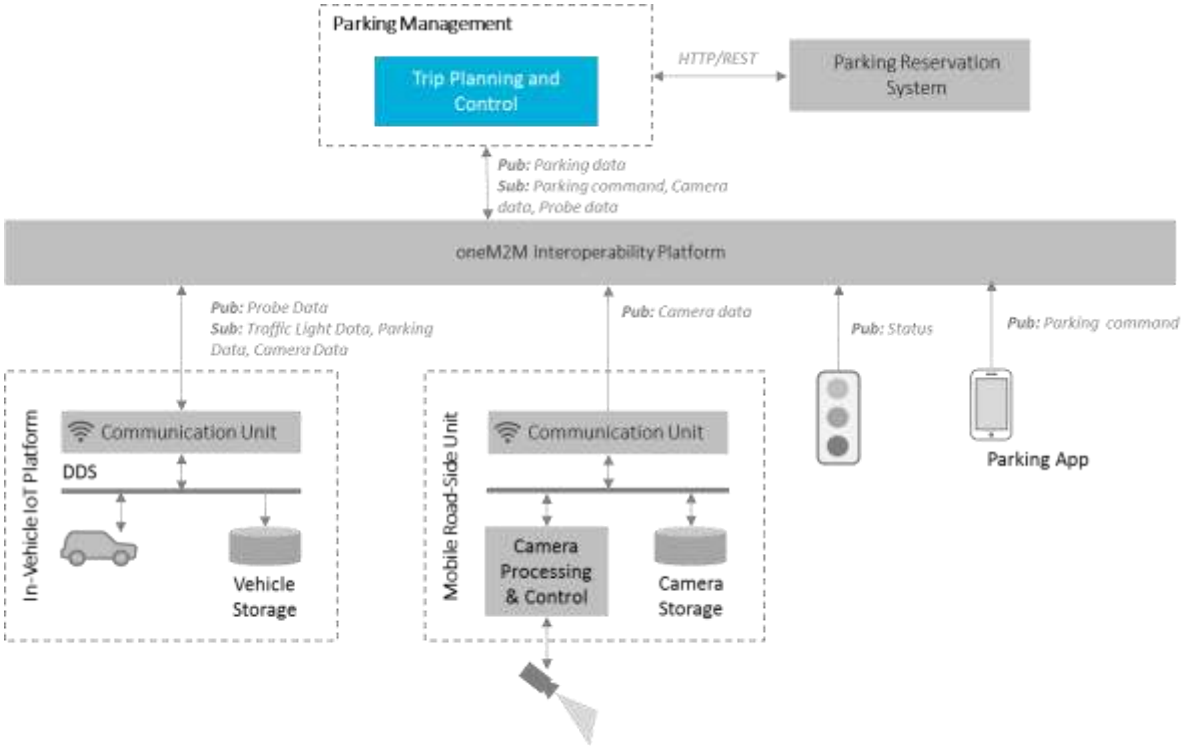

Figure 17. Finnish Pilot Site Overall Architecture

Two use cases have been implemented in the Finnish pilot site: urban driving and valet parking. Figure 17 shows the architecture of the Finnish pilot site and the required components for both use cases under development. The pilot site devices include: IoT connected cameras and connected traffic lights. Parking spots are booked through a parking reservation application, developed by Mattersoft in the Transforming Transport project, using a HTTP/REST protocol.

The traffic camera, vehicles, and the parking management application are connected through an MQTT broker to the open IoT platform.

4.1.2 IoT Ecosystem Implementation in the Finnish Pilot Site

Details of the implemented IoT ecosystem components for each of the Finnish pilot site use cases are provided in the following subsections.

4.1.2.1 Automated Valet Parking Use Case Implementation

The goal of this use case is to provide to the user a service that, through IoT technologies and autonomous car driving functionalities, can automate and improve the way of reserving parking spots and enable the car to automatically park itself once left in a designated drop-off area. The AVP use case in Tampere is achieved through the following components and functionality.

Object Detection with Cameras

As shown in Figure 11, a camera, installed at the mobile road side unit, is used to detect pedestrians or obstacles on the parking site, as well as the occupancy of the parking places. The AI tool YOLOv3 is used for detecting obstacles on the path and on the parking places. Processing is performed locally, and the status of the parking place (parking place occupancy and objects on the path) is published on the IoT platform. Both the in-vehicle platform and the parking management system receive this information. The parking management system uses this information to reserve a free parking spot and to calculate the available route for the vehicle. The in-vehicle IoT platform uses this information to verify that the track is obstacle-free.

Parking Management System

The Parking management system both reserves parking spot and optimises routes to the parking spots. When the vehicle is driving in unmanned mode, the operator in the parking management system acts as a driver, and can force the vehicle to start moving or to come to a stop.

Connected Car with in-Vehicle IoT Platform

A connected car also acts as an IoT device publishing information to the IoT platform. The vehicle receives information from the parking management system related to the parking manoeuvre, and from the camera on objects on the path. The in-vehicle platform is further defined in the AUTOPILOT deliverable D2.1 [D2.1].

4.1.2.2 Urban Driving Use Case Implementation

The goal of this use case is to show how IoT can impact the safety of VRUs and the performance of autonomous driving at signalised intersections.

Pedestrian Detection with Camera (VRU)

The traffic camera, which is installed at the mobile road-side unit, is used to detect VRUs (pedestrians or cyclists) that have a green traffic light at the same time as the turning vehicle. The detection information is processed locally and published to the IoT platform and to the RSU.

The in-vehicle IoT platform in the vehicle receives the information, and the vehicle stops before the pedestrian crossing until the crossing is free.

Traffic Lights

Traffic light status information and time to the next signal phase are received either through V2X or from the server of the traffic light operator (Dynniq). The information is transmitted to the vehicle.

Connected Car with in-Vehicle IoT Platform

The vehicle, as in the AVP use case, acts as an IoT device publishing information to the IoT platform. The vehicle will receive information from the traffic camera and the traffic signals.

4.2 France (Versailles)

4.2.1 Overall IoT Architecture of the French Pilot Site

In Versailles, the French pilot site, three use cases have been implemented through urban driving and mobility focus: car sharing, urban driving and platooning. The IoT architecture for these three

use cases is provided in Figure 18.

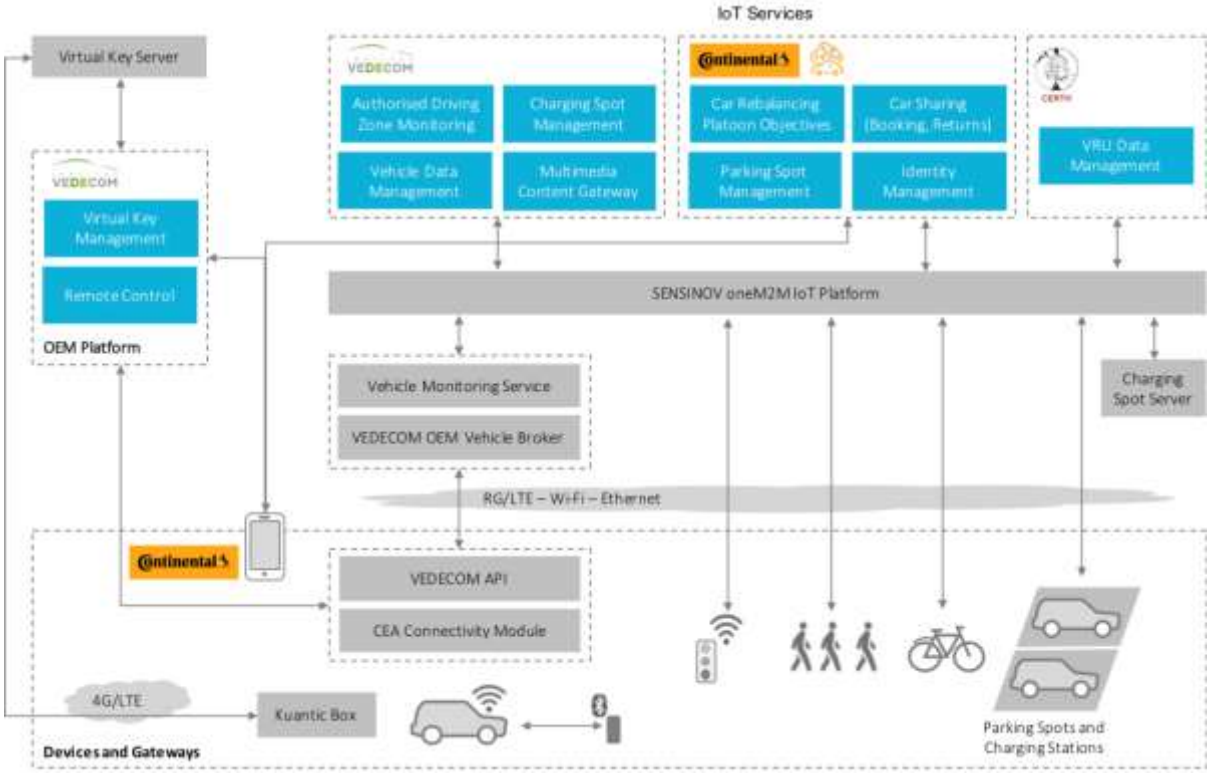

Figure 18. French Pilot Site Overall Architecture

4.2.2 IoT Ecosystem Implementation in the French Pilot Site

The implementation of the IoT ecosystem in the French Pilot Site is summarised in the following sections.

All the data mentioned in the following sections are stored and shared between services on the SENSINOV oneM2M platform. This platform is provided by Sensinov and accessible to all the partners involved in the pilot site.

4.2.2.1 Car sharing

The car sharing booking service is available through a dedicated mobile application. To ensure real time data, the use case requires multiple IoT devices on the infrastructure. Each parking slot used in the experiments is equipped with sensors able to detect whether the slot is vacant or occupied. Parking sensors are assigned to a specific slot and have capacity to determine which car of the fleet is parked, allowing the FMS (Fleet Management System) to manage location of the whole fleet. Moreover, the charging spots provide information about their status, indicating whether an electric vehicle's battery is charging or not.

4.2.2.2 Urban driving

The cars used in the experiments are Renault Twizy and have been equipped with multiple sensors and communication systems. These devices are used by the cars to detect objects in its environment and communicate with other users, other cars or road equipment.

Some sightseeing information about interesting places in the city of Versailles are sent to the driver. To do so, Bluetooth low energy (BLE) beacons are installed on selected POI (Points of Interest). These beacons communicate with the embedded screen of the car to provide audio or video content to the driver.

4.2.2.3 Autonomous driving

During a journey, the driver has a possibility to drive through the gardens of Versailles castle, where, though embedded HMI, they can switch to an autonomous driving mode during a short trip. Once the AD function is activated, the vehicle follows the predefined path.

Vulnerable road users, such as pedestrians and cyclists, carry specific sensors to determine their positions and state. This information is sent directly to the car in order to increase the accuracy of the perception of its environment.

4.2.2.4 Platooning

The aim of this function is to allow continuous usage of the service by moving cars towards a place where a car sharing request is initiated. To rebalance several cars available at the different parking sites of the experiments, platooning is used allowing a single operator to move multiple cars. The FMS pushes information to the service operator when a platoon is needed through a dedicated mobile application. This use case requires different sensors and communication systems in the cars. It also requires communication with road side units to handle road crossings with the platoon. The traffic lights on the route of a platoon are equipped with such units accordingly.

4.3 Italy (Florence-Livorno)

The Italian pilot site is a testing infrastructure encompassing the Florence-Livorno freeway together with road access to the Livorno sea port settlement. The testbed consists of three zones: the Livorno-Florence freeway, the Traffic Control Centre (TCC) located in Empoli, and the port landside just in front of the cruise terminal. The vehicles that were used in the test site are FCA Jeep Renegade with different functions and roles:

- Two vehicles by CRF with automated driving functions, were used to demonstrate the performance of the IoT-ITS ecosystems when the automated driving scenarios beyond SAE 3 levels are running;
- Five service vans by CRF and AVR with advanced V2X communication capabilities, were used for tuning and pre-testing the vehicular systems and services in the IoT enhanced ITS environment.

Two use cases are implemented in Livorno:

- **Highway Pilot:** A cloud service merges the sensor measurements from different IoT devices, such as vehicles and roadside cameras, to locate and characterise road hazards.
- **Urban Driving**: Focuses on the interaction with traffic lights and legacy traffic, on the robustness of the AD functions of the vehicle, safety when dealing with vulnerable road users, and positioning.

In all the use cases the relevant data is sent to the OneM2M platform following the Common IoT Data Model (presented in Section 6). Moreover, the OBUs log all the relevant messages following the InterCor data model as defined in D2.1 - Vehicle IoT Integration Report.

4.3.1 Overall IoT Architecture of the Italian Pilot Site

Figure 19 shows the IoT architecture of the Livorno pilot site for both highway driving and urban driving use cases. Details of the architectural elements for both use cases are provided in the subsequent subsections.

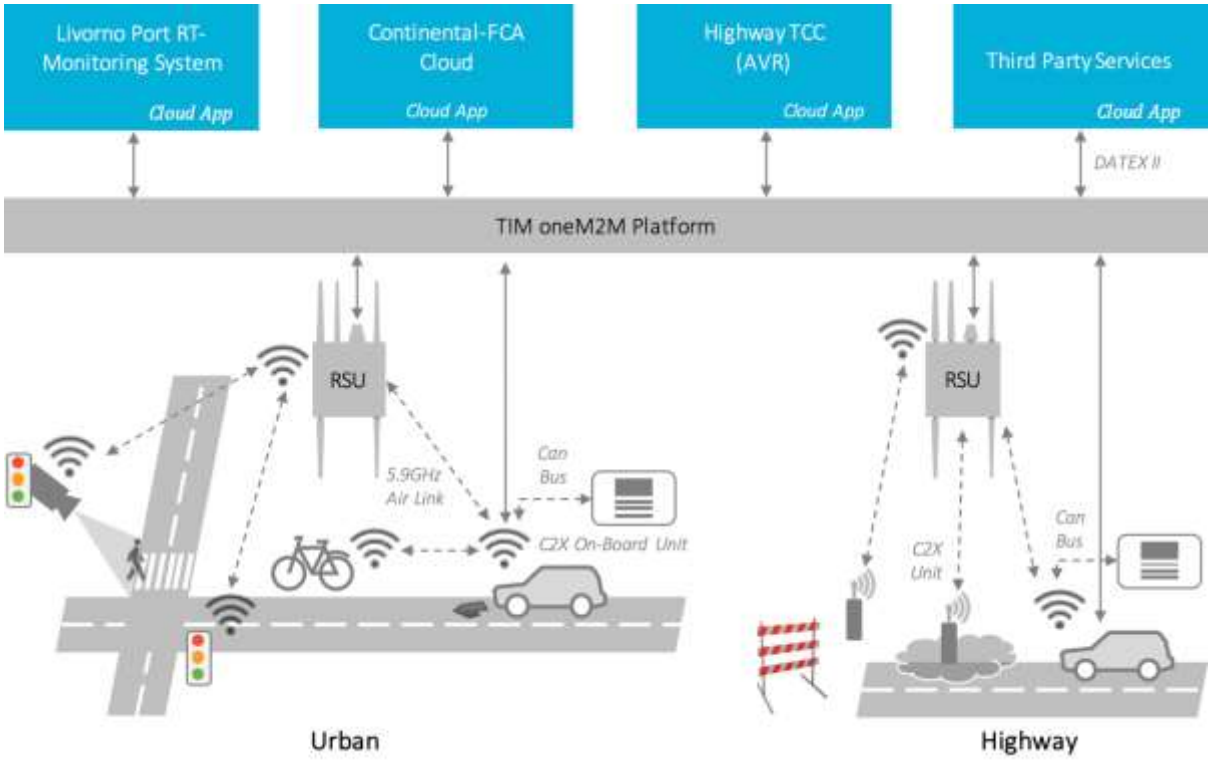

Figure 19. Livorno Pilot Site Overall Architecture

4.3.2 IoT Ecosystem Implementation in the Italian Pilot Site

Details of the implemented IoT ecosystem components for each of the Italian pilot site use cases are provided in the following subsections.

4.3.2.1 Highway Pilot Use Case Implementation (Livorno-Florence)

The scope of these tests involves cars with IoT-enhanced AD functions, driving in a "smart" highway. The cars are Jeep Renegades with on-board equipment, the so-called IoT open vehicular platform, enabling IoT-triggered AD functions: speed adaptation, lane change, lane keeping. Some cars have special sensors also, such as the IoT based pothole detector.

The "smart" highway is a freeway where a pervasive IoT ICT system is deployed based on a network of roadside sensors or other sources, capable of collecting information and making it available to cloud-based applications. Connected cars and the traffic control centre have an important role. For safety reasons, connected cars drive in a convoy, following the AD car.

The goal is to show how the combined use of IoT and C-ITS can mitigate the risk of accident for an AD car when hazards occur on the road. Here, we deal with two types of hazards: (1) puddles and (2) road works.

Puddles

- *Puddle IoT sensors:* IoT sensors placed along the highway continuously monitor the presence of puddles. When a hazard is detected, the sensors send an alert to the road-side unit (RSU)

with detailed information, using IoT standard protocols. The RSU broadcasts the Decentralized Environmental Notification Message (DENM) [DENM14] to vehicles and to the traffic control centre (TCC). The TCC validates the alert and forwards the DENM message to farther away RSUs. At the same time, the TCC feeds the TIM OneM2M compliant platform with alert-related data. Two kinds of sensors are deployed, using different communication technologies: the 6loWPAN puddle sensors send messages to the Road Side ITS-Station using CoAP[3], the NB-IoT puddle sensor sends the message straight to the oneM2M platform using the Long-Time Evolution (LTE) cellular network and REST protocols.

- *Road-Side ITS Station:* Road-Side ITS Station is a programmable gateway with multi access technologies (notably 6LowPAN, ETSI ITS G5, LTE, Eth, etc.). It is a road side unit, compliant with ISO/TC204 WG16 standards [ISOTC204], able to exchange information over different networks, using different protocols, including the IoT ones. The RSU always listens the 6loWPAN sensors and sends the measurement to the OneM2M IoT platform of the PS with a certain frequency. When the hazard occurs the RSU broadcasts a DENM with the lowest quality level of the information (i.e., not yet validated by the TCC) to the approaching vehicles via the IEEE 802.11OCB network, and the oneM2M platform via LTE cellular network. Furthermore, the RSU publishes on the oneM2M platform the cooperative awareness messages (CAM) collected from the vehicles in the Dedicated Short-Range Communications (DSRC) communication range [CAM14].
- *Traffic Control Centre:* The TCC implements a DATEX II[4] node that can supply information from the whole highway network. The TCC is also responsible for managing ITS on the oneM2M platform of the Italian PS. Two kinds of services are provided leveraging the subscription to the oneM2M platform: hazard validation and DENM forwarding. It also publishes to the oneM2M platform the relevant traffic information from the DATEX II node, to be consumed by the highway infotainment service (Fi-PI-LI App). When a hazard (flooding on the road) occurs, the TCC is notified through its subscription to the oneM2M platform, and then, after assessing the severity of the danger, it validates the hazard and broadcasts a DENM with the highest quality level of the information (i.e., validated by the TCC) to the RSUs along the Highway, using the cabled LAN. The TCC subscribes the CAMs of the vehicles published by the RSUs on the oneM2M platform. The information is combined with the Bluetooth and Wi-Fi transit data loggers to perform the travel time analysis and live overview on the TCC video wall. The TCC subscribes the AD car's sensor data on the oneM2M platform to provide ITS services to the users of the highway.
- *Autonomous Driving Car:* The AD car broadcasts CAMs over the IEEE 802.11OCB network; at the same time the AD car publishes data from its sensors to the oneM2M platform. The AD car is approaching the hazard on the road: the in-vehicle application (Connected eHorizon) subscribes the alert from the oneM2M platform. The in-vehicle IoT platform combines the information obtained by the Connected eHorizon (CeH) with that obtained by DENM via the IEEE 802.11.OCB network. It, then, feeds the appropriate autonomous functions, which performs the necessary adaptation of the driving style in a "smooth" way if the message is received well in advance of reaching the hazard. However, if the vehicle is close to the hazard and for some reason (e.g., the warning from the IoT service wasn't received, or the warning was received just by the safety channels of DSRC), then an emergency braking is

[3] Constrained Application Protocol (CoAP): http://coap.technology
[4] DATEX II: http://www.datex2.eu

needed, this event is registered by the in-vehicle application and sent to the OneM2M IoT platform of the pilot site. At the same time, the FCA cloud monitors the performance of the vehicle, checks that the in-vehicle application feeds the appropriate autonomous functions, send notification/warning to the in-vehicle HMI.

- *Connected Cars: C*onnected cars lead and follow the AD car; they continuously broadcast CAMs over the IEEE 802.11OCB network, at the same time, they publish its sensor's data the oneM2M platform. The connected cars are approaching the hazard on the road, so the in-vehicle IoT platform receives the information from both the RSU along the track and the OneM2M IoT Platform. The in-vehicle application pre-alert the driver about the hazard using the information obtained by the OneM2M IoT Platform of the PS and by the DENM. The AD car on-board unit (OBU) can instantiate either smooth IoT-enabled speed adaptation and lane change, or eventually suggest an alternative route (if any) to the driver through HMI.

Roadworks

- A (WSN) sensor node is attached to the road works trailer and announces the presence of roadway works to an RSU. The RSU, in turn, triggers DENM messages, broadcasting information about available lanes, speed limits, geometry, alternative routes, etc. This RSU is located close to the roadway works and it can be a temporary ITS station as well, because if the road works are far away from the permanent RSUs put on gantry, they cannot be reached by the signal transmitted by the trailer. The temporary RSU will use the LTE network to communicate with the TCC, as Ethernet wiring is not available on site.
- The TCC broadcasts the DENM messages to farther away RSUs. At the same time, the TCC feeds the ETSI OneM2M platform with road works related data. This information is consumed by the Connected eHorizon application from CONTINENTAL and transmitted to the FCA cloud as a modified dynamic speed limit based on the generated dynamic event. The FCA cloud immediately notifies the enabled vehicles about the updated information for CeH devices installed on prototypes. The in-vehicle application, then, feeds the appropriate autonomous functions, which perform the necessary adaptation of the driving style, taking into consideration information obtained from DENM messages. A notification/warning through the in-vehicle HMI can then be generated.

4.3.2.2 Urban Driving Use Case Implementation (Livorno-Florence)

This use case demonstrates how IoT may impact the safety of VRUs in an urban-like scenario (instantiated at a harbour settlement) with AD cars, pedestrians at a traffic light crossing, connected bicycles, and a sea port monitoring centre.

Pedestrian Detection with Cameras

Figure 20 shows a smart traffic light detecting a pedestrian or an obstacle on the lane.

Figure 20. Picture of the smart traffic light with RSU and camera for pedestrian detection

The information is processed locally and submitted to the RSU using IoT protocols and to vehicles via standard C-ITS messages. Moreover, a connected traffic light sends information about the time-to-green/red (SPAT/MAP messages [AG15]). The RSU receives the information, fuses the data and sends it by DENM to all the interested actors on the roads.

The OBU of the AD car receives the information and smoothly adapts the speed to the situation, e.g., if a pedestrian is crossing the road when the traffic light is green for the cars, the AD car will behave as if the traffic light is red. Moreover, the detection of VRUs and the traffic light status is displayed on the HMI in the car. The information from RSUs (SPAT, detected pedestrian) and OBUs (DENM) is also sent to the IoT data platform via IoT standard protocols and it can then be processed by the Port Monitoring Centre for real time risk assessment and safety services.

Connected Bicycle

An OBU installed on an ITS G5 bicycle sends standard CAM notifying its presence on the road. The in-vehicle IoT platform of the AD car receives the information and smoothly adapts the speed to the situation. The driver is informed about the proximity of vulnerable users with a notification on the HMI while the AD car algorithms decide what actions to take. Information from the OBUs is also sent to the oneM2M IoT platform over CoAP and/or HTTP.

Potholes Detection

The goal of this use case is to demonstrate how additional IoT sensors placed in the AUTOPILOT prototype can enhance the functions of the car itself. In such a way, the vehicle can be used as an IoT sensor for detecting the surface conditions for both highway and urban scenarios. The Italian Pilot Site has decided to implement the study of a “Virtual Sensor” for pothole detection, using the same approach with three different sensors.

The data of the raw signal accelerations on the 3 axes is collected and analysed using a Nokia 6 smartphone, an inertial 6LoWPAN sensor and the accelerometer sensor of an inertial measurement unit (IMU).

These different wireless vibration sensors are deployed on the connected vehicle, which transmits to the OBU via 6LowPAN/MQTT protocols the occurrences of vibrational shock above a certain level. When a pothole is encountered by the car, the vibration exceeds the threshold, and the IoT vibration sensor sends an alert to the OBU. The latter combines this information with data from the CAN bus (speed, odometer, etc.) and GPS and sends this data to the IoT platform, where they can be fed into the AD car applications.

Additionally, the OBU reports to the OneM2M platform also an idea about the status of the road

surface depending on the data coming from the sensors.

4.3.3 IoT Platform and IoT Devices Integration

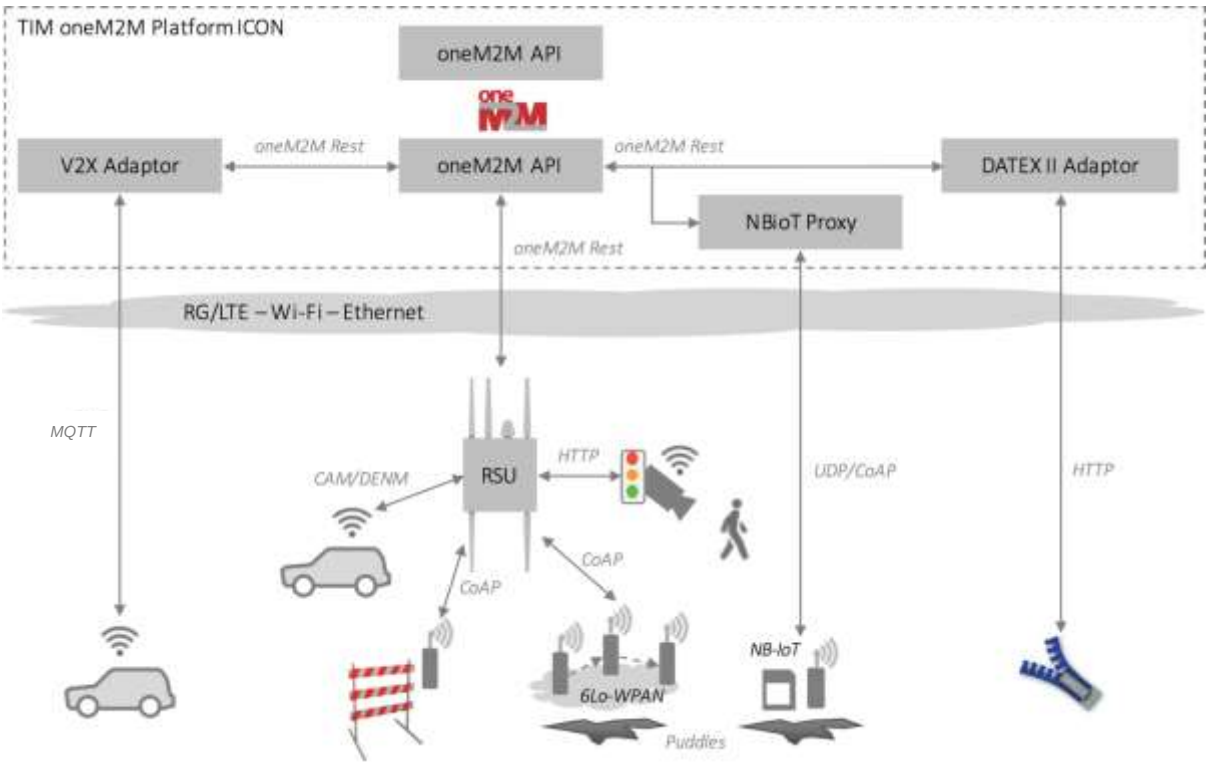

Figure 21. Integration of IoT Devices into the oneM2M IoT Platform in the Italian Pilot Site

In the Italian pilot site, IoT devices are connected to the IoT oneM2M platform through the oneM2M standard, as shown in Figure 21.

Devices (e.g., OEM in vehicle components, inertial sensors, smartphone) in the Italian PS communicate with the oneM2M platform through gateways (On-Board Units) that integrate the IoT information from various sensors.

The in-vehicle IoT platform provided by LINKS Foundation, offers communication interfaces to components from the following partners: TIM (OneM2M Cloud platform), CNIT (accelerometer sensor for the pothole algorithm), CRF and AVR (intra-vehicle sensors).

The OBU can also exchange data with additional IoT devices such as inertial sensors and smartphone motion sensors: these data are interfaced with the in-vehicle IoT platform using CoAP/6LoWPAN or MQTT.

4.4 Netherlands (Brainport)

The Dutch pilot site (Brainport) has two locations:

- Helmond site, covering both highway (A270) and parking area (of the Automotive Campus),
- Technical University Eindhoven (TU/e) campus.

In addition, specific locations are targeted in the neighbourhood where the road surface has some detectable deficiencies.

All use cases have been implemented In Brainport:

- Automated Valet Parking
- Car/Ride Sharing
- Highway Pilot

- Platooning
- Urban driving

The following vehicles have been used in Brainport to support the above use cases:

- TNO/Tass: 3 Toyota Prius,
- TU/e: 1 Toyota Prius and 3 VFLEX,
- VALEO: 1 VW Tiguan,
- TT: 1 mobile mapping van,
- NEVS: 3 electric vehicles (D-class) (planned).

In cooperation with the Innovatieve Verkeerscentrale and TASS (third-party partners), a simplified Traffic Management functionality has been emulated to accommodate the exchange of information about the A270 conditions. This includes information about lane accessibility and maximum allowed speed per lane.

4.4.1 Overall IoT Architecture of the Brainport Pilot Site

Unlike the remaining pilot sites, which have centralised IoT platforms, the Brainport IoT platform is federated and follows the same architecture as the project's IoT platform illustrated in Figure 3.

Five IoT platforms have been deployed in Brainport, and inter-connected exactly in the same way as in in Figure 3:

- FIWARE IoT platform, provided by NEC,
- OceanConnect IoT platform, provided by HUAWEI,
- Watson IoT Platform, provided by IBM,
- oneM2M interoperability platform provided by SENSINOV and instrumented and deployed by TNO.
- MobiMaestro platform provided by Technolution.

4.4.2 IoT Ecosystem Implementation in Brainport

Figure 22 shows the Brainport IoT ecosystem and the device connectivity.

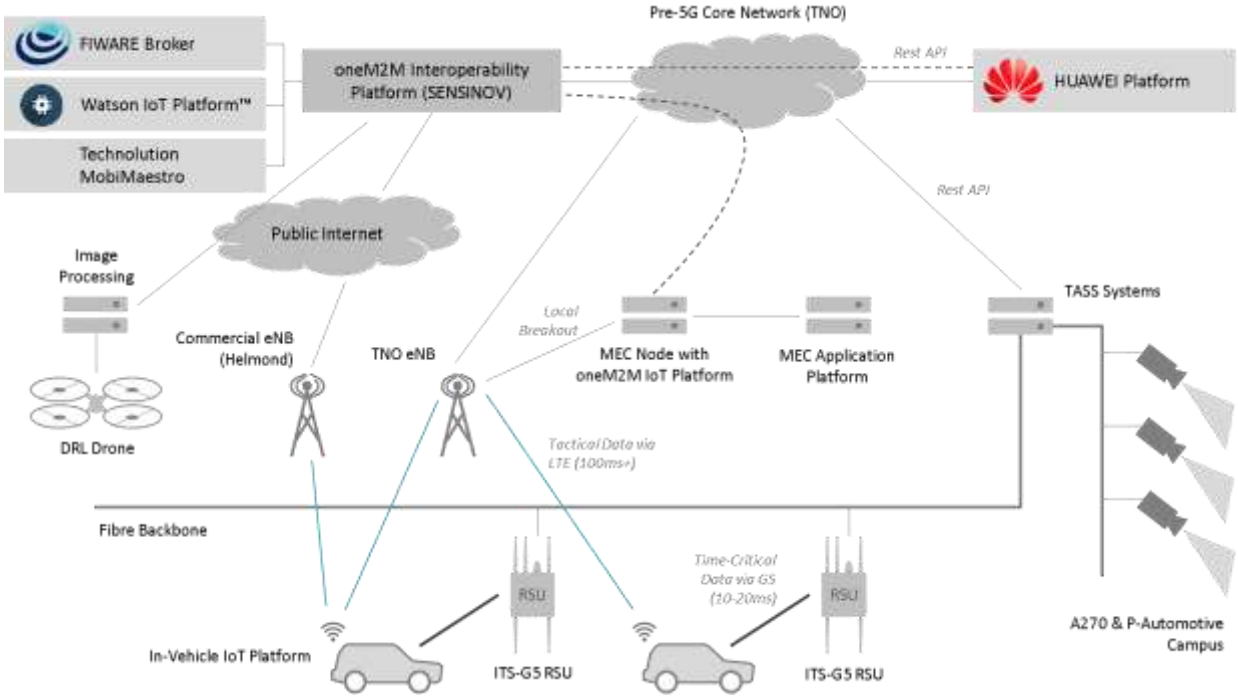

Figure 22. Integration of IoT Devices into the oneM2M IoT Platform in the Dutch Pilot Site

On highway A270 (Helmond – Eindhoven), 5 kilometres of road are covered by ITS-G5 RSUs and TASS cameras. Every 500 meters, there is one RSU and camera. RSU and cameras are connected by a fibre backbone, reaching the TASS offices. All data from ITS-G5 RSUs and cameras are collected and processed by the TASS back-office servers. The processed camera data is pushed to the oneM2M platform with a minimum update rate to make it available to other vehicles and applications using the A270 trajectory. At the Automotive Campus, several cameras are installed for Valet Parking.

All use cases in the Brainport area made use of available commercial LTE services (A270, Automotive Campus, TU/e campus site). The TU/e Campus also offers Wifi services. In addition, TNO has deployed an eNB as part of a pre-5G mobile network for R&D purposes, operating in a licence-free band. This eNB, which offers a coverage area of a few kilometers, supports also NB-IoT, but not yet LTE-V2X (availability is very uncertain at this point). Having our own small scale mobile network allows us to conduct experiments with a Mobile Edge Computing (MEC) node at the road side running a oneM2M middle node platform instantiation which allows the use of more time critical data gathering and distribution. A specific edge application which makes use of the available IoT functionality is a Local Dynamic Map / Shared World Model application which makes IoT sourced information available to participating vehicles which are entering the LDM service area. The application runs on the MEC node and is fed by the central oneM2M platform and can be considered as a bulletin board to which participating vehicles can subscribe.

Vehicles has been equipped with a combination of communication technologies: some has Ultra-Wide Band (UWB), while all have ITS-G5 and LTE interfaces. For validation purposes, several VRU's has been equipped with ITS-G5.
The various communication technologies have been used for different purposes as follows:

- **ITS-G5**: For time-critical communication, only ITS-G5 can support the required latencies. Therefore, ITS-G5 is the protocol used for time-critical communication. Data is communicated using ETSI-G5.
- **LTE**: Vehicles are equipped with LTE modems to connect to commercial LTE services which are widely available. For the sake of the BP pilots one commercial provider has been selected (purchase of SIM cards). TNO's Hi-5 pre 5G platform provides a specific LTE service which can be subscribed to with a regular LTE interface on the vehicle(s). Depending on the type of data (near-time-critical or non-real-time), data is processed on the MEC node or in the cloud.
- **Fixed Connection:** LTE eNB is connected via fixed (fibre) connection to the pre-5G core network, and then to the one2M platform, which is deployed at the premises of TNO The Hague.

4.4.2.1 Automated Valet Parking

In this use case a user drops an AD vehicle at a car park (Automotive Campus terrain) and the vehicle is then able to park itself at a free parking spot. To leave the car park, the user can call the car, which drives itself to the pick-up location. Detection of free parking spots is applied through cameras installed on the premises and equipped with software. This information is used when a vehicle path is being planned.

4.4.2.2 Car/Ride Sharing

The objective was to enable customers to share a fleet of cars (either self-driving or not) to reach their destinations. A car/ride sharing service finds the closest available car and assigns it or dispatches it to the customer. Car/Ride sharing in Brainport is intended as a pure ride sharing, where multiple customers that possibly have different origins and destinations share a part of the ride on a common car. The Car/Ride Sharing use case has been used as a starting used which is linked to other use cases like Platooning and AVP. This was demonstrated at ITS congress in June 2019.

4.4.2.3 Highway Pilot

Information about the state of the road surface (potholes, cracks, patches of ice, etc.) is collected and sent (published) to a central server and is also used to update electronic map data. This data is then made available to other vehicles and applications, for example to adjust their speeds. The use case involves at least two vehicles: a registration vehicle and at least one user vehicle.

4.4.2.4 Platooning

The goal was to show added value of IoT data to platoon formation and subsequent platooning. This goes beyond the direct information exchange between the platooning vehicles, but also involves a so-called platooning service, which runs in the cloud and makes use of available IoT originated data to guide the platooning vehicles on the tactical level, e.g. on speed and lane advice. The platooning use case makes use of the oneM2M IoT platform and MobiMaestro platform for route and interception calculation. Platooning services depicted on Figure 23.

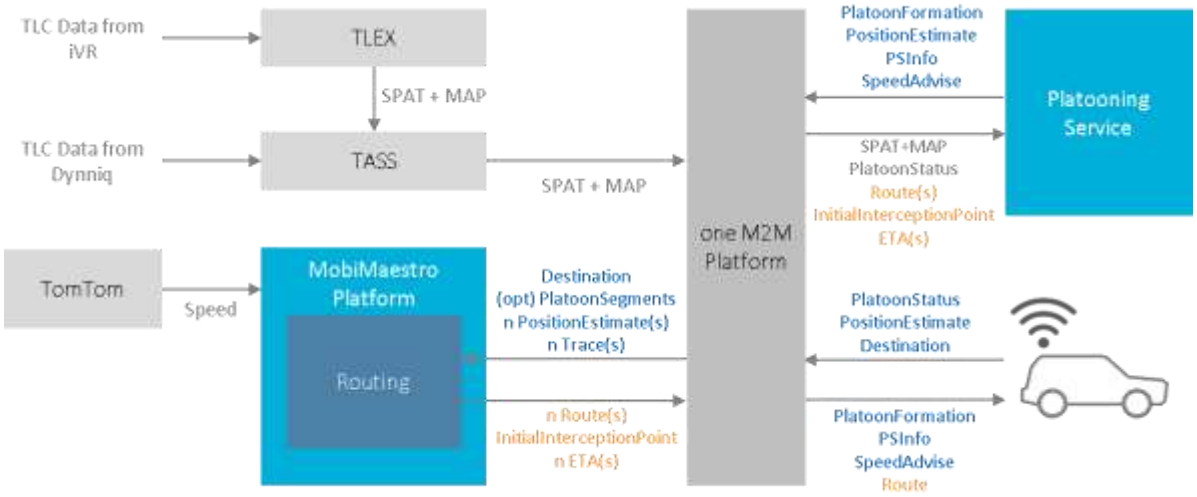

Figure 23. Platooning architecture

Technolution has built a route planner as a new module of the MobiMaestro Network Manager that calculates routes for platooning cars, which create significant overlap on a highway. Using traffic information, computed route represents the shortest path based on the actual traffic conditions and includes average speeds from origin to destination and initial interception point.

Once the route is calculated and the vehicles are driving, the platooning service continuously gives speed advises based on calculated routes, average speeds and updated interception point.

4.4.2.5 Urban Driving

The objective was to demonstrate the added value of IoT information in the process of AD car rebalancing, i.e., automatically moving a fleet of shared AD cars from a place where they are least needed to a place where they are most needed. In this case, we used information about VRUs to schedule car rebalancing and reroute the cars to avoid VRUs.

4.4.3 Driver license virtualization as end-user authentication to Car/ride sharing

Brainport car/ride sharing use case serves as an example service provider for integration of Driver license virtualization service and an authentication service leveraging on virtualized documents. Authentication is based on strong user identification and subsequent issuance of semi-anonymous authentication tokens. The tokens are used for authentication to the car/ride sharing proxy and user information is used to access underlying IBM IoT automotive platform.

Integration has been done using following components:

- **Document virtualization service:** Service that reads user document and issues anonymous PKI credentials with information from the document, in this case age of the document holder and eligibility to drive.
- **Document virtualization SDK:** A library in the car/ride sharing application that is used during document virtualization and authentication.
- **Car/Ride sharing mobile App:** An application that provides car/ride sharing functionality: to select start and end point of a ride integrated with the Document Virtualization SDK.
- **Car/Ride sharing proxy:** A thin backend service that works as a bridge between IBM Watson IoT platform and mobile applications. The main scope was authentication and authorization of the users. The end-user's credentials (virtualized driver license) were translated into access token credentials consumed by the platform.
- **Authentication server based on OpenID Connect protocol:** a standard application for remote verification of virtualized documents.

4.5 South Korea (Daejeon)

Only Urban Driving use case has been implemented at the Korean pilot. There are two locations at the pilot site:

- ETRI site location is convenient and efficient to test Intersection Safety System as of existing V2X communication and ISI service concept developed by ETRI.
- K-City has been built for the automated driving test and provides intersection test facilities including V2X communication network, traffic light and integrated traffic centre.

4.5.1 Overall architecture of the Korean pilot site

Figure 24 shows an overview of the Korean pilot site architecture. This includes:

- An IoT platform that combines data comes from the traffic light and road infrastructure sensors such as a radar or a camera.
- Processed IoT information is sent to OBU for warning generation and IoT Server for other ADs
- Infrastructure for intersection safety information provisioning is set up.

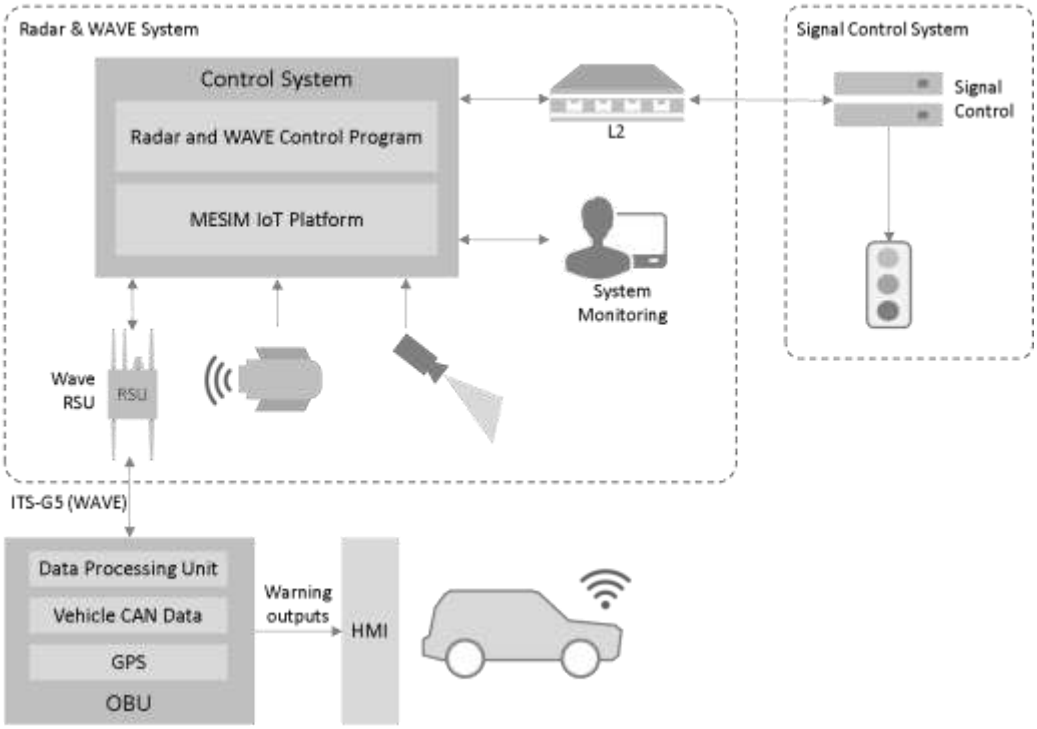

Figure 24. Architecture overview of the Korean pilot site.

4.5.2 IoT Ecosystem Implementation in the Korean pilot site

Details of the implemented IoT ecosystem components for each of the Korean pilot site use cases are provided in the following subsections.

4.5.2.1 Urban Driving

Urban driving use case addresses provision of Intersection Safety Information Service that warns vehicles of pedestrians or vehicles presence in order to avoid accidents. A test road segment is equipped with a radar that detects pedestrians and transmit information to the vehicle's OBU.

Interaction with the devices is following:

- A test driver stops at the intersection and is waiting for traffic lights signal to become green. Simultaneously the road radar scans for moving vehicles on the test road in real time. If vehicles are detected, the information regarding detected vehicles is sent to Intersection Safety System (ISS).
- A traffic light is connected to ISS and traffic light phase information is sent to ISS on a regular basis.
- ISS receives a notification from the road radar that there's another vehicle at the intersection and traffic light phase information. ISS combines two received messages and sends an update to a service terminal via V2I radio communication. The service terminal decides whether the moving vehicle will cross the vehicle's trajectory and potentially may cause vehicle collision. Finally, the service terminal generates a warning signal in case of danger.

Another scenario tests for pedestrian presence at the augmented crosswalks:

- Pedestrian detected by the road radar and their presence combined with the current traffic light state,
- This combined information is transmitted to OBU,
- OBU generates warning timely and displays it on the driver's device.

4.6 Spain (Vigo)

The Spanish pilot site combines the efforts from PSA, CTAG, and Vigo City Council to provide a test environment in the city of Vigo. This test environment supports the use cases of urban driving (UD) and automated valet parking (AVP), which are developed. Both use cases are implemented and tested in different parts of the city. The UD use case is tested in a central street in Vigo, while the AVP is tested in the public indoors parking of the city council.

The Spanish pilot site includes three vehicles, two contributed by PSA, and one contributed by CTAG (PSA branded).

4.6.1 Overall IoT Architecture of the Spanish Pilot Site

The Spanish pilot site ecosystem involves the mentioned vehicles with a oneM2M IoT platform, several IoT devices connected to the IBM Watson IoT platform, a few RSUs working as gateways for the devices, and some IoT applications connected to the IBM Watson IoT Platform.

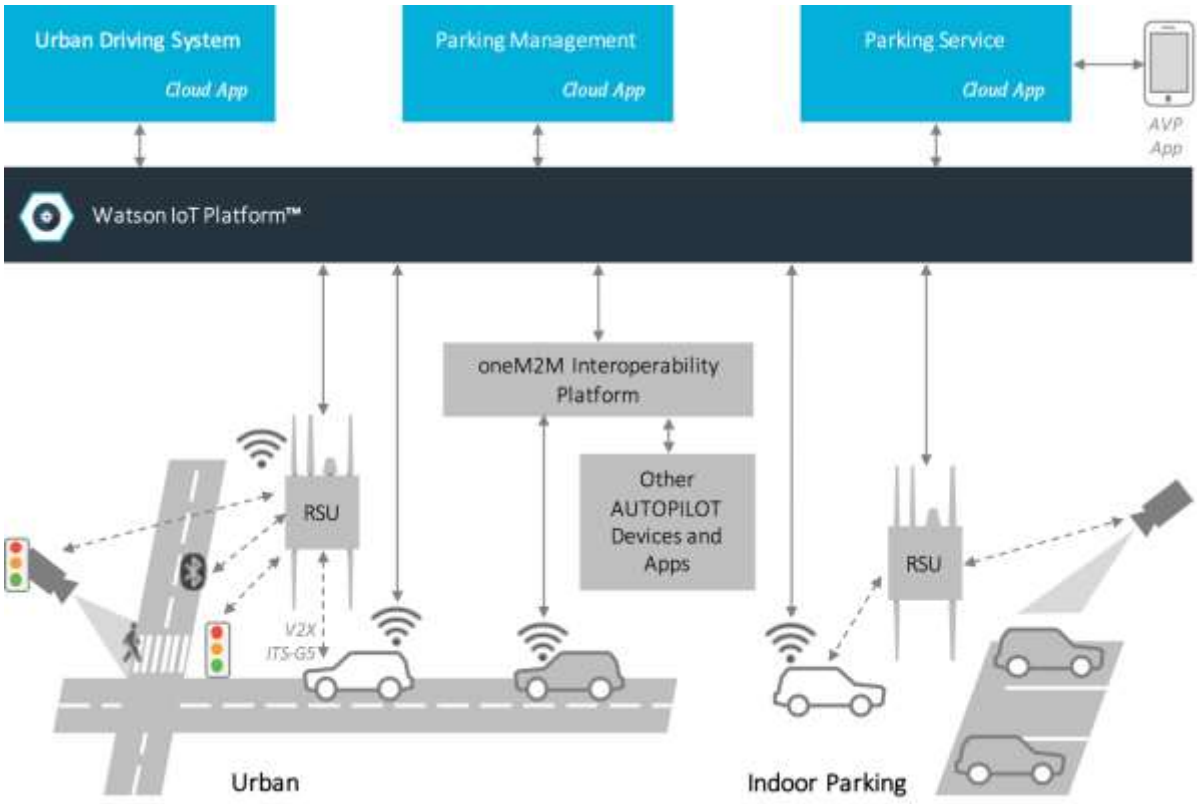

Figure 25. Spanish Pilot Site Overall Architecture

Figure 25 shows the architecture of the Spanish pilot site and the required components for both urban driving and valet parking use cases. The pilot site devices include: IoT connected cameras, connected traffic lights, and a hazard event server that will provide all available information about road conditions as road warnings, traffic jams or accidents.

All these devices are connected to the IBM Watson IoT platform, providing continuous updates of their statuses and making these available to the different applications that might make use of them. Some of these applications (urban driving system, parking management, and parking service) are connected to the IBM Watson IoT platform, while others are connected through the oneM2M connector.

The indoor parking that is used for the AVP use case also provides information to the IoT platform, indicating the available parking spots in real time and instructing the routes for the cars to reach

their assigned parking spaces.

4.6.2 IoT Ecosystem Implementation in the Spanish Pilot Site

Details of the implemented IoT ecosystem components for each of the Spanish pilot site use cases are provided in the following subsections.

4.6.2.1 Automated Valet Parking Use Case Implementation

The goal of this use case is to provide to the user a service that, through IoT technologies and autonomous car driving functionalities, can automate and improve the way of finding available parking spots and enable the car to automatically park itself once left in a designated drop-off area. This is achieved through the following components and functionality.

Pedestrian Detection with Cameras (VRU)

As shown in Figure 25, a camera is used to detect pedestrians or obstacles on the parking site. The information is processed locally and published to the IoT platform and to the RSU. The in-vehicle IoT platform receives this information and smoothly adapts the speed to the situation or stops completely to avoid any collision.

Parking Management System

As mentioned before, the AVP use case also provides information about available parking spots and allow these to be booked in advance. This is performed by the parking management system, which publishes to the IBM IoT platform the status of the parking site, including available spots. Using the DATEX II schema as a reference, all this information can be provided and linked, from the different parking sites to the specific parking spots or groups of parking spots. These data can then be processed and managed by the parking management system, which enables the booking and management of the Spanish parking site.

Android AVP Application

For this AVP use case, the Spanish pilot site has an Android application that, using the IoT messaging protocols and the provided APIs of the implemented services, is the interface that the user uses to perform the possible actions for AVP.

Connected Car with in-Vehicle IoT Platform

A connected car also acts as an IoT device publishing information to the IoT platform. Moreover, the in-vehicle IoT platform has the required services to translate the IoT messages that command the pickup and drop-off actions of the AVP use case into the needed AD messages to start the manoeuvres.

The in-vehicle platform is defined in the AUTOPILOT deliverable D2.1.

4.6.2.2 Urban Driving Use Case Implementation

The goal of this use case is to show how IoT can impact the safety of VRUs and the performance of autonomous driving in an urban-like scenario with AD cars, pedestrians at a traffic light crossing, hazards and connected traffic lights.

Pedestrian Detection with Camera (VRU)

As shown in Figure 25, cameras are used to detect pedestrian or obstacles on the lane. The detection information is processed locally and published to the IoT platform and to the RSU.
The in-vehicle IoT platform or the OBU of the AD car receives the information and smoothly adapts the speed to the situation or stops completely to avoid any collision.

Traffic Light Events

Connected traffic lights send information about the time-to-green/red to the IoT platform. From the IoT platform, the vehicle can receive this information and adjust its speed depending on the traffic light status, stopping when lights are red and moving when they are green. Moreover, the car can display the traffic light status in its HMI according to the received IoT messages from the traffic lights.

Hazard Events

A traffic control centre, connected to the IoT platform, provides information about different hazard events, such as road works, accidents, etc. The traffic control centre sends every registered event to the IoT platform, allowing the AD vehicles to access this information and act accordingly, by lowering their speeds.

Connected Car with in-Vehicle IoT Platform

Similarly, to the AVP use case, the car is always publishing its own status as an IoT device.

5 Interoperability

As already mentioned in Chapter 3, the AUTOPILOT IoT architecture was designed as a federation of IoT platforms, allowing it to be **open** and **flexible**. Developers may plug their own (proprietary) IoT platforms or devices in the architecture and exchange data with existing IoT platforms and devices. As each IoT platform provides a different set of services (features) and may expose a different interface and use a different data exchange protocol, an effort is needed to achieve interoperability while allowing for openness and flexibility.

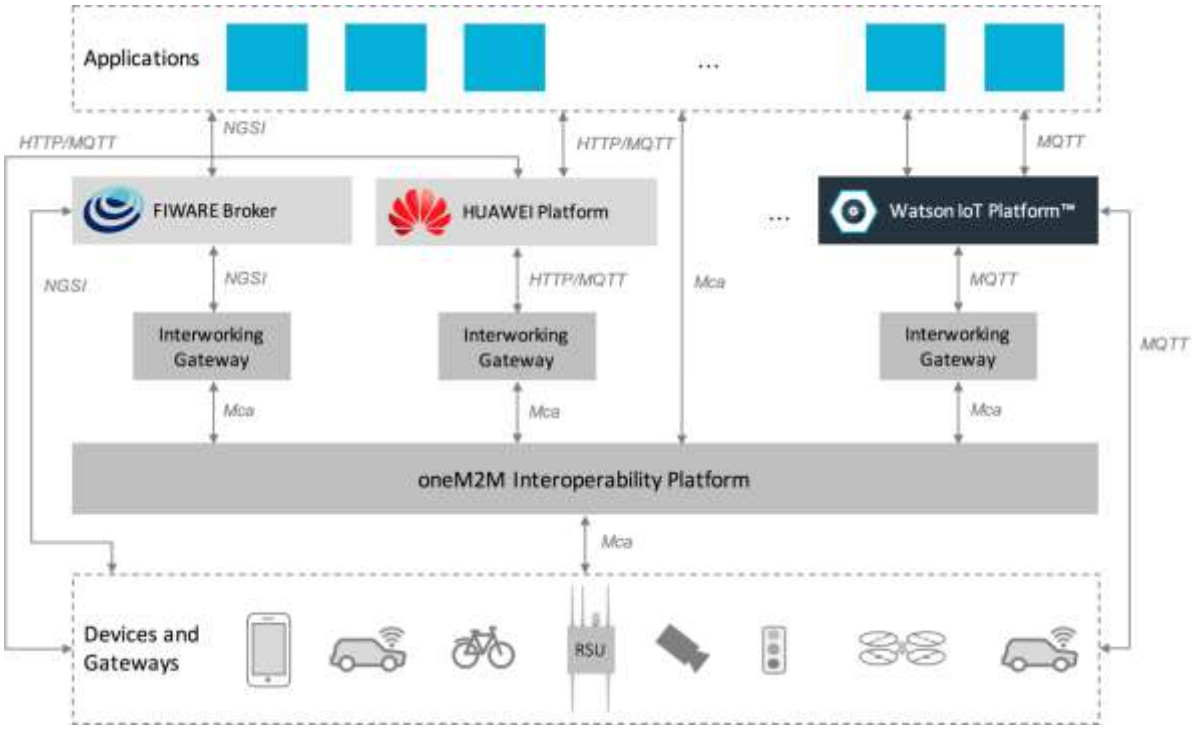

Figure 26. Autopilot Federated IoT Architecture (Copied from Chapter 3 for convenience).

Interoperability in AUTOPILOT is achieved based on the following three principles:

- **oneM2M Interoperability Platform and Interworking Gateways**: As shown in Figure 26 (copied from Chapter 3), proprietary IoT platforms are interconnected through interworking gateways and the oneM2M interoperability platform.
- **Standardised IoT Data Models**: IoT data requiring to be exchanged across the IoT platforms

are standardized:

- IoT Data Models specify the syntax, i.e. how the IoT data has to be represented.
- IoT Data Models also require agreement on the semantics of the data, i.e. what the data means – this may be defined in a written specification that developers have to adhere to – or in form of an explicit ontology that is available in electronic form. E.g. the FIWARE NGSI-LD Broker refers to concepts identified by unique URIs, which may be defined in an ontology.

Each of the above interoperability principles is discussed in the following sections.

5.1 oneM2M Interoperability Platform and Interworking Gateways

An interworking gateway is considered as a oneM2M wrapper belonging to a proprietary IoT platform, which enables to expose a oneM2M interface and to be connected to the oneM2M interoperability platform. We refer to the gateway between Watson IoT and oneM2M platforms as "Watson-oneM2M Interworking Proxy and to the gateway between FIWARE and oneM2M as "Semantic Mediation Gateway (SMG)" or "Morphing Mediation Gateway (MMG)". MMG is a more dynamic and advanced version of SMG.

The oneM2M platform serves as the bridge for interoperability, allowing data to flow from one IoT platform to another in both directions. Using this architecture, an IoT platform may push data to other IoT platforms and receive data from them.

Mapping between the internal data representation of an IoT platform and the oneM2M message contents are specified in the interworking gateways. These gateways also act as filters allowing only selected data to be exchanged.

In this architecture, data providers or consumers, such as applications, may use any of the available IoT platforms according to their requirements. For instance, a data provider may publish data to Watson IoT platform and this data can be shared with FIWARE through the oneM2M platform, so that an application developer who uses FIWARE can access it through the FIWARE platform.

This approach offers flexibility to the pilot sites and application developers. However, to enable this oneM2M interoperability platform data flow, data providers and consumers need to exchange data using standard data models and vocabularies as explained in the following sections.

5.2 Standardised Data Models

oneM2M provides a standard protocol for exchanging IoT messages, but it does not specify the content of the messages as this is domain specific. To achieve interoperability in AUTOPILOT, we standardised the contents of the oneM2M messages exchanged between the IoT platforms, devices, applications, and vehicles, through the oneM2M interoperability platform. A Data Modelling Activity Group (DMAG) was created in AUTOPILOT for this purpose.

The scope of the data model standardisation activity in AUTOPILOT covers the IoT messages and data fields required to implement the project's use cases uniformly across the pilot sites. On the one hand this includes the syntactic structure of the messages, e.g. if a certain field expects a number or a string. On the other hand, the semantics is specified, e.g. that the number expected by a certain field is a speed in kilometres per hour. All standardized data models are described in a written specification. Some may have an alternative representation, e.g. according to the NGSI-LD model for use with the FIWARE platform with a corresponding ontology representation.

Standardised data models allow AD vehicles to access the same types of data regardless of their locations (pilot sites), and to be able to process the data and work with it. For instance, a message notifying AD vehicle about a hazard on the road, or instructing them to avoid a given road lane, should be the same in all pilot sites, allowing vehicles to consume these messages and react to them

correctly as they are moving from one place (e.g., pilot site) to another.

Details of the data model standardisation work are provided in chapter 6.

6 Common IoT Data Model

This chapter provides an overview of the IoT data model standardisation work carried out by the AUTOPILOT Data Modelling Activity Group (DMAG).

For the final version of the AUTOPILOT data models, please refer to the dedicate GitLab repository at https://gitlab.com/autopilot/iot-data-model and wiki at https://gitlab.com/autopilot/iot-data-model/wikis/home.

6.1 Scope of the Work

It is important to note that it is not the intention of the DMAG to standardise all the data across all the use cases and pilot sites. Rather, the scope of the DMAG work covers only the IoT messages used for exchanging **information** or **instructions** between IoT devices, services, and the AD vehicles. This includes, for example, messages notifying AD vehicles about the presence of a hazard, or object, or instructions for AD vehicle to avoid a given road lane. Raw sensor data (example LiDAR, camera images, etc.) and service internal data models (e.g., parking data, user accounts, etc.) are beyond the scope of the data modelling activity.

Work of the DMAG is based on reusing, and possibly extending, existing standards rather than creating new models.

6.2 Use Case IoT Data Requirements

Use case IoT data models initially were gathered by NEC from the project partners. This allowed the DMAG to examine the requirements pertaining to the IoT data model and to select the messages to standardise. The final list of IoT messages under standardisation by the DMAG is provided in Table 11.

Table 11. AUTOPILOT Use Case IoT Messages Selected for Standardisation.

Message Type	Pertaining Use Cases
Vehicle Probe Data (GPS position, speed, status)	Car/ride sharing, AVP
Notifications about detected objects	AVP, highway pilot, urban pilot, platooning, car/ride sharing
Notifications about VRUs	AVP, highway pilot, urban pilot, platooning, car/ride sharing
Notifications about hazards and obstacles	AVP, highway pilot, urban pilot, platooning, car/ride sharing
Notifications about traffic conditions	Highway pilot, urban pilot, platooning, car/ride sharing
Notifications about environmental conditions	Highway pilot, urban pilot, platooning, car/ride sharing
Traffic light states and time to red/green	Highway pilot, urban driving, platooning
Notifications about parking space availabilities	AVP, parking, car/ride sharing
Notifications about charging spot availabilities	AVP, parking, car/ride sharing
Routing instructions	Highway pilot, urban driving, AVP, car/ride sharing
Platoon instructions	Platooning

As can be seen in Table 11, data sources (sensors) are not mentioned here. In fact, we were aiming to standardise the IoT messages regardless of the originating sensor or application.

6.3 Common IoT Data Model

Following the initial IoT data requirement identification phase, the DMAG has elaborated a project standard for the IoT messages. The final version of the data models and how they are used in AUTOPILOT are available from the GitLab repository at https://gitlab.com/autopilot/iot-data-model.

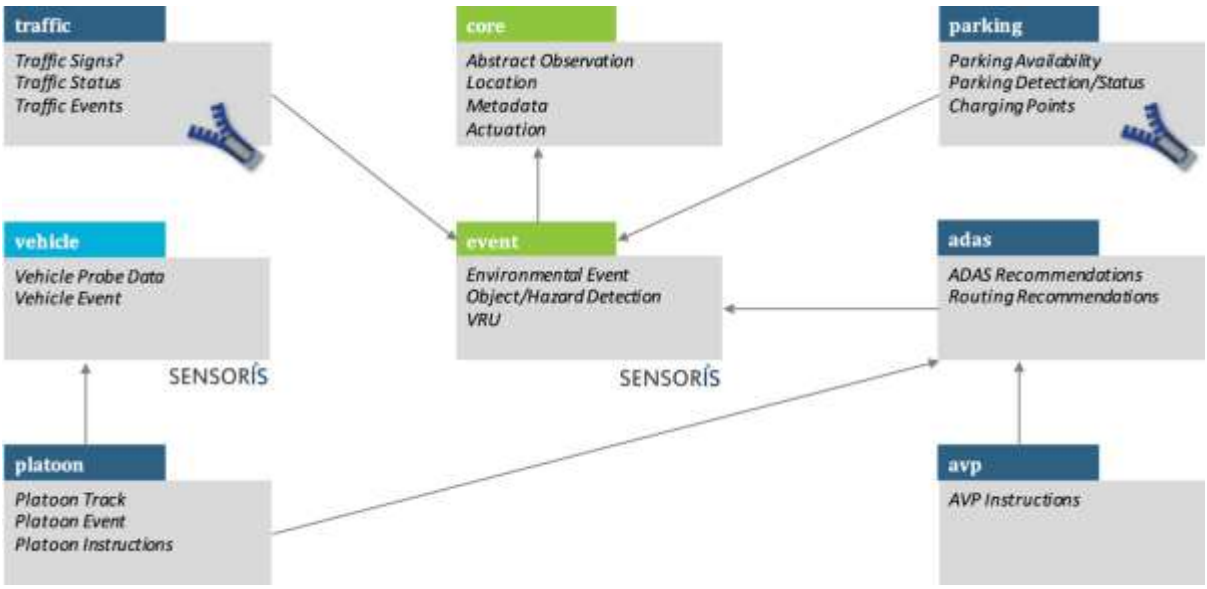

Figure 27. Overview of the AUTOPILOT IoT Data Model Packages

The AUTOPILOT common IoT data model is split into several packages, based on different standards (e.g., SENSORIS, DATEX II). Figure 27 shows an overview of the current packages being designed and their dependencies. Packages leaders are listed in Table 12.

Table 12. IoT Data Model Package Lead Partners

Package	Lead Partner
core and event	NEC, Gurkan Solmaz <gurkan.solmaz@neclab.eu>
vehicle	IBM Ireland, Anton Dekusar <adekusar@ie.ibm.com>
road side equipment	CNIT, Mariano Falcitelli <mariano.falcitelli@cnit.it>
parking	CTAG, Silvia Alén <silvia.alen@ctag.com>
traffic	CNIT, Mariano Falcitelli <mariano.falcitelli@cnit.it>
avp	DLR, Louis Touko Tcheumadjeu <louis.toukotcheumadjeu@dlr.de>
platoon	TNO, Jacco van de Sluis <jacco.vandesluis@tno.nl>

An overview of the packages that were developed is provided in the following subsections.

6.3.1 Event package

The Event package covers messages sent primarily to AD vehicles about various road and traffic situations pertaining to the AUTOPILOT use cases, such as accidents, road works, weather conditions, object detection, etc.

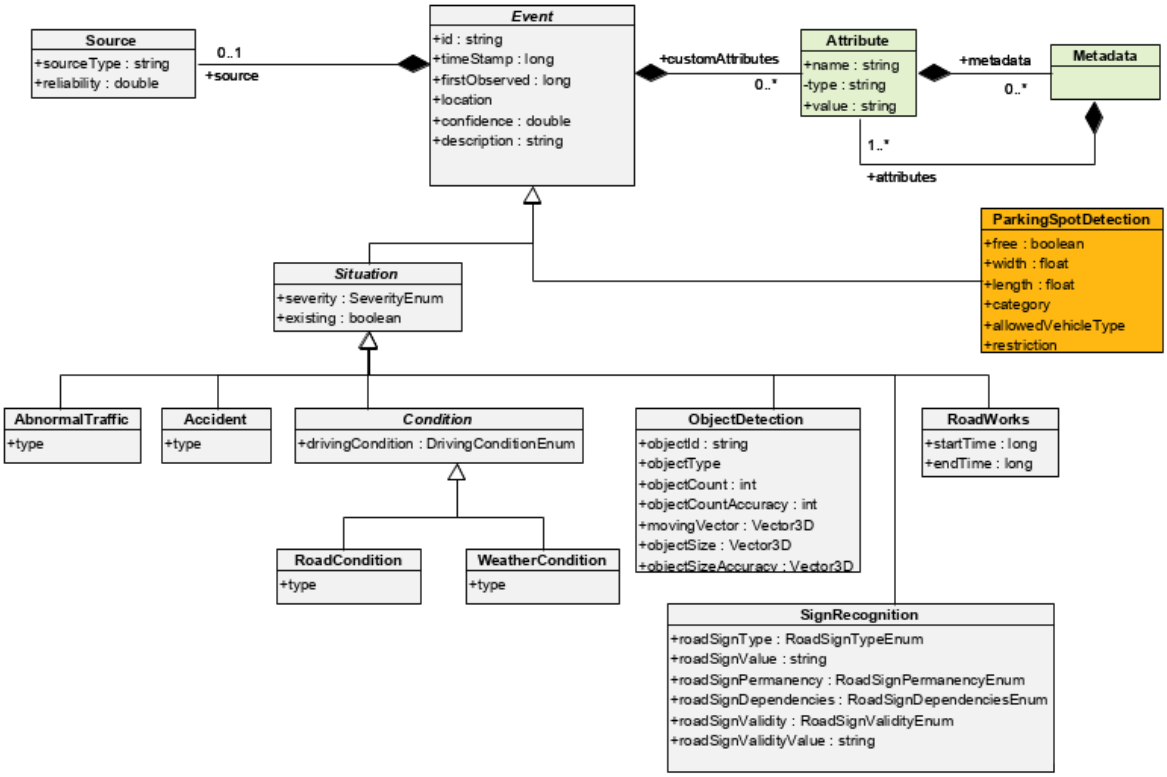

Figure 28. Overview of the Event Classes

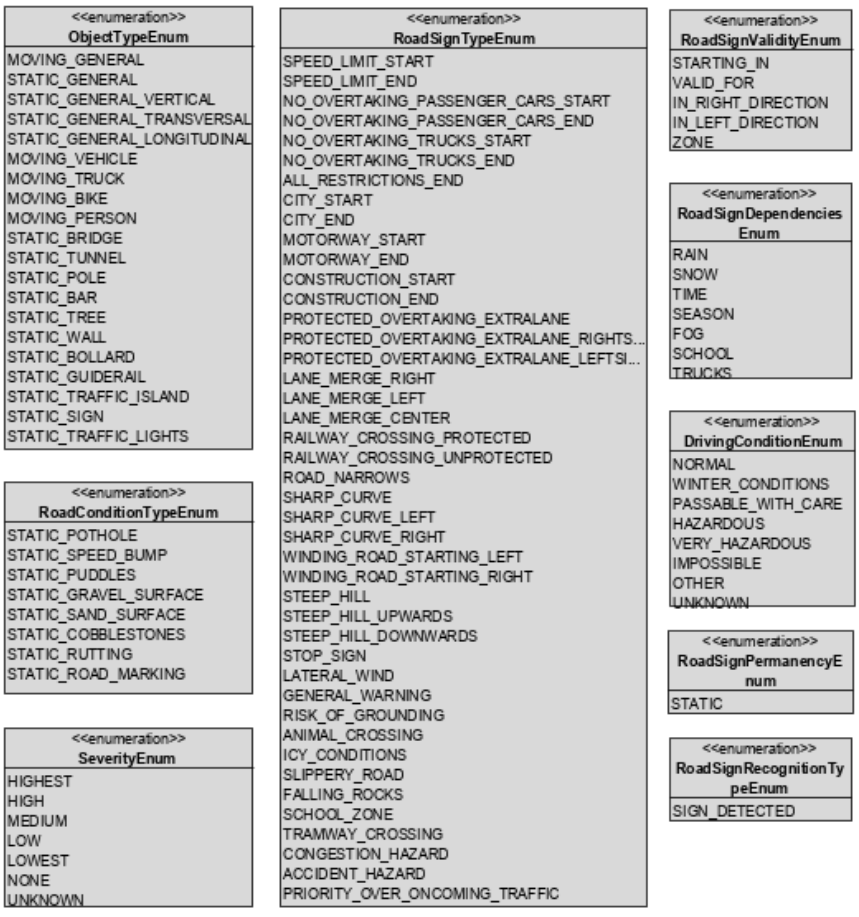

Figure 29. Overview of the Event Enumerations

The event package classes and enumerations are illustrated in Figure 28 and Figure 29 respectively. As shown in Figure 28, different types of events exist:

- The abstract class Situation represents events related to the road. These include:
  - *AbnormalTraffic*: traffic jams
  - *Accident:* defines an accident occurred on a specified road at certain timestamp.
  - *Condition*: abstract class representing environmental or infrastructure conditions:
    - *RoadCondition*: potholes, hazardous substance spills, etc.
    - *WeatherCondition*: fog, ice on road, etc.
  - *ObjectDetection*: detection of an object on, or close to, the road, such as animal, person, crowd, obstacle, etc.
  - *RoadWorks:* defines road works with start and end time

6.3.1.1 Crowd Modelling

Crowd behaviour is modelled as events of ObjectTypeEnum MOVING_GENERAL. While some key aspects like location, observation timestamp and object count are directly mapped as event attributes, the more detailed aspects of the NGSI modelling are modelled as custom attributes, enabling the translation between the Autopilot instance that can be stored in oneM2M and the NGSI modelling stored in a FIWARE Broker (IoT Broker or Scorpio).

The semantics of the FIWARE crowd model is modelled according to an ontology whose structure is visualized in Figure 30.

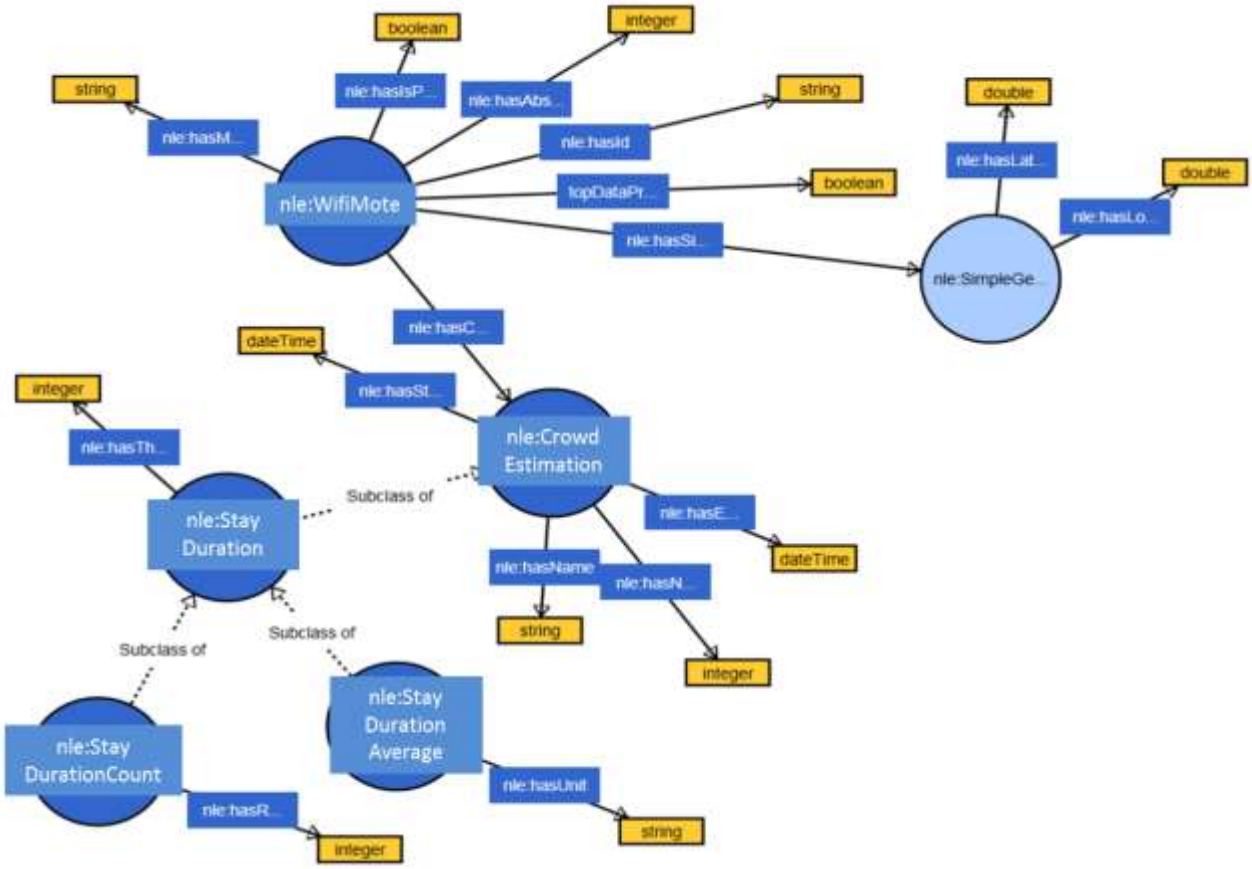

Figure 30. Crowd Mobility Ontology

The core entities being modelled are the WifiMote, which does the measurments, and the CrowdEstimation, which provides the actual information about crowd. The latter has a subclass StayDuration, which in turn has subclasses DurationCount, which models the number of people detected which have stayed in the area for a minimum duration, and DurationAverage, which

provides the average time people stay in the detection area. The instances of each of these concepts have further attributes, e.g. an identifier, a unit (e.g. for the stay duration) or a name (e.g. for crowd estimation).

6.3.2 Vehicle Package

The vehicle package covers the messages sent from the AD vehicles to the IoT platform. This is relevant to the car/ride sharing, AVP, and platooning use cases, where vehicles receiving a service need to be tracked. Vehicle data are consumed by the relevant car/ride sharing, AVP or platooning service, but not shared with other services.

The vehicle package is based on the SENSORIS data model [HERE15]. An overview of the elements of a SENSORIS message is provided in Figure 31. Greyed elements will not be supported in AUTOPILOT.

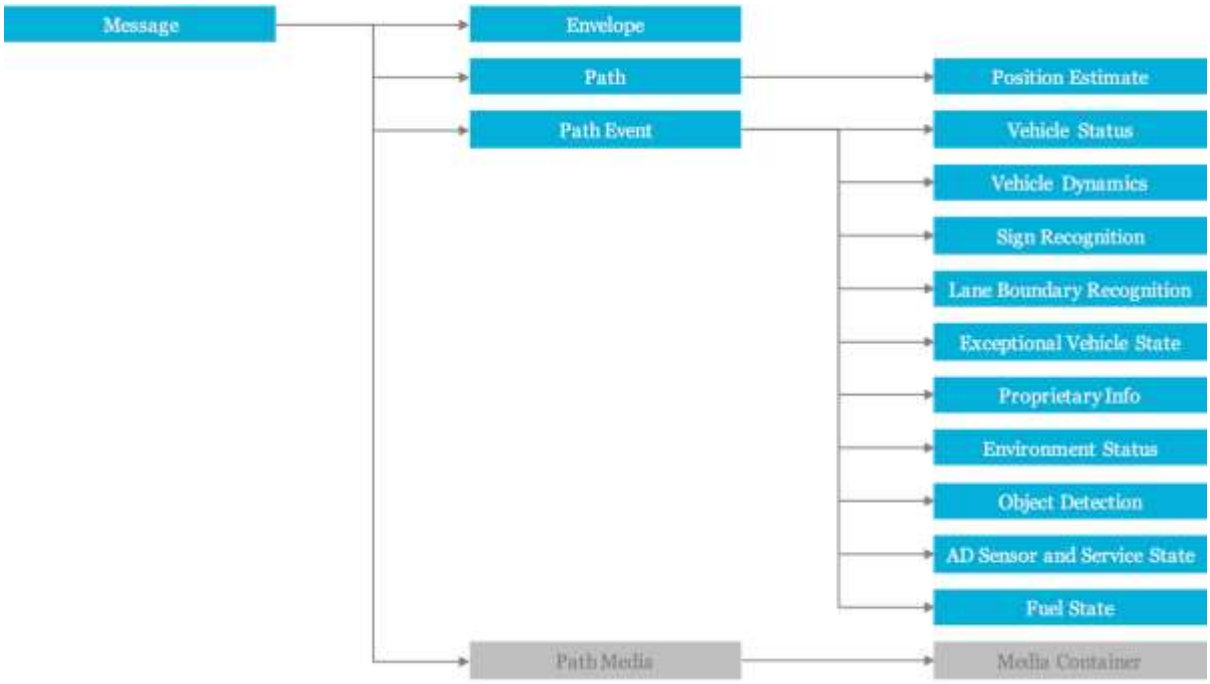

Figure 31. Overview of the SENSORIS Message Elements

In addition to vehicle GPS data and statuses, SENSORIS messages include events detected by vehicles along their paths, e.g., object detection, environment status, lane boundary detection, etc. SENSORIS event information is supported in AUTOPILOT, but is considered as raw data. This means that SENSORIS event messages submitted by vehicles are not consumed directly by other vehicles. Rather they are intercepted by event processing services that retrieve the event detection sections from the vehicle messages, analyse them (e.g., for quality check) and republish them as standardised event messages (as specified in the package event).

6.3.3 Traffic Package

The IoT devices, notably sensors and gateways, deployed along the roads support both the Traffic Control Centers to manage the traffic operation and the IoT enabled AD cars to perform automated functions in potentially dangerous situations. In the AUTOPILOT project, new devices of this kind have been developed, such as water level sensors, IoT-G5 ITS stations, smart traffic light, etc. (see D2.4). The Traffic Package concerns the information generated and consumed from those devices, harmonized with the legacy systems currently used by the TCCs and similar service providers for the exchange of traffic information and traffic data, for example DATEX II.

Well assessed data models for the devices already standardized in the scope of the Intelligent

Transport Systems have been adopted whenever it was possible. On the other hand, for the novel IoT devices such as the water level sensors, and the jaywalking detectors, ad hoc data models have been specified.

Traffic package has four different sub-packages. An example of the resource tree of the Application Entity for the Livorno PS on the oneM2M platform is shown on Figure 32.

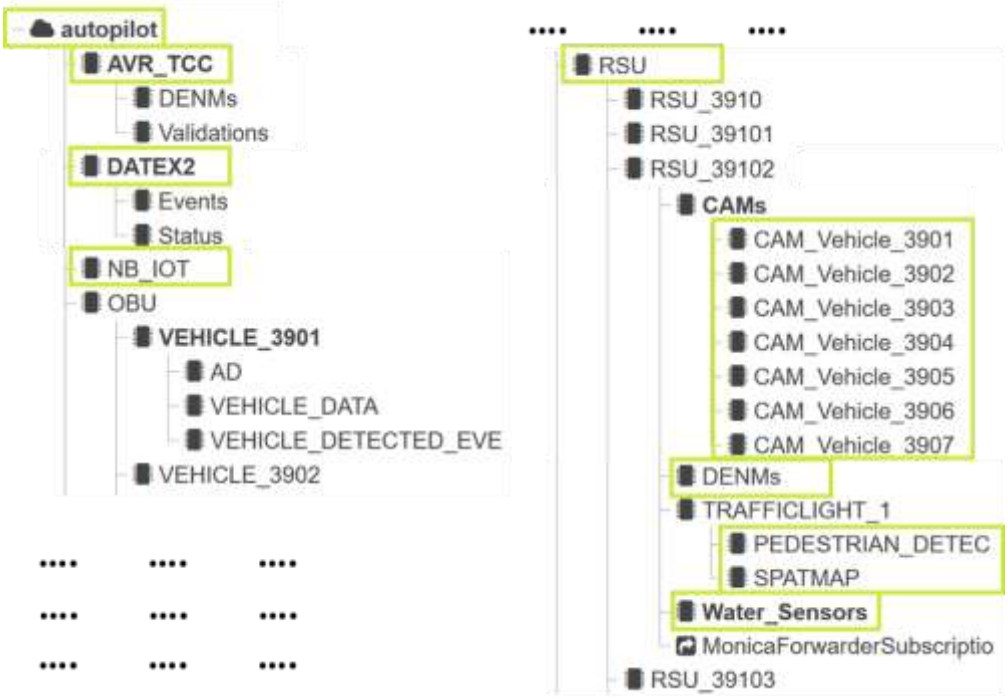

Figure 32. An example of the resource tree of an AE.

- Traffic Control Center sub-package
  - DENMs generated or validated by the TCC
  - Content of validations
- DATEX II sub-package
  - Events (pushed at occurrence)
  - Status (periodically updating)
- NB-IoT sub-package
  - Water level sensors
- RSU sub-package
  - JSON format of ETSI ITS-G5 facilities (CAM, DENM, SPAT, MAP, …)
  - Gathering CAM from vehicles
  - Content and status changes of DENMs
  - Status of the 6LoWPAN Water Level sensors
  - Status and events of the “Smart Traffic Light”

Traffic Control Center sub-package

The Traffic Control Center sub-package holds the messages and the information managed by the Traffic Control Center when it performs the function of a centralised ITS station. The DENM messages for the RSU along the highway are published there according to the ETSI C-ITS standard (see for example Figure 33), as well as the validation for the DENMs generated from the RSUs detecting hazards on the highway by means of their own sensors.

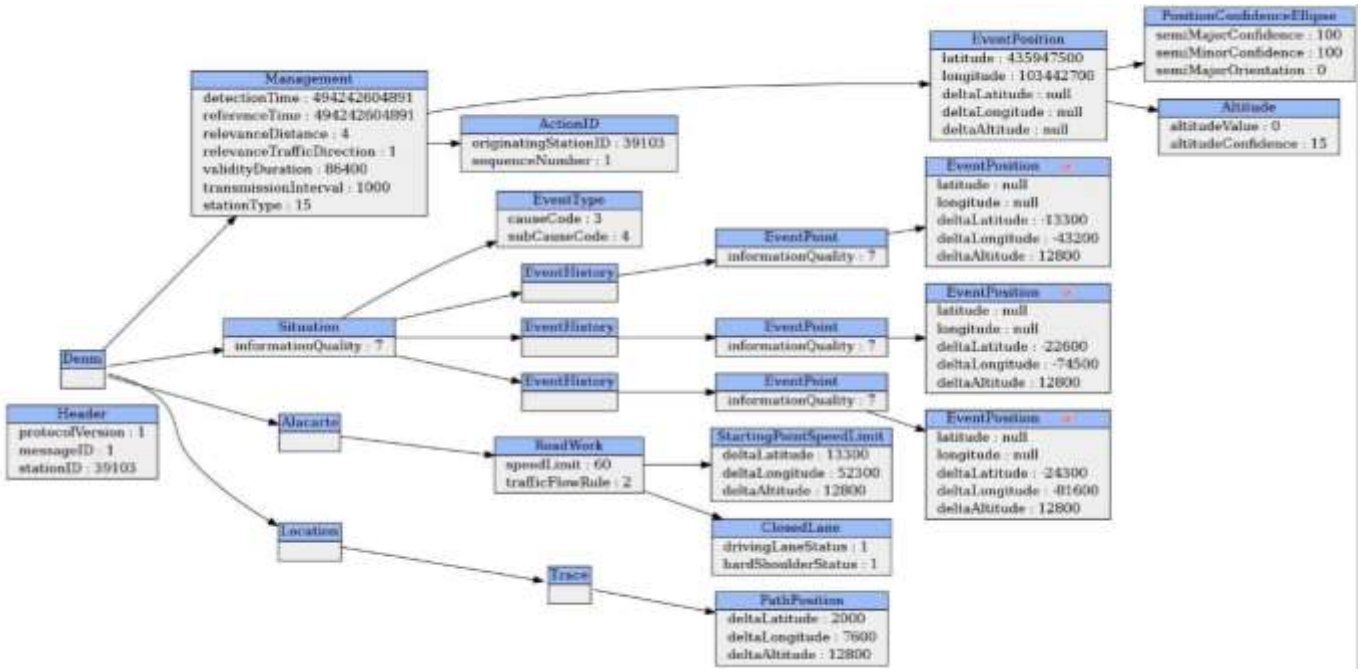

Figure 33. Overview of the ETSI C-ITS DENM Message Elements

DATEX II sub-package

The DATEX II sub-package shows the traffic events of the highway according to the DATEX II node deployed in the TCC. The traffic events are stored in two containers: “Events” receives via real time Push notification the last event occurred on the highway (e.g. roadworks, accidents, abnormal traffic, etc.); “Status” takes a snapshot of the current situation every hour, notably each content instance encompass all the active events occurring in the highway at question time. The data model of the messages is exactly DATEX II, but the format is JSON instead of XML; as an example, in Figure 34 the “Maintenance Works” messages are shown.

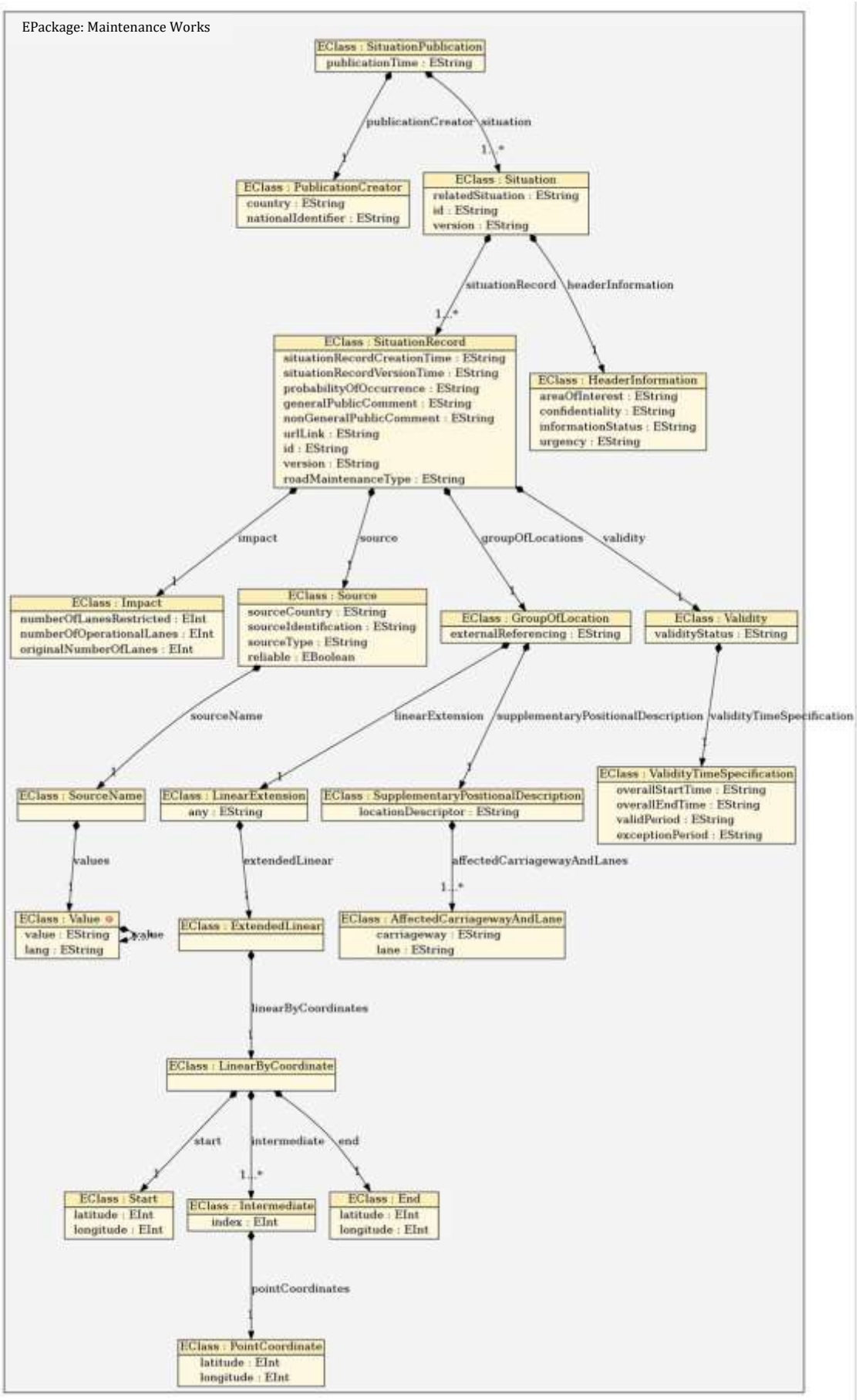

Figure 34. Overview of the DATEX II "Maintenance Works" Message Elements

NB-IoT sub-package

The NB-IoT sub-package collects the data published by the water level sensor specific to the Livorno pilot site. In this case, the structure is very essential: the keys are just “state” (1 or 0), “timestamp”, “position”.

RSU sub-package

The RSU sub-package encompasses the data produced by the IoT enhanced ITS Station placed at the road side. The RSU receives and transmits data communicating with vehicles, sensors, traffic light and TCC. The CAM messages received from the vehicles via the ETSI ITS G5 radio are published in JSON format to the oneM2M sub-container, as showed in Figure 35.

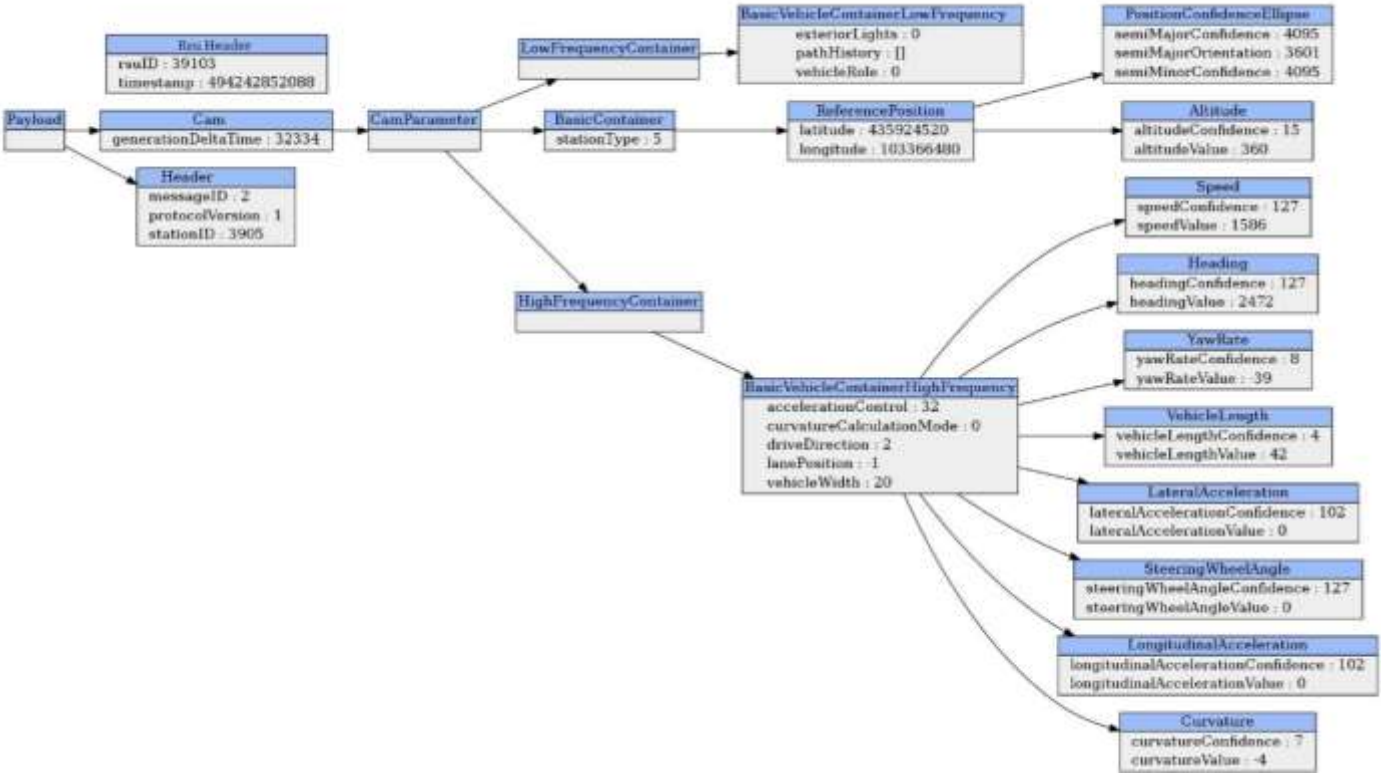

Figure 35. Overview of the ETSI C-ITS CAM Message Elements

The DENM broadcasted to the vehicle via ETSI ITS G5 radio is also published to the oneM2M when a new hazard event is triggered and every time that the DENM is updated or terminated. In these cases, the format is the same as the one shown in Figure 33.

The RSU can forward to the oneM2M platform also the data coming from a sensor because it is a border router for Wireless Sensor Networks as well. For those devices the data model is the same as that of the NB-IoT sensor. When the RSU is placed in the urban context, it is connected to the “smart” traffic light (see D2.4), thus the data published to the oneM2M include the message for the pedestrian detection at crossroad (see Figure 36) and the Signal Phase and Timing Message (ETSI ITS G5 SPAT message) of the traffic light, according to the scheme shown in Figure 37.

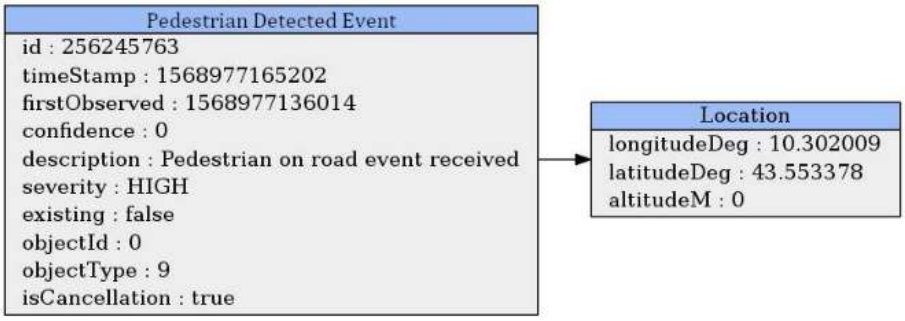

Figure 36. Pedestrian Detected Event Message

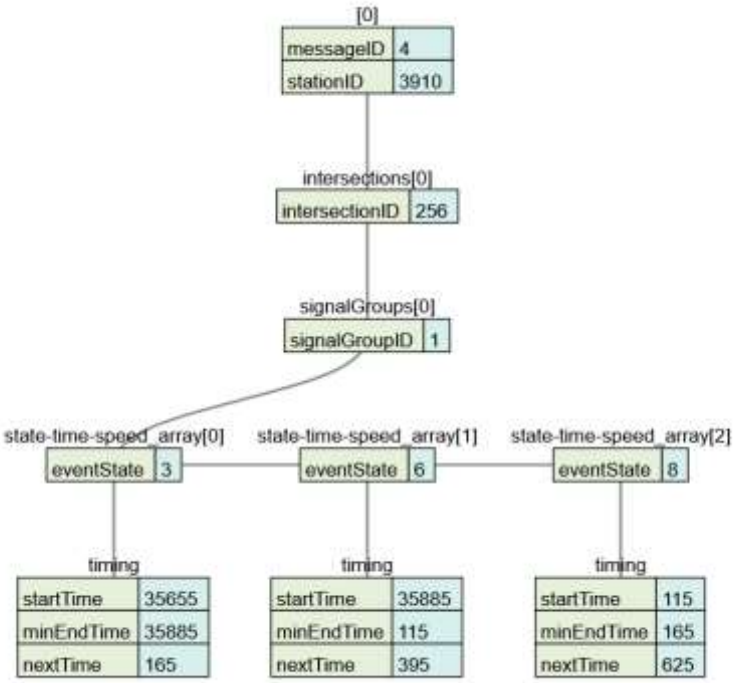

Figure 37. Signal Phase and Timing Message ETSI C-ITS SPAT Message Elements

6.3.4 Parking Package

The parking spot service allows end users to access information about parking spot locations and availability. Operational work of the service requires receiving real-time updates about the states of parking spots. These updates are published through an IoT Platform which allows multiple users to aggregate this information. Information about parking spots include both static and dynamic data. Static parking spot data are the time-invariant properties of the parking spots, such as their geographical locations, shapes, access points, etc. These properties usually remain constant or might change only rarely. Dynamic data represent parking spot properties that are expected to change over time. A typical example of the dynamic information is a parking spot occupation status (occupied or free) which changes as soon as a car enters or leaves the parking spot. Only dynamic information is going to be published through the IoT platform.

Different data models for parking spots are introduced by the transport community:

- FIWARE Parking Harmonized Data Models [FPM18]
- TomTom On-Street Parking [TTP18]
- Datex II Parking Publications Extension [DATEX2]

We selected DATEX II to represent the parking package, since DATEX II is an official multi-part standard, maintained by CEN Technical Committee 278, CEN/TC278, (Road Transport and Traffic Telematics).

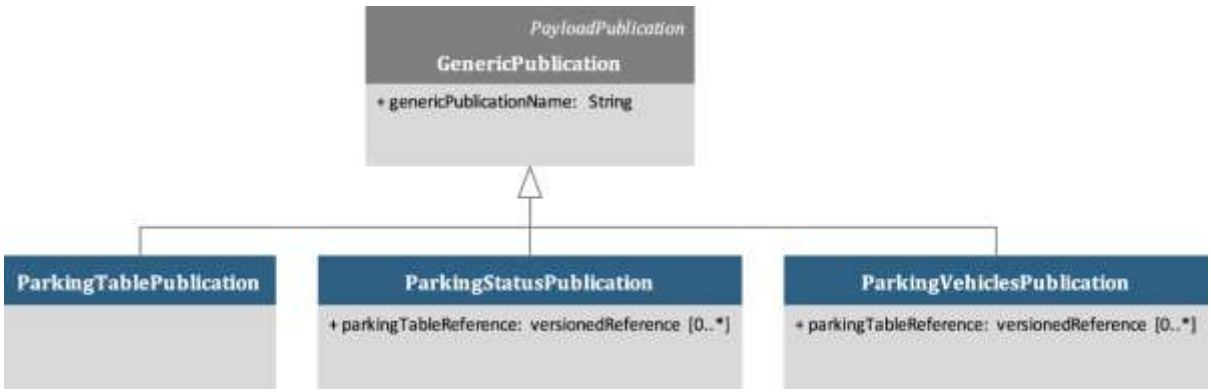

Figure 38. DATEX II Parking Extension – High-Level Data Elements

Figure 38 shows high-level data elements (classes) of the DATEX II parking model extension.

- Element *ParkingTablePublication* contains information about the parking spots (location, basic properties).
- Element *ParkingStatusPublication* is used for operational updates about parking spot occupation.
- Element *ParkingVehiclesPublication* is dedicated for the parked vehicle and is out of scope our work.

The Parking Table contains static information of parking sites and groups of parking spots. This covers urban or interurban parking areas, with both on-street and off-street parking (e.g., parking garages, parking places, motorway parking, etc.). Within one parking area, information can be specified down to groups of parking spots or even for individual parking spots.

The list below represents the main topics and properties covered by *ParkingTablePublication*:

- Parking allowance for different types of users and vehicles
- Basic parking information (number of parking spots, parking time and fees, etc.)
- Special conditions for specific users or vehicles (e.g., disabled permit, private permit)
- Information about an owner of a parking spot or group (name, address, etc.)
- Geographical location and drop off and pick up points
- Electric charging possibilities including technical characteristics and connection types
- Detailed rate information including specifications for different user / vehicle types, different seasons, payment options and booking fees
- A colour mapping for visualisation of possible occupancy statuses

The *ParkingStatusPublication* element is used for publishing information about parking spot occupancies and other dynamic properties, such as validity of parking spots. We extended the *ParkingStatusPublication* element to support project-specific information, such as confidence levels indicating the reliability of the published status (since this is typically detected by sensors).

6.3.5 AVP Package

The AVP package covers IoT “event” and “command” data models that define messages exchanged between IoT devices like an AD vehicle, a drone and an IoT platform in the corresponding use case. For the AVP use case the defined IoT event data model supports three types of IoT-devices (AD-vehicle, drone and camera). The IoT command data model defines communication messages that flow between an AD-vehicle and a MAV drone.

Figure 39 shows IoT devices like a vehicle, MAV and camera publish IoT event messages to the IoT platform and subscribe to AVP command messages published by the IoT AVP application.

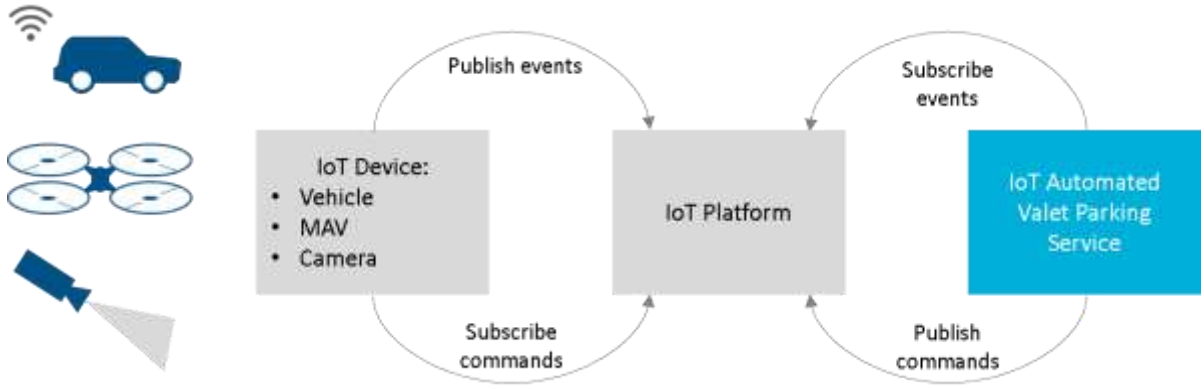

Figure 39. IoT event and command message exchange between IoT devices and IoT platform

6.3.5.1 AVP IoT Event Data Model

The AVP IoT event data model supports three types of IoT devices (AD-vehicle, drone and camera) and is described in this section. The model is based on the existing standardized SENSORIS vehicle data model. This model has been extended in case of vehicle and adapted in case of RSU-camera and MAV to support information specific to the AVP use case (see Figure 40). The AVP IoT event message itself consists of status information (e.g. avp vehicle status, current vehicle charging status, current vehicle position, parking spot occupancy and obstacle detection) published by an AD-vehicle, MAV or RSU camera to the IoT platform.

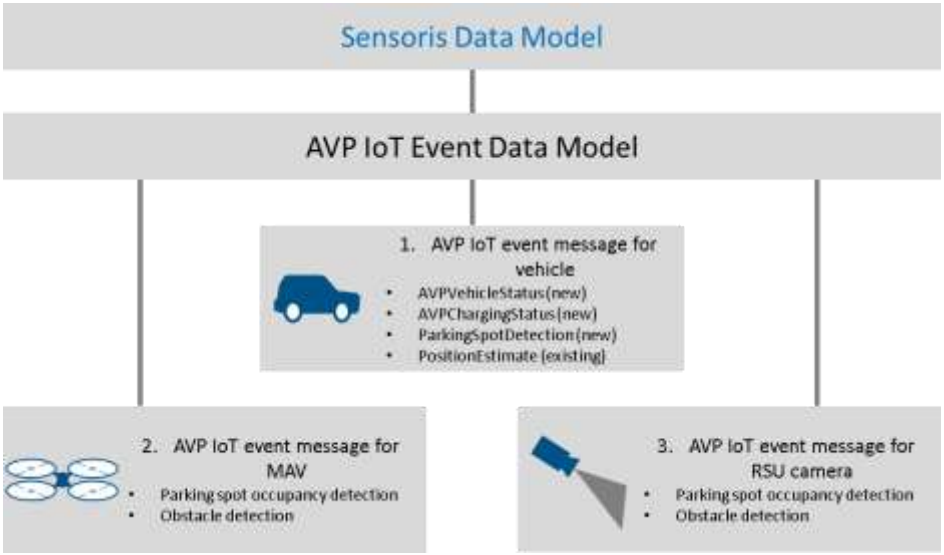

Figure 40. Supported AVP IoT event data models

The already existing SENSORIS vehicle data model has been extended to support the AVP use case. SENSORIS vehicle model was adopted to support raw IoT data from coming from vehicles in the AUTOPILOT project. Specific to the AVP use case status information (AVP status message) has been identified, specified and integrated into the existing SENSORIS data model (see Figure 40). In addition to the existing PositionEstimate event message relevant to the AVP use case, there are three new AVP event messages AVPVehicleStatus, AVPVehicleChargingStatus and ParkingSpotDetection has been defined and integrated into the SENSORIS model:

- PositionEstimation (existing): an AD-vehicle has to send on regular basis its current position

to the IoT platform, so that the driver can observe in real time on their smartphone how's valet parking procedure is going on. The SENSORIS vehicle data model provides PositionEstimation data model.

- AVPVehicleStatus (new): Models AVP status of a vehicle.
- AVPVehicleChargingStatus(new): Contains charging status of a vehicle.
- ParkingSpotDetection (new): Contains parking spot occupancy detection information in case of vehicle is equipped with sensors to detect free parking spots.

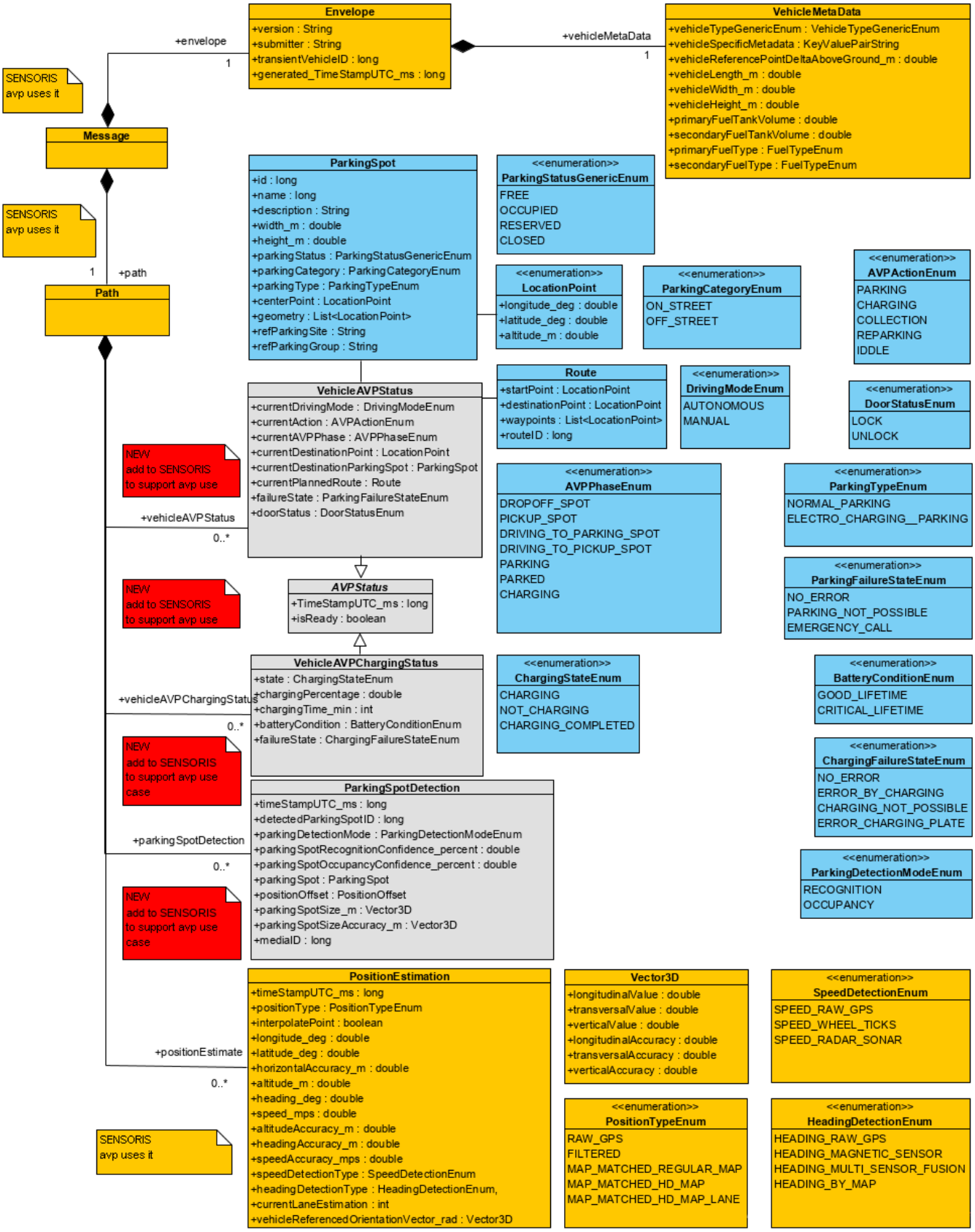

Figure 41. AVP IoT event data model specification for AD-vehicle as extension of SENSORIS

RSU-Camera IoT event data model

Camera as an IoT device in the AVP use case provides parking spot occupancy and object (e.g.

obstacle) detection information. For such purposes a camera was installed in the corresponding AVP parking area, configured to detect parking spot occupancy and connected to the IoT platform. This camera publishes events to the platform. An RSU-camera data model specification is new and is based on the SENSORIS vehicle data model. A full specification of the camera event message as defined within the DMAG is shown on the Figure 42.

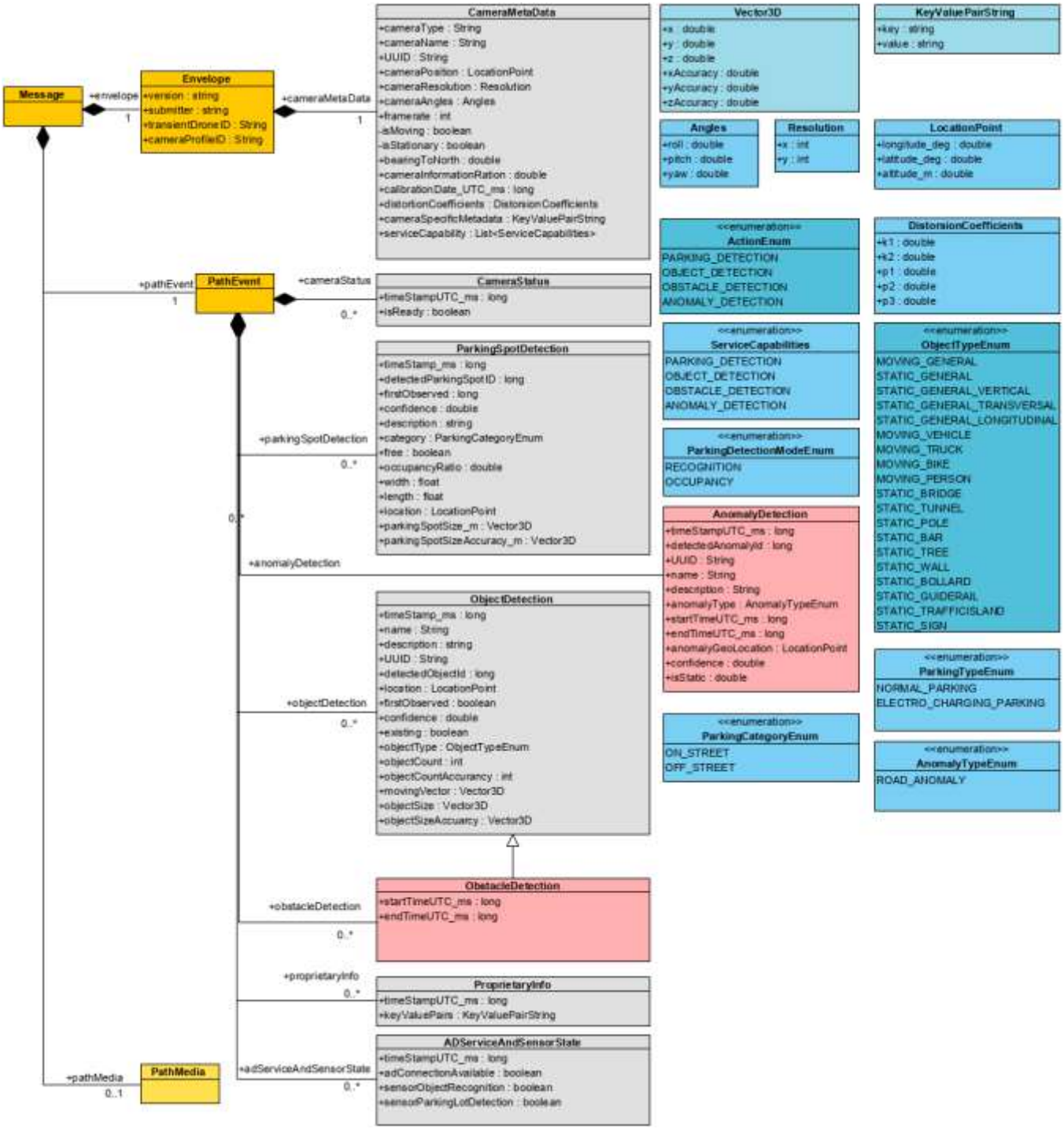

Figure 42. AVP IoT event data model specification for RSU-camera based on SENSORIS

MAV (drone) IoT event data model

MAV as an IoT device in the AVP use case provides parking spot occupancy and object (e.g. obstacle) detection information. After the MAV has received an AVP command flies to the specified parking

spots, verifies if the spots are empty and/or if there are obstacles on the road and publishes this information to the IoT platform. The MAV data model is new and is based on the SENSORIS vehicle data model. A full specification of the MAV IoT event message as defined within the DMAG is shown in the Figure 43.

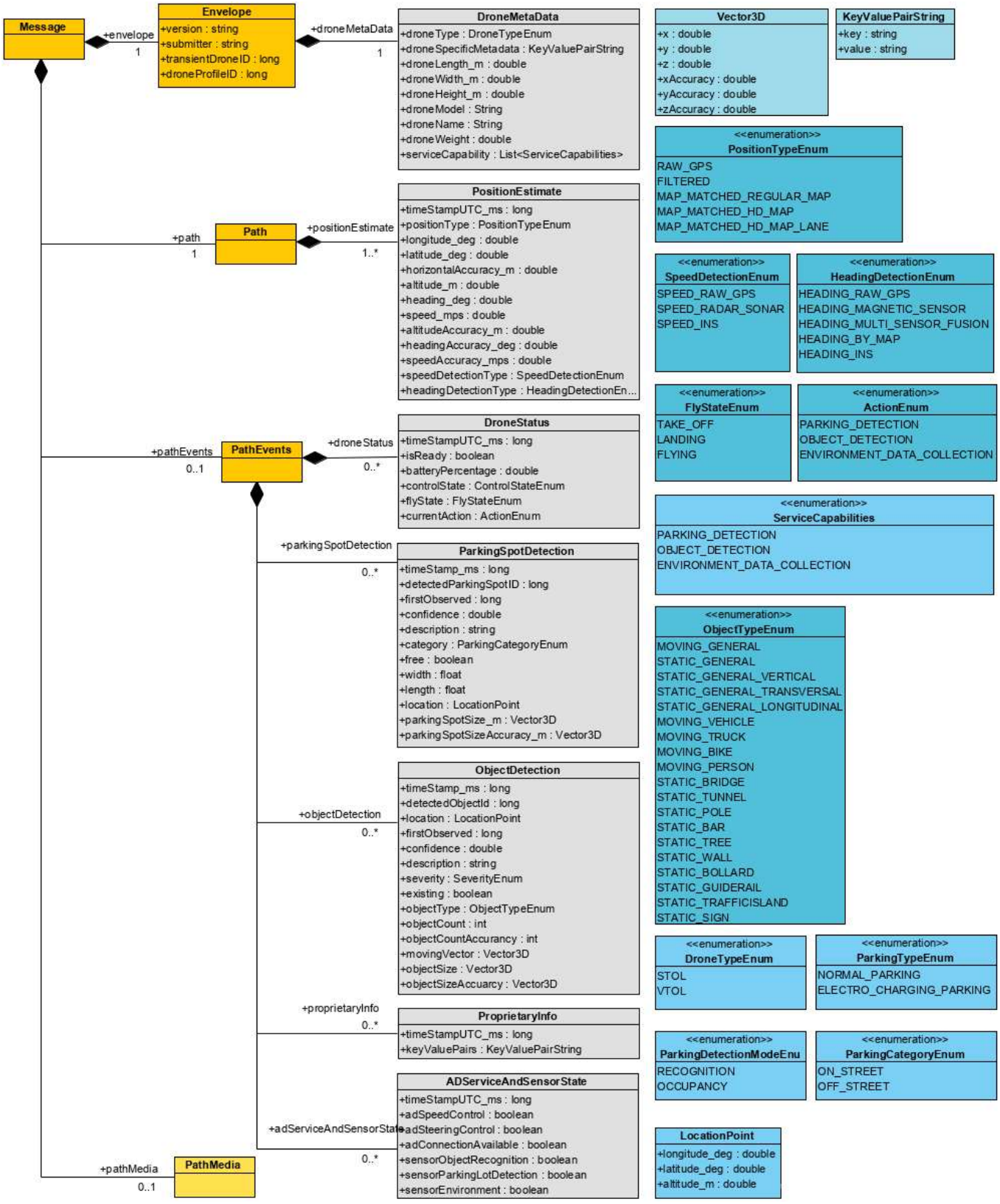

Figure 43. AVP IoT event data model specification for MAV based on SENSORIS

6.3.5.2 AVP IoT Command Data Model

Command data model defines an AVP command message that contains instructions required by an IoT device like an AD-vehicle or a drone to run valet parking or initiate detection of free spots. These IoT devices must first subscribe to AVP commands published on the IoT platform. Depending on the type of the target IoT a command message consists of two parts: VehicleAVPCommand and DroneAVPCommand. A full specification of the AVP command data model is depicted in Figure 44.

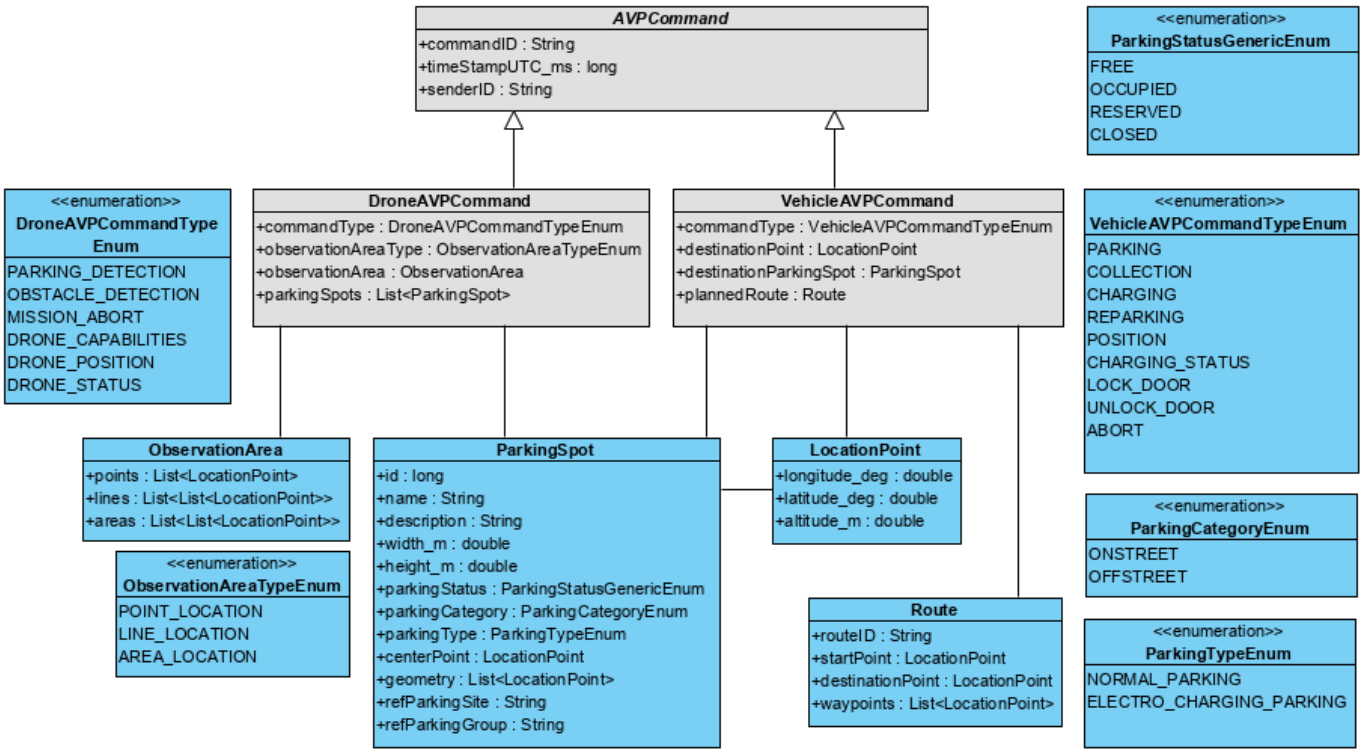

Figure 44. AVP IoT command data model specification for AD-vehicle and MAV

6.3.6 Platoon Package

Platooning use case implementation usually based on a combination of:

1. Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communication using ITS-G5 short-range communication. This is used for exchanging operational data (related to vehicle control, safety information) which typically require real-time low latency communication.
2. V2I communication and I2V using long-range cellular communication to support communications with IoT infrastructure. This is used to exchange tactical and strategic information related to platoon management, tracking, routing etc. This is typically less time critical or even not real-time.

Platooning IoT data model only focus on the information exchange with IoT infrastructure for platoon management implemented in a platooning service.

Platooning service needs to support a complex use-case which includes platoon formation, lane assignments, traffic light interaction and speed advices. Therefore, a service implementation (e.g. in Brainport) consists of two main modules:

- Platoon Formation: long distance communication is used to get vehicles which want to join a platoon to the agreed location at the agreed time.
- Platoon Management: information exchange needed to manage a platoon when vehicles are platooning (a short range). This is required for platoon monitoring (tracking and tracing) and for platoon management in local traffic (lane and speed advice, passing traffic light).

To supports these two modules multiple messages are used. There are two types of messages identified:

- Coming from individual vehicles and/or a platoon.
- Coming from the Platooning Service

IoT Messages

All messages defined in the platoon package have a set of typical properties:

- message identifier,
- station identifier which identifies an entity (e.g. device, vehicle, service) which generated a message,
- time stamp of the message.

Figure 45 shows message types defined in the platoon package.

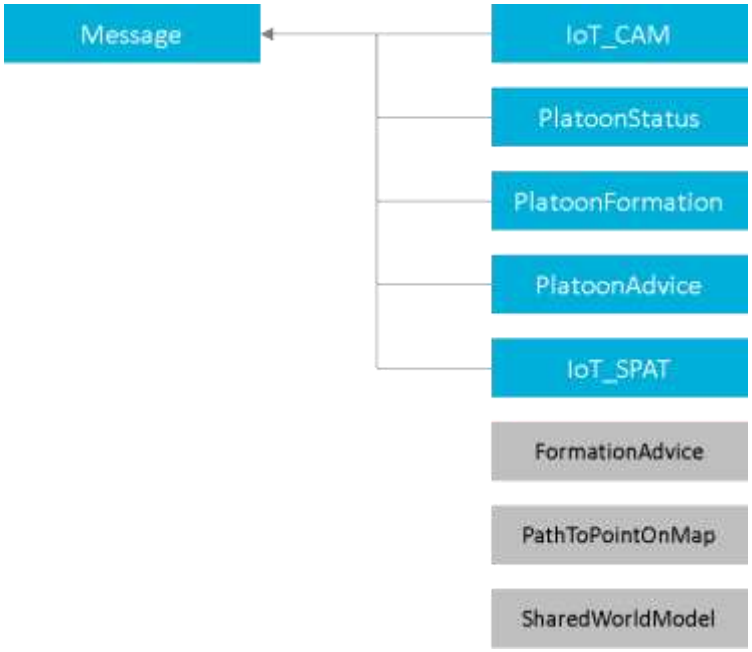

Figure 45. Platoon package message types.

Messages defined in the package:

- **IoT_CAM** messages are received from individual vehicles. This is based on Cooperative Awareness Message (CAM) standard which is available from ITS-G5 (ETSI EN 302 637-2). This message type includes information related to vehicle and its state. A limited set of attributes from the IoT_CAM type of message are used in platooning, these attributes describe vehicle state such as vehicle position, heading, and speed. With this information it is possible to track and trace individual vehicles.
- **PlatoonStatus**. Messages of PlatoonStatus type come from vehicles in a platoon and contain information actual platoon status like, amongst others, a unique platoon identifier, platoon state (available, forming, platooning, leaving), vehicle role (leader, following, trailing), the numbers of vehicles in the platoon.
- **PlatoonFormation**. Messages of PlatoonFormation come from vehicles that want to join a platoon. Such messages are actively used in the platoon formation stage and include a rendez-vous location and an estimated time of arrival.
- **PlatoonAdvice**. Message of this type are used when vehicles are already in a platoon and their purpose is to align a platoon with local traffic conditions and road situation. This message type includes information on lane assignment/advice, speed advice or traffic intensity.

- **IoT_SPAT**. This is based on SPAT (Signal Phase and Timing) messages which are available from local Traffic Light Controllers at the intersection. This message type is a subset of the original SPAT message type and includes information regarding intersection, related direction of travel across intersection and contains state-time-speed information.
- And additional auxiliary object types such as FormationAdvice, PathToPointOnMap, and SharedWorldModel which are used in the messages defined above.

6.4 Conclusion

The DMAG has proposed the common IoT data models based on existing data models (SENSORIS and DATEX II) and extensions on them to support the use case implementations in the project and increase interoperability level between platforms and pilot sites.

Table 13 shows conformance to the common models at the pilot sites.

Table 13. Status of the common data model implementation

Pilot Sites vs. Use Cases	Finland (Tampere)	France (Versailles)	Italy (Livorno)	The Netherlands (Brainport)	Spain (Vigo)	South Korea (Deajeon)
Automated Valet Parking	–			+	+	
Highway Pilot			+	–		
Platooning		–		+		
Urban Driving	–	–	+	–	+	–
Car/Ride Sharing		–		–		

There are two common reasons why some of the use case implementations are not compliant with the designed common data models:

- An implementation of a use case has started earlier than the model proposed and finalized by DMAG.
- A proprietary or incompatible standard already exists and supported by local infrastructure and/or devices/sensors.

Even though not many pilot sites actually followed recommendations from DMAG usually data models that are used by the pilot sites are based on industry standards such as SENSORIS or DATEX II which were considered as a baseline for the models elaborated in the project. Having said that, enforcing industry standards and/or additional legislations to decrease a number of standards/formats used in AD function implementations may reduce further interoperability costs.

So standardisation or enforcing standards is a crucial task when an implementation of a feature requires communications with several systems and services especially AD features that affect quality of life and safety of mobility.

7 Evaluation and Conclusion

In this section we provide overall conclusion of the IoT platform implementation and lessons learned during the project implementation, tests and runs handled in the project.

7.1 Evaluation of the IoT Platform Implementation

There are eight IoT platforms were identified and implemented in the project, more than the total number of the pilot sites organized in the project. That means we have more than one IoT platform per pilot site in average. There's an absolute leader of the pilot sites, Brainport, where five platforms

were implemented and used in the test cases. So, despite the features provided by the platforms are comparable to each other, maintaining several platforms and making them interoperable may be challenging. So, common terminology, support of standards by the platforms and interoperability are vital to have.

In general, all IoT platforms serve a goal of bringing devices online, collecting data from many devices, providing access to the data, and analyzing the data. As IoT platform is a broad term and almost every vendor define it in its own way, users may hit a terminological problem when assessing functionality of IoT platforms. In fact, basic functionally of any platform requires following:

- Device connectivity/discovery
- Device management
- Device messaging
- Information security

All the project platforms provide this minimal basic functionality and in fact they provide much more features. But a common problem is that they have their own terms and definitions. So, when a user tries to register a device the process may be similar on a high level but differs in details. This difference in IoT terminology and concepts defined by the platforms should be considered when one builds a federated IoT architecture that spans across geographically distributed sites and requires participation of many partners and IoT platforms.

Even though the platforms implemented in the project are from different vendors and support different protocols/standards, there is one standard that should be probably considered as a common or as an interoperable standard for potential future implementations – oneM2M standard. Out of eight IoT platforms three of them (Sensinov, TIM, openMTC) are oneM2M compliant and more and share the same concepts. An IoT platform from Huawei implemented in the project provides a oneM2M Interworking Proxy Entity which implements oneM2M MCA interface and therefore the platform may interface with other oneM2M compliant platforms. In total this makes four platforms compatible and this is a very notable achievement.

In Brainport where most of the platform were used, we also verified that the platforms are either interoperable out of box or can be made interoperable. There were implemented three gateways connecting FIWARE, Huawei and IBM platforms to the interoperability oneM2M platform by Sensinov. This brings IoT architecture to another layers – a federated IoT architecture as it was designed initially in the project and such an architecture is fundamental in large IT environments such cities and countries.

7.2 Lessons Learned

This task 2.3 “Development of the open service platform” as part of the Work Package 2 “Development Integration and Verification” was one of the AUTOPILOT’s most complex tasks, which includes implementation and integration of several proprietary and open source IoT platforms across six pilot sites. These IoT platforms had to be efficiently designed, built to work in synergy as specified in the federated IoT architecture and had to integrate various IoT ecosystems deployed at the pilot sites.

An important point is that the results of this task were the prerequisites of the implementation and testing of the use cases at the pilot site. Therefore, implemented IoT platform infrastructure as an outcome of this task had to be delivered in a short time and be operational to facilitate activities in other tasks. Next activities are closely related to the task:

- Development and deployment of an optimal and efficient open IoT service platform that can be used at different pilot sites to demonstrate autonomous driving use cases.
- Specification of the common IoT data models that can be used at the pilot sites to exchange information between components, e.g. between IoT devices, AD enabled vehicles and IoT

platforms.

- Development of the gateway proxy to interconnect several IoT platforms to ensure platform interoperability.

Besides the abovementioned activities there were other activities that may look straightforward, but they are in fact time consuming and important for overall success of task 2.3:

- Coordination of the IoT platform implementation activities between the pilot sites, which was not an easy task due to a number of involved partners from the different pilot sites. Several phone conferences and face-to-face meetings have been initiated to discuss open questions and action points required to solve in the implementation and integration of IoT platforms at pilot sites.
- Synchronization and coordination of work efforts for the implementation of the several gateway proxies required to interconnect different IoT platforms and achieved platform federation and interoperability. For these purposes a few collaborations were created:
    - Collaboration 1 (IBM, Sensinov, TNO, and DLR): to implement a gateway between IBM Watson IoT platform operated by IBM and oneM2M interoperability platform developed by Sensinov and operated by TNO.
    - Collaboration 2 (Huawei, Sensinov, and TNO): to implement a gateway between Huawei OceanConnect IoT platform operated by HUAWEI and oneM2M interoperability platform developed by Sensinov and operated by TNO.
    - Collaboration 3 (NEC, Sensinov, and TNO): to implement a gateway between FIWARE IOT Broker operated by NEC and oneM2M interoperability platform developed by Sensinov and operated by TNO .
- Organization of the data modeling activity group (DMAG) to coordinate activities on standardization and modeling of data formats to be used in all the use cases. The DMAG started to work on the models with a delay so that the delivered IoT data models could not be adapted by all partners in the development of the use cases at the pilot sites. Hence early data modelling and standardization activities are essential for compatibility between implementations.

7.3 Conclusion

The open IoT platform architectures for autonomous driving use cases (e.g. automated valet parking, urban driving, highway pilot, platooning, car/ride sharing) have been designed, developed, deployed and tested in the six AUTOPILOT pilot sites Brainport (The Netherlands), Versailles (France), Vigo (Spain), Tampere (Finland), Livorno (Italy) and Deajeon (South Korea) as shown in Table 14:

Table 14. Pilot sites and use cases in the AUTOPILOT project

Pilot Sites vs. Use Cases	Finland (Tampere)	France (Versailles)	Italy (Livorno)	The Netherlands (Brainport)	Spain (Vigo)	South Korea (Deajeon)
Automated Valet Parking	+	-	-	+	+	-
Highway Pilot	-	-	+	+	-	-
Platooning	-	+	-	+	-	-
Urban Driving	+	+	+	+	+	+
Car/Ride Sharing	-	+	-	+	-	-

There are eight different IoT platforms developed and integrated into different architecture components and interfaced/connected to the infrastructure of each pilot site (see Table 15). The IoT platforms developed in the scope of the AUTOPILOT project consist of a proprietary solution (e.g.

IBM Watson IoT platform, Huawei OceanConnect IoT platform) and an open source solution like openMTC oneM2M IoT platform. The IoT platforms are used for collecting, exchanging and processing data from IoT devices and AD-vehicles in the different use cases at the different pilot sites.

Table 15. Pilot sites and deployed IoT platforms in the AUTOPILOT project

Pilot Sites vs. Use Cases	Finland (Tampere)	France (Versailles)	Italy (Livorno)	The Netherlands (Brainport)	Spain (Vigo)	South Korea (Deajeon)
IBM Watson IoT	-	-	-	+	+	-
FIWARE	-	-	-	+	-	-
HUAWEI OceanConnect	-	-	-	+	-	-
SENSINOV oneM2M	-	+	-	+	-	-
openMTC oneM2M	+	-	-	-	-	-
TIM oneM2M	-	-	+	-	-	-
MESIM IoT	-	-	-	-	-	+
MobiMaestro	-	-	-	+	-	-

The AUTOPILOT IoT services platform was designed as federation of several IoT platforms to achieve interoperability between the different platforms. For this purpose, an oneM2M standard compatible IoT platform (operated by SENSINOV) was deployed as an “oneM2M interoperability platform” and an interconnection gateway proxy was implemented to connect the interoperability platform to the proprietary platforms like IBM Watson IoT, HUAWEI OceanConnect, and FIWARE platforms. Interoperability between the platforms was successfully tested at the Dutch pilot site (Brainport) in the automated valet parking use case and in the car/ride sharing use case.

During pilot site test activities various IoT devices, autonomous driving vehicles, and different IoT platforms from different vendors have been used.

A common set of standardized IoT data models were required to support interoperability between use case implementations and between pilot sites (e.g. AVP use case in Brainport, Tampere and Vigo). These models standardize information data exchange between IoT devices (e.g. road site camera, MAV-drone, parking lot), AD enabled vehicles, road side units (e.g. traffic lights), and implemented IoT platforms in the project. These models leverage interoperability between all components. The IoT common data models were designed and implemented within the Data Modelling Activity Group (DMAG) and consist of several packages for traffic, parking, AVP, platooning data objects. The models were used in the use case implementations at the pilot sites in Italy, The Netherlands, and Spain.

The results achieved in the AUTOPILOT project show that IoT technologies with cloud support can facilitate autonomous driving functions in use cases such as urban driving/vehicles rebalancing, car/ride sharing, platooning, highway driving and automated valet parking.

8 Annex

8.1 Interacting with the SENSINOV oneM2M Platform

This annex provides examples how to interact with the Sensinov oneM2M platform.

Table 16. Interacting with Sensinov oneM2M Platform

Generate credentials for a new Application Entity

```
POST /server HTTP/1.1
Host: oneM2M-endpoint
X-M2M-Origin: Cae-admin
X-M2M-Key: changeme
Content-Type: application/json;ty=19
Accept: application/json
{
            "m2m:asar": {
                  "rn":"asarTest",
                  "aae":["Cae-test"],
                  "apci":["mypassword"]
            }
}
```

Register Application Entity (AE)

```
POST /server HTTP/1.1
Host: oneM2M-endpoint
X-M2M-Origin: Cae-test
X-M2M-Key: mypassword
Content-Type: application/json;ty=2
Accept: application/json
{
            "m2m:ae": {
                  "rn": "aeTest",
                  "rr": false,
                  "api": "test.company.org"
            }
}
```

Create Content Instance inside Container

```
POST /server/aeTest/cntTest HTTP/1.1
Host: oneM2M-endpoint
X-M2M-Origin: Cae-test
X-M2M-Key: mypassword
Content-Type: application/json;ty=4
Accept: application/json
{
            "m2m:cin": {
                  "rn": "cinTest",
                  "cnf": "application/json",
                  "con": "{\"temperature\":21,\"timestamp\":1517912099}"
            }
}
```

Retrieve latest Content Instance from Container

```
GET /server/aeTest/cntTest/la HTTP/1.1
Host: oneM2M-endpoint
X-M2M-Origin: Cae-test
X-M2M-Key: mypassword
Accept: application/json
```

Retrieve all Content Instance from Container

```
GET /server/aeTest/cntTest?rcn=4 HTTP/1.1
Host: oneM2M-endpoint
X-M2M-Origin: Cae-test
X-M2M-Key: mypassword
Accept: application/json
```

Subscribe to Container

```
POST /server/aeTest/cntTest HTTP/1.1
Host: oneM2M-endpoint
X-M2M-Origin: Cae-admin
X-M2M-Key: changeme
Content-Type: application/json;ty=23
Accept: application/json
{
            "m2m:sub": {
                  "rn": "subTest",
                  "nu": ["http://subscriber-ip:subscriber-port"],
                  "nct": 2,
                  "enc": {
                        "net": 3
                  }
            }
}
```

9 References

[AG15] Amsterdam Group: Signal Phase and Time (SPAT) and Map Data (MAP), white paper, 01 September 2015: https://amsterdamgroup.mett.nl/downloads/handlerdownloadfiles.ashx?idnv=500795. Accessed on 20 February 2018.

[CAM14] ETSI EN 302 637-2 (V1.3.2): "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service". November 2014.

[D2.1] D2.1 – Vehicle IoT Integration Report. AUTOPILOT Deliverable. February 2018.

[DATEX2] DATEX II User Guide: https://gitlab.com/autopilot/iot-data-model/uploads/f4c9c67169cad3dc30e0ec444ce7f34a/DATEXuserGuide.pdf. Accessed on 25 February 2018.

[DENM14] ETSI EN 302 637-3 (V1.2.2): "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service". November 2014.

[FPM18] FIWRE Data Models, Parking Harmonized Data Models: http://fiware-datamodels.readthedocs.io/en/latest/Parking/doc/introduction/index.html. Accessed on 25 February 2018.

[FW] FIWARE Wiki: https://forge.fiware.org/plugins/mediawiki/wiki/fiware. Accessed on 29 October 2019.

[FWa] FIWARE Architecture Description, IoT Backend, IoT Discovery: https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FIWARE.Architecture Description.IoT.Backend.IoTDiscovery. Accessed on 29 October 2019.

[HERE15] Vehicle Sensor Data Cloud Ingestion Interface Specification (v2.0.2), 2015: https://lts.cms.here.com/static-cloud-content/Company_Site/2015_06/Vehicle_Sensor_Data_Cloud_Ingestion_Interface_Specification.pdf. Last accessed on 25 February 2018.

[HOC] HUAWEI Developers Official Site, OceanConnect: http://developer.huawei.com/ict/en/site-oceanconnect_doc. Accessed on 24 February 2018.

[IBM18a] Getting Started with Watson IoT Platform: https://console.bluemix.net/docs/services/IoT/index.html. Accessed on 07 February 2018.

[IBM18b] Cloudant NoSQL DB: https://console.bluemix.net/catalog/services/cloudant-nosql-db. Accessed on 07 February 2018.

[IBM18c] Watson IoT for Automotive: https://www.ibm.com/support/knowledgecenter/en/SSNQ4V/iot-automotive/overview/overview.html. Accessed on 12 February 2018.

[IBM18d] IBM IoT for Automotive : Vehicle Data Hub: https://developer.ibm.com/api/view/id-551:title-IBM_IoT_for_Automotive___Vehicle_Data_Hub. Accessed on 12 February 2018.

[IBM18e] IBM Watson IoT Context Mapping: https://developer.ibm.com/api/view/id-263:title-IBM_Watson_IoT_Context_Mapping. Accessed on 12 February 2018.

[ISOTC204] ISO/TC 204, Intelligent Transport Systems: https://www.iso.org/committee/54706.html. Last visited on 20 February 2018.

[ONE17] oneM2M - Standards for M2M and the Internet of Things: http://onem2m.org/. Accessed on 13 February 2018.

[oneM2M -TR-0039] "Developer guide: Interworking Proxy using SDT", oneM2M TR-0039-V-0.0.5, 21-09-2017, to be retrieved via (seen in June 2019)

[OIP17] Ocean IoT platform: http://www.iotocean.org/. Accessed on 23 February 2018.

[OSGi] OSGi Alliance official website, https://www.osgi.org . Accessed on 27 February 2018.

[MQTT] MQTT official website, http://mqtt.org. Accessed on 25 June 2018.

[NGSI10] FI-WARE NGSI-10 Open RESTful API Specification: https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FI-WARE_NGSI-10_Open_RESTful_API_Specification. Accessed on 23 February 2018.

[NGSI9] FI-WARE NGSI-9 Open RESTful API Specification: https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FI-WARE_NGSI-9_Open_RESTful_API_Specification. Accessed on 23 February 2018.

[TS09] oneM2M Technical Specification: TS-0009-V2.6.1 – HTTP Protocol Binding: http://www.onem2m.org/images/files/deliverables/Release2/TS-0009-HTTP_Protocol_Binding-V2_6_1.pdf. Accessed on 25 February 2018.

[TS10] oneM2M Technical Specification: TS-0010-V2.4.1 – MQTT Protocol Binding: http://onem2m.org/images/files/deliverables/Release2/TS-0010-MQTT%20Protocol%20Binding-V2_4_1.pdf. Accessed on 25 February 2018.

[TS08] oneM2M Technical Specification: TS-0008-V1.0.1 – CoAP Protocol Binding: http://onem2m.org/images/files/deliverables/TS-0008-CoAP_Protocol_Binding-V1_0_1.pdf. Accessed on 25 February 2018.

[TTP18] TomTom On-Street Parking: https://developer.tomtom.com/street-parking/street-parking-api-draft-one-page. Accessed on 25 February 2018.