Fifth Generation Cross-Border Control

Deliverable D3.1
Final Application Architecture
Version: v1.0
2021-01-31

DISCLAIMER: This 5GCroCo D3.1 deliverable is not yet approved by the European Commission.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 825050. Any 5GCroCo results reflects only the authors’ view and the Commission is thereby not responsible for any use that may be made of the information it contains.

http://www.5g-ppp.eu
# Deliverable D3.1

## Final Application Architecture

<table>
<thead>
<tr>
<th>Grant Agreement Number:</th>
<th>825050</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Name:</td>
<td>Fifth Generation Cross-Border Control</td>
</tr>
<tr>
<td>Project Acronym:</td>
<td>5GCroCo</td>
</tr>
<tr>
<td>Document Number:</td>
<td>5GCroCo/D3.1</td>
</tr>
<tr>
<td>Document Title:</td>
<td>Final Application Architecture</td>
</tr>
<tr>
<td>Version:</td>
<td>v1.0</td>
</tr>
<tr>
<td>Delivery Date:</td>
<td>2021-01-31</td>
</tr>
<tr>
<td>Editor(s):</td>
<td>Ulf Larson (Volvo), Eric Perraud (RSA), Maciej Muehleisen (ERI)</td>
</tr>
<tr>
<td>Authors:</td>
<td>Kurt Eckert, Theo Floess, Jens Schwardmann, Erik Walossek, Stefan Foell, Tobias Müller (BOSCH), Francisco Vázquez-Gallego, Roshan Sedar, Juan Luis de la Cruz, Ramon Casellas (CTTC), Oliver Goldich (DTAG), Maciej Muehleisen (ERI), Soumya Kanti Datta (EUR), Juri Kuhn (FORTISS), Apostolos Kousaridas (HWDU), Frédéric Gardes, Marc Lacoste, Mathieu Lefebvre, Jean-Marc Odinot (Orange), Rodrigo Silva, Laurent Dizambourg, Antonio Eduardo Fernandez Barciela (PSA), Andreas Schimpe (TUM), Eric Perraud, Stefania Sesia (RSA), Ulf Larson, Henrik Segesten (Volvo), Andreas Pfadler, Israel Gonzalez Vazquez (Volkswagen AG), Xavi Vilajosana (WSE)</td>
</tr>
<tr>
<td>External Reviewers:</td>
<td>Mazen Abdel Latif, Mikael Fallgren, Yunpeng Zang (ERI), Daniel Rau (HTW), Francesca Moscatelli (NXW), Alexander Rechberger, Steffen Robertz (SEC), Andreas Perme, Jakob Larsson (Volvo)</td>
</tr>
<tr>
<td>Keywords:</td>
<td>Application Architecture, Tele-operated Driving, ToD, Anticipated Cooperative Collision Avoidance, ACCA, HD Mapping, Backend Architecture, MEC, QoS Prediction</td>
</tr>
<tr>
<td>Status:</td>
<td>Final</td>
</tr>
<tr>
<td>Dissemination level:</td>
<td>Public</td>
</tr>
</tbody>
</table>
## Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date of release</th>
<th>Description of changes</th>
<th>Last editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2021-01-31</td>
<td>Original version 1.0</td>
<td>Ulf Larson</td>
</tr>
</tbody>
</table>
Abstract

This deliverable takes the user story descriptions from Deliverable D2.2 [1] and the use case requirements from Deliverable D2.1 v3 [2], to provide an overview of the 5GCroCo application architecture of the vehicle and the backend side for the three 5GCroCo use cases: Tele-operated Driving (ToD), High-Definition (HD) mapping, and Anticipated Cooperative Collision Avoidance (ACCA). Project partners provided descriptions of the individual solutions they designed, developed and evaluated in the project.

After introducing the scope and objective of the document, the application architecture for the vehicle and the backend side for each use case is described in detail. First, the main software components are identified, and an overall architecture picture is provided. Then, the vehicle software architecture and message flows between the vehicle and the backend, consisting of public Internet and MEC are described. Then, aspects of predictive Quality of Service are discussed, followed by a description and discussion of security solutions for the use case. Finally, a plausible real-world architecture for the use case is described.

The architecture described in this document has been verified through trials and is generalized to reflect commercial real-world deployments.
Contents

1 Introduction ........................................................................................................................................... 9
  1.1 Objective of the Document ........................................................................................................ 9
  1.2 Structure of the Document ......................................................................................................... 9
2 5GCroCo Use Cases ......................................................................................................................... 11
  2.1 Tele-operated Driving ................................................................................................................. 11
  2.2 High-definition Mapping ............................................................................................................. 12
  2.3 Anticipated Cooperative Collision Avoidance ........................................................................... 13
  2.4 User Story Overview .................................................................................................................. 13
3 Detailed Application Architecture Description ................................................................................ 15
  3.1 Use Case 1: ToD ......................................................................................................................... 15
    3.1.1 Overall Trial Architecture and Main Components ................................................................... 15
    3.1.2 Interface between Vehicle and VCoC ..................................................................................... 16
    3.1.3 Detailed Component Architectures ...................................................................................... 20
    3.1.4 Tele-operated Driving Session ............................................................................................... 26
    3.1.5 Predictive Quality of Service ................................................................................................. 30
    3.1.6 Security ................................................................................................................................. 32
    3.1.7 Real-world Architecture ....................................................................................................... 33
  3.2 Use Case 2: HD Mapping ............................................................................................................. 35
    3.2.1 Overall Trial Architecture and Main Components ................................................................... 35
    3.2.2 Vehicle Side Architecture Components ................................................................................ 35
    3.2.3 Backend Side Architecture Components ............................................................................... 36
    3.2.4 Main Application Flows ......................................................................................................... 37
    3.2.5 Predictive Quality of Service ................................................................................................. 40
    3.2.6 Security ................................................................................................................................. 41
    3.2.7 Real-world Architecture ....................................................................................................... 41
  3.3 Use Case 3: ACCA ..................................................................................................................... 44
    3.3.1 Overall Trial Architecture .................................................................................................... 44
    3.3.2 Communication between Vehicle and Backend ................................................................. 51
3.3.3 Onboard Software Architecture ................................................................. 60
3.3.4 Security .................................................................................................... 65
3.3.5 Real-world Architecture ............................................................................ 65

4 Summary and Way Forward ........................................................................... 80

5 References ....................................................................................................... 81

A Methods of Georeferencing and Mapping to MQTT Topics ......................... 84
   A.1 Fixed Quantified Latitude and Longitude .................................................... 84
   A.2 Quadtree ................................................................................................... 84
   A.3 Other ......................................................................................................... 85

B Common Legend for Architecture Figures ...................................................... 86
## List of Abbreviations and Acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3GPP</td>
<td>Third Generation Partnership Project</td>
</tr>
<tr>
<td>ACCA</td>
<td>Anticipated Cooperative Collision Avoidance</td>
</tr>
<tr>
<td>AD</td>
<td>Automated Driving</td>
</tr>
<tr>
<td>ADAS</td>
<td>Advanced Driving Assist System</td>
</tr>
<tr>
<td>AMQP</td>
<td>Advanced Message Queuing Protocol</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AS</td>
<td>Application Server</td>
</tr>
<tr>
<td>BTP</td>
<td>Basic Transport Protocol</td>
</tr>
<tr>
<td>CAM</td>
<td>Cooperative Awareness Message</td>
</tr>
<tr>
<td>CAN</td>
<td>Controller Area Network</td>
</tr>
<tr>
<td>CAT</td>
<td>Category</td>
</tr>
<tr>
<td>CCAM</td>
<td>Cooperative, Connected, and Automated Mobility</td>
</tr>
<tr>
<td>CCU</td>
<td>Communication Control Unit</td>
</tr>
<tr>
<td>CPM</td>
<td>Cooperative Perception Message</td>
</tr>
<tr>
<td>DATEX</td>
<td>Data Exchange</td>
</tr>
<tr>
<td>DENM</td>
<td>Decentralized Environmental Notification Message</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name Service</td>
</tr>
<tr>
<td>DSRC</td>
<td>Dedicated Short Range Communication</td>
</tr>
<tr>
<td>ETSI</td>
<td>European Telecommunications Standards Institute</td>
</tr>
<tr>
<td>FQDN</td>
<td>Fully Qualified Domain Name</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global Navigation Satellite System</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>GSM</td>
<td>Global System for Mobile Communications</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HD</td>
<td>High Definition</td>
</tr>
<tr>
<td>HMI</td>
<td>Human Machine Interface</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technology</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>ITS</td>
<td>Intelligent Traffic System</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>JWT</td>
<td>JSON Web Token</td>
</tr>
<tr>
<td>KPI</td>
<td>Key Performance Indicator</td>
</tr>
<tr>
<td>LTE</td>
<td>Long Term Evolution</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>MEC</td>
<td>Mobile Edge Cloud</td>
</tr>
<tr>
<td>MIMO</td>
<td>Multiple Input Multiple Output</td>
</tr>
<tr>
<td>MNO</td>
<td>Mobile Network Operator</td>
</tr>
<tr>
<td>MQTT</td>
<td>Message Queueing Telemetry Transport</td>
</tr>
<tr>
<td>MSC</td>
<td>Message Sequence Chart</td>
</tr>
<tr>
<td>NAP</td>
<td>National Access Point</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>NWDAF</td>
<td>Network Data Analytics Function</td>
</tr>
<tr>
<td>OEM</td>
<td>Original Equipment Manufacturer</td>
</tr>
<tr>
<td>OBD</td>
<td>On-Board Diagnostic</td>
</tr>
<tr>
<td>OBU</td>
<td>On-Board Unit</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>OTT</td>
<td>Over-the-Top</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PCAN</td>
<td>Peak Controller Area Network</td>
</tr>
<tr>
<td>PCIe</td>
<td>Peripheral Component Interface Express</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>PDN</td>
<td>Packet Data Network</td>
</tr>
<tr>
<td>PER</td>
<td>Packet Error Rate</td>
</tr>
<tr>
<td>PF</td>
<td>Prediction Function</td>
</tr>
<tr>
<td>P-GW</td>
<td>PDN Gateway</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure</td>
</tr>
<tr>
<td>PPS</td>
<td>Pulse-per-second</td>
</tr>
<tr>
<td>pQoS</td>
<td>predictive Quality-of-Service</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>ROI</td>
<td>Region of Interest</td>
</tr>
<tr>
<td>RSU</td>
<td>Roadside Unit</td>
</tr>
<tr>
<td>RTA</td>
<td>Road Traffic Authority</td>
</tr>
<tr>
<td>RTCP</td>
<td>Real-time Transport Control Protocol</td>
</tr>
<tr>
<td>RTP</td>
<td>Real-time Transport Protocol</td>
</tr>
<tr>
<td>RTSP</td>
<td>Real-time Streaming Protocol</td>
</tr>
<tr>
<td>RTT</td>
<td>Round-trip time</td>
</tr>
<tr>
<td>SP</td>
<td>Service Provider</td>
</tr>
<tr>
<td>SRTP</td>
<td>Secure Real-time Transport Protocol</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TMS</td>
<td>Traffic Management Service</td>
</tr>
<tr>
<td>ToD</td>
<td>Tele-operated driving</td>
</tr>
<tr>
<td>TTL</td>
<td>Time to live</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UE</td>
<td>User Equipment</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>VCoC</td>
<td>Vehicle Control Center</td>
</tr>
<tr>
<td>WAVE</td>
<td>Wireless Access in Vehicular Environments</td>
</tr>
<tr>
<td>WCDMA</td>
<td>Wideband Code Division Multiple Access</td>
</tr>
</tbody>
</table>
1 Introduction

This document describes the application architecture of the vehicle and the backend side for the three 5GCroCo use cases Tele-operated Driving (ToD), High-Definition (HD) mapping, and Anticipated Cooperative Collision Avoidance (ACCA). The architecture described in this document has been verified through trials and is generalized to reflect commercial real-world deployments and interface with the intermediate network architecture described in Deliverable D3.2 [3].

1.1 Objective of the Document

This document describes the software architecture of the different components of each use case, up to a level of detail that allows to implement and integrate the different use cases. Deliverable D3.1 describes the software architecture that is implemented on-board (in vehicles), the backend services that are deployed either in the public Internet or the Mobile Edge Computing/Cloud (MEC). Deliverable D3.1 also describes accurately the message flows of the 5GCroCo use cases, which are exchanged between the different software components (covering both the data plane flows and the control plane flows), the underlying protocols and the triggering conditions as well as the expected actions. When multiple architecture options are possible, they are listed, and the preferred ones are justified. This document also describes and discusses security solutions used for the trials for each use case. For each use case, a plausible real-world architecture is also described.

This document does not describe the network architecture: the network is considered as a black box offering some features like Quality-of-Service (QoS) prediction, data routing towards MEC, etc. Some, but not all interfaces for that are standardized. Connections between vehicle and backend are data pipes with certain performance that can vary over time.

1.2 Structure of the Document

First, Section 2 presents the three 5GCroCo use cases together with their respective user stories. Then, Section 3, provides for each Use Case a detailed application architecture description. Section 4 then provides a summary and way forward.

In Section 3, the same template is applied for the three 5GCroCo use cases. First, the main software components are identified, and an overall architecture picture is provided. The section then further provides details of the vehicle software architecture and the message flows between the vehicle and the backend, consisting of public Internet and MEC. Then, aspects of QoS predictions are discussed for the ToD and HD Mapping Use Cases. This is followed by a description and discussion of security solutions for each use case. Finally, differences for a real-world architecture for each use case are described.

The real-world architecture sections are used to point out differences compared to the architecture described in the main part of the use case specific sections. Those main parts are considered to
be “trial architectures”. As long as not stated differently in the real-world architecture section, the trial architecture corresponds to the real-world one. For all use cases the real-world architecture contains a discussion about potential roles of providing and/or operating the components required for the use case. Those are different between trial architecture, where responsible project partners are in some cases not representative for the entities that would have this role in a real-world deployment. Sometimes simplifications were done in the trial architecture that might require adjustments for a commercial real-world deployment. Such simplifications are pointed out in the real-world architecture sections. The trial architectures are limited to describing what is needed to conduct use case trials. Sometimes several further options than what is trialled are known. In selected cases further important options that should be considered are then described in the real-world architecture. The ACCA Use Case real-world architecture discusses how it is embedded in a broader scenario of current and potential future Intelligent Transportation Systems (ITSs).
2 5GCroCo Use Cases

In this section, the three 5GCroCo use cases are briefly described. The section concludes with an overview table also including short descriptions of the related user stories. More details are provided in Deliverable D2.2 [1].

2.1 Tele-operated Driving

Current automated driving vehicle prototypes prove the feasibility of truly driverless cars. Tele-operated Driving (ToD) can be leveraged as an enabling technology to smooth this transition, as edge cases remain, which necessitates falling back on human operators. The overall architecture of ToD, which is considered, is shown in Figure 2-1. For ToD, an interface over the mobile 5G network is created that allows a human to remotely control a vehicle. Through such an interface, sensor and vehicle data, e.g., video feeds and velocity are transmitted from the vehicle to the vehicle control centre. There, the data are displayed to the human tele-operator who generates control commands, e.g., desired steering wheel angle or velocity. These are then transmitted back to the vehicle for execution. ToD technology faces a number of challenges that need to be overcome.

![Vehicle (AD & ToD Functions) 5G Network Vehicle Control Center](image)

Figure 2-1: Schematic Overview of the ToD Use Case

Reduced situational awareness poses one of the greatest challenges for ToD, as the tele-operator is not physically located in the vehicle. Additional mental effort is required to compensate for distortions and recreate missing information from the sensor data.

The transmission of signals over mobile networks introduces latency, which can be critical if the vehicle is remotely controlled at stabilization level, i.e., the teleoperator produces direct steering commands. If the latency is too large, different control concepts may be applied, such as an indirect, trajectory-based control scheme. Nevertheless, with 5G technology, nowadays limitations posed by network latency are subject to change.
2.2 High-definition Mapping

One of the cornerstones of autonomous driving is an accurate, actual, and seamless high definition map. The basic functionality is to determine the vehicle’s position — which road and which lane it is in — but also information about traffic rules like speed limitations, or more dynamic conditions like road closures or construction areas. High-definition (HD) map users expect a continuous availability of the map content, even in cross-border scenarios. Autonomous cars, however, require the map to be constantly up-to-date, and thus when reality changes, the map needs to be updated. Regular map updates by the map provider, typically done a few times a year by driving mapping vans along the roads, are not at all sufficient. To ensure a high reliability of autonomous cars, the map needs to be updated constantly, by as many contributing cars as possible. Broadly speaking, the cars collect information about their surroundings using their on-board sensors, and then use their connectivity to send this information to some backend. Here, the received data is compared to the existing map, and if differences are found, the map can be updated. The data might even come from sources other than cars, e.g., roadside cameras. The HD map can also be used as the base upon which more dynamic information can be stored, for example, accidents. All these procedures have to work seamlessly across borders. For example, map updates from cars on one side of a border have to be distributed also to cars on the other side, served by a different operator with the backend running on a different Mobile Edge Cloud/Computing (MEC) architecture.

A schematic overview of a user story of the HD Mapping use case is shown in Figure 2-2: A first vehicle, A, detects a change on the road, e.g., a lane that is closed because of construction work. It sends this deviation to a backend in the cloud through Mobile Network Operator 2 (MNO 2), where the new information is incorporated into the map. The updated map is then sent back to all connected vehicles in the vicinity.

![Figure 2-2: Schematic Overview of the HD Mapping Use Case](image-url)

Document: 5GCroCo/D3.1
Version: v1.0
Date: 2021-01-31
Status: Final
Dissemination Level: Public
2.3 Anticipated Cooperative Collision Avoidance

Towards the realization of autonomous vehicles, car manufacturers are adopting and developing sensors that allow vehicles to sense their environment and control the vehicles. Driving automation systems rely on a variety of sensors like cameras, radar, lidar, etc. Despite the increasing number of in-vehicle sensors, the environmental perception of the vehicle remains limited. In certain situations, typical stand-alone sensing systems will not be able to detect and localize dangerous events on the road with sufficient level of anticipations. In such situations, too late detection of a dangerous event will trigger a hard braking or a dangerous manoeuvre or potentially lead to a collision.

The Anticipated Cooperative Collision Avoidance (ACCA) use case relates to the possibility to anticipate certain potentially critical events in order to reduce the probability of collisions in situations when typical sensors will have no visibility or a short detection range (e.g., a few 100 m). The aim of the ACCA use case is to induce smoother and more homogeneous vehicle reactions by facilitating the anticipated detection and localization of temporarily static events such as traffic jams, high deceleration, emergency braking or unexpected manoeuvres of vehicles in front, etc. An example of a typical scenario of the ACCA use case is depicted in Figure 2-3.

![Figure 2-3: Typical Scenario of the ACCA Use Case where an Abnormal Situation cannot be Detected by On-board Sensors and MNOs From Two Different Countries are Used for Communication](image)

2.4 User Story Overview

The particular and specific implementation of each of the use cases for the tests is referred to as User Story in 5GCroCo. Table 2.1 provides a list and brief description of the different user stories that will be implemented in the tests and trials of 5GCroCo.
### Table 2.1: 5GCroCo User Story Overview

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Short Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ToD</strong></td>
<td>User Story 1 - Remotely Controlled Manoeuvring</td>
<td>The vehicle follows the commands (steering, braking, acceleration) it receives from the tele-operator.</td>
</tr>
<tr>
<td></td>
<td>User Story 2 - Remotely Controlled Path-based Driving</td>
<td>The tele-operator provides a path, which will be tracked by the vehicle.</td>
</tr>
<tr>
<td></td>
<td>User Story 3 - Remotely Supervised Control</td>
<td>The vehicle is not controlled remotely but driven by a trained safety driver. Transmission of data, uplink and downlink, will be similar to US#1, but the control commands will not be forwarded to the vehicle actuators.</td>
</tr>
<tr>
<td></td>
<td>User Story 4 – Slim Uplink for ToD</td>
<td>Same as User Story 2, but with the aim to significantly lower the data throughput.</td>
</tr>
<tr>
<td><strong>HD Mapping</strong></td>
<td>User Story 1 – Map Download</td>
<td>Download HD map data for a map tile before entering that tile.</td>
</tr>
<tr>
<td></td>
<td>User Story 2 – Map Deviation Upload</td>
<td>Upload changes to the map detected by on-board sensors to the HD map content provider.</td>
</tr>
<tr>
<td></td>
<td>User Story 3 - Map Deviation Upload Followed by Download</td>
<td>One vehicle executes User Story 2 and a following one executes User Story 1 to download the freshly updated HD map data.</td>
</tr>
<tr>
<td></td>
<td>User Story 4 – Map Download with Quality of Service Prediction</td>
<td>Same as User Story 1 but the best place/time to conduct the download is predicted.</td>
</tr>
<tr>
<td><strong>ACCA</strong></td>
<td>User Story 1 - Ego Detected Event</td>
<td>A stationary vehicle imposes a hazard, detects this itself and sends this information to the backend so others can be warned.</td>
</tr>
<tr>
<td></td>
<td>User Story 2 - Remotely Detected Event</td>
<td>Same as User Story 1, but either other vehicles passing by detect the hazard or a detection algorithm in the backend detects it by analysing information coming from the stationary vehicle imposing a hazard.</td>
</tr>
<tr>
<td></td>
<td>User Story 3 - Traffic Jam Detection</td>
<td>Same as User Story 2, but the hazard is not imposed by a single stationary vehicle but a traffic jam consisting of many vehicles.</td>
</tr>
</tbody>
</table>
3 Detailed Application Architecture Description

This section provides detailed descriptions of the application architecture of the vehicle and the backend side for the three 5GCroCo use cases ToD, HD mapping, and ACCA. Overview architecture figures for each use case clarify the implementation responsibilities of the involved partners and necessary collaborations due to interfaces between components or joined development of one component.

3.1 Use Case 1: ToD

Tele-operated Driving (ToD) is defined as remote control of automated vehicles by a human over mobile radio networks within one country and across borders. It is meant to complement Automated Driving (AD) by bringing the tele-operator, located in the Vehicle Control Center (VCoC), into the control loop in situations an automated vehicle cannot handle [1]. Within 5GCroCo, ToD is trialled with two vehicles, one each provided by BOSCH and Volkswagen AG. Both vehicles have integrated longitudinal and lateral control, which, based on input from vehicle sensors, can automatically steer the vehicle on defined lanes, hand over control to a tele-operator and take back control as well. Due to legal and safety reasons, a human safety driver is required at all times to take over control if needed. The Technical University of Munich (TUM) and by BOSCH both provide one implementation of a VCoC.

In this section, a detailed description of the final application architecture of the ToD Use Case, as realized within 5GCroCo, is provided. A description of the overall architecture, the main components of the use case and the two considered control concepts are provided in Section 3.1.1. The interface between the vehicle and the VCoC is described in Section 3.1.2. The detailed component architecture is given in Section 3.1.3. The procedures taking place during a complete ToD session are described in Section 3.1.4. Details on the usage of predictive Quality of Service (pQoS) for ToD are provided in Section 3.1.5. Considerations of security aspects and deployment of ToD in the real world are given in Section 3.1.6 and Section 3.1.7, respectively.

3.1.1 Overall Trial Architecture and Main Components

The considered high level architecture for the ToD Use Case is shown in Figure 3-1. The vehicle, provided with automated and ToD driving functions, is transmitting data to (uplink) and receiving data (downlink) from the tele-operator in the VCoC via the mobile network.
Figure 3-1: Overall Component Architecture for Tele-operated Driving

Uplink signals to the VCoC have the purpose of informing the tele-operator about the current status of the vehicle and need for tele-operation support. An equivalent downlink signal informs the vehicle about the availability of the latter. During the tele-operation, a continuous exchange of sensor, mainly video, and vehicle data in uplink, and control commands in downlink is taking place.

Within 5GCroCo, two control concepts are being evaluated in the ToD Use Case. For direct control, the tele-operator is generating control commands, which consist of the desired steering wheel position, and the desired velocity or acceleration. These are transmitted at a high rate and the tele-operator is controlling the vehicle directly at stabilization level. The vehicle control loop is closed via the mobile network. With the indirect control concept, the control loop is closed in the vehicle. The operator’s task is to generate desired waypoints, forming a path. On completion, the path is transmitted to the vehicle. It is then the role of the path tracking controller of the vehicle to compute the steering wheel angle and to command the actuators of the vehicle, in order to follow the desired path.

While the architecture developed here is designed and implemented for use in the 5GCroCo trials, we also expect it to be comparably close to a real-world architecture. Details on this are discussed in the section 3.1.7 Real-world Architecture.

### 3.1.2 Interface between Vehicle and VCoC

Figure 3-2 illustrates the interface between the vehicle and the VCoC which contains several signals. All signals marked in orange are sent from the vehicle to the VCoC and all yellow signals in the opposite direction. These signals are described in Sections 3.1.2.1 and 3.1.2.2.

---

1 The BOSCH vehicle receives the desired velocity. The Volkswagen AG receives the desired acceleration.
In order to develop a scalable and versatile solution, the Collective Perception Message, which is currently in the standardization process at European Telecommunications Standards Institute (ETSI)-ITS [4], was used as a reference point for the development of the signals. This is to facilitate cross-vendor and cross-Original Equipment Manufacturer (OEM) solution by reusing already defined signals. Otherwise, there is a risk of creating unnecessary specification and implementation effort for multiple, incompatible interfaces and promoting vendor-dependent approaches that prevent interoperability.

The interface is defined using Protocol Buffers (Protobuf) [5] as common cross vehicle and VCoC interface. It was chosen because it is very suitable for specifying the required interfaces, uses small amounts of data for transmission, and offers good support from development tools.

![Figure 3-2: Interface between Vehicle and VCoC](image)

An example of a message declaration with Protobuf is shown below. This message, `msgGpsData`, defines the signals and datatypes for transmitting and receiving Global Positioning System (GPS) positions. It is used in the two vehicles provided by Volkswagen AG respectively BOSCH vehicle as well in the two VCoC provided by TUM respectively BOSCH.

```protobuf
define message msgGpsData {
  int64 i64_gpsTime = 1; /*Time stamp of the ego vehicle in ms*/
  double d_gpsPositionLatitude = 2; /*(Global) Latitude wgs84 (north) decimal degree*/
  double d_gpsPositionLongitude = 3; /*(Global) Longitude wgs84 (east) decimal degree*/
  double d_gpsPositionAltitude = 4; /*(Global) Altitude wgs84 (height) meter*/
}
```
The messages are distributed using Message Queueing Telemetry Transport (MQTT), which itself is transmitted via Transport Control Protocol/Internet Protocol (TCP/IP). The video streams from the vehicle to the VCoC are realized via GStreamer\textsuperscript{2} using the H.264 Codec\textsuperscript{3}. The compressed video data are then transmitted via User Datagram Protocol (UDP), using the Real-time Transport Protocol (RTP) and the Real-time Streaming Protocol (RTSP).

3.1.2.1 Uplink

In the following, we give a brief description of the used messages sent from the vehicle to the VCoC in the uplink.

Vehicle Registration

The vehicle registration message is used by the remote vehicle to provide basic information to the VCoC. It contains the Intelligent Transportation System (ITS) station type to define its role and provide the vehicle dimensions. This allows the VCoC to determine the type of transportation vehicle using the same approach as seen on Section A.78 of the ETSI TS 102 894-2 V1.3.1 (2018-08) \cite{6} documentation. The VCoC application then can render the remote operator human interface accordingly.

Vehicle Status

The vehicle status message contains the actual state of the ToD application. We distinguish the following two states: automated and teleoperated. This allows the VCoC to determine if the controlled vehicle has the state “teleoperated”, meaning that the VCoC is controlling the automated vehicle.

Ego Vehicle Data

This message contains the vehicle ego data sent by vehicle to the VCoC. Depending on the ITS station type and the vehicle capabilities the elements of this message change. It contains relevant vehicle status information, such as: indicators, horn, wipers, gear position, among other data relevant in a tele-operation session.

Position and Odometry Data

The position and odometry data of the tele-operated vehicle allows the VCoC to track the movement of the controlled vehicle. The VCoC is monitoring the system while performing direct or indirect control. Particularly the odometry data aggregates specific details on the displacement of the automated vehicle position over time, such as: the longitudinal and vertical acceleration.

Object Data

The tele-operated vehicle perceives objects using its sensors. The resulting data is a list of the relevant objects detected including information about the dynamic state and properties of the


detected objects. For the composition of the object data message is followed the approach that the ITS-CPM message is addressing and getting standardized [7]. The automated vehicle then sends the list of perceived objects to the VCoC. In User Story 4 (see Table 2.1, and [1]) this information is used at the VCoC.

**Sensor Stream Data**

The sensor stream data provides the VCoC with all video streams and their individual configurations. It contains the framerate, bitrate, camera state and more information regarding the cameras used in the automated vehicles. The video streams are then transmitted with GStreamer pipelines and the RTP/RTSP protocols to the VCoC. In trials, the H.264 Codec is usually parametrized as follows. To minimize delays in the video compression, the “tune” and the “preset” are set to “zerolatency” and “ultrafast”, respectively. Furthermore, to distribute the refresh of the key frame across multiple frames, the “intra-refresh” option is enabled. The aforementioned parameters are set in the “x264enc” module\(^4\) of the GStreamer pipeline. Other video stream parameters such as the framerate, bitrate or resolution are different across the vehicles and camera models and are adjusted dynamically at runtime.

### 3.1.2.2 Downlink

In the following, a description of the messages, sent from the VCoC to the vehicle in the downlink, are given.

**Operator Status**

The operator status message contains the ToD session information and state of the ToD application from the VCoC perspective. For instance, unique identifiers of the ToD session and operator are contained in the message. Furthermore, the tele-operator can specify the readiness, i.e., all vehicle and sensor data are received in the VCoC correctly. Also, a flag to terminate the ongoing ToD session is included. This is further elaborated in Section 3.1.4.3.

**Control Commands**

Depending on the control mode, either direct or indirect control commands are transmitted from the VCoC to the vehicle.

*Direct Control Commands:* This message contains the commands when the vehicle is controlled in direct control mode. For instance, there are the values of the desired steering wheel position and the vehicle velocity or acceleration. In addition, secondary actions such as wipers, direction indicators or the horn can be requested from the tele-operated vehicle.

*Indirect Control Commands:* This message contains the commands when the vehicle is controlled in indirect control mode. It holds a list of desired waypoints that are specified by the tele-operator. The vehicle receives these, performs collision checks and then tries to follow the waypoints as accurately as possible.

---

3.1.3 Detailed Component Architectures

Figure 3-3 shows the detailed component architectures of vehicle and backend with the network in between. Besides the individual components themselves, also responsible partners are indicated by small, coloured squares. Components only present in one vehicle, either the BOSCH or Volkswagen AG one, are indicated with a small circle in the respective colour.

![Figure 3-3: Detailed Component Architecture for ToD Components (Legend in Appendix B)](image)

Figure 3-4 provides the simplified architecture, focusing on the communication between applications. All information, on how the connections between them are realized, is omitted.
In the following, the architecture and interaction of the components will be described in more detail.

### 3.1.3.1 Vehicle

The Communication Module in the vehicle, as shown in the above Figure 3-3, manages the connectivity to the network as the basis for the communication of the in-vehicle services with the backend. That includes the initiation, adaptation and continuous monitoring of the connection status. The Communication Control Unit (CCU), as main part of the communication module, manages multiple physical communication channels based on several technologies to allow an adaptation of the connectivity properties. Supported communication technologies within 5GCroCo are 4G, 5G and WiFi. The used technologies can be changed “on the fly” to have an alternative communication channel during test execution, e.g. 4G and WiFi as backup on private test sites. Also, the characteristics of the different technologies like throughput, latency and reliability can be tested.

The BOSCH Advanced Driving Assist System (ADAS)-V2X hybrid CCU, connected via BroadR-Reach [8] to the in-car components, is an embedded hybrid communication device designed to fulfil the automotive requirements. It contains several radio modules to cover 4G (Long Term Evolution (LTE)), 5G (New Radio), ITS-G5 (Dedicated Short Range Communication (DSRC)), C-
V2X (PC5), WiFi and Bluetooth technology. Further communication devices can be connected via Universal Serial Bus (USB) or BroadR-Reach. Two different radio technologies are projected to be used for the ToD Use Case, 4G (LTE) and 5G (New Radio). The CCU and the currently used communication technology can be controlled with Protobuf messages, transmitted via MQTT. Additionally, a web interface is provided by the CCU to control the functionality using the same Protobuf messages. The preferred browsers to use this interface are Firefox or Chrome.

The **4G Modem** is already included in the CCU and covers the LTE bands B1, B3, B5, B7, B8, B20, B28A, B32 and B38. The User Equipment (UE) is specified as Category (CAT).13 in uplink and CAT.16 in downlink direction (Third Generation Partnership Project (3GPP) Release 12 [9]). Wideband Code Division Multiple Access (WCDMA) and Global System for Mobile Communications (GSM) air interface standards are also supported but not intended to be used in the project.

The **5G New Radio Modem** shall be used as main connectivity option during the ToD Use Case. Because of missing suitable 5G CCUs on the market, the BOSCH hybrid CCU has been equipped with a 5G communication module. The goal was to design a CCU which is very close to a future product. Based on market analysis and user experiences the RM500Q 5G module from Quectel [10] has been selected which was already successfully tested in the Ericsson 5G radio area network, see Deliverable D4.1 v3 [11]. The 5G module with M.2 interface is connected via Peripheral Component Interface Express (PCIe) bus to the CCU and supports the 5G New Radio bands n41, n77, n78, n79 in non-standalone mode. Alternatively, an external 5G modem or router can be connected via the external USB interface as done during the Munich test trial with Huawei equipment.

The **Antenna Array** connected to the CCU as shown in Figure 3-3 above consists of five multiband diversity Multiple Input Multiple Output (MIMO) antennas. All antenna housings are magnet mount to be flexible in placing it on the vehicle roof. The total available antenna count and types are listed in Table 3-1. Five 5G New Radio and one Global Navigation Satellite System (GNSS) antennas are currently unused.

**Table 3-1: Antenna Specifications**

<table>
<thead>
<tr>
<th>Count</th>
<th>Antenna Type</th>
<th>Frequency and Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Global 4G (LTE)</td>
<td>694-960 MHz, 3 dBi &amp; 1.7-3.7 GHz, 4 dBi</td>
</tr>
<tr>
<td>9</td>
<td>Global 5G (New Radio):</td>
<td>2.1-2.7 &amp; 3.3-3.8 GHz, 4 dBi</td>
</tr>
<tr>
<td>2</td>
<td>WiFi</td>
<td>2.4-2.5 &amp; 4.9-6.0 GHz, 5 dBi</td>
</tr>
<tr>
<td>2</td>
<td>GNSS</td>
<td>1575.42 +/- 2 MHz, 26 dB, 5 dBi</td>
</tr>
</tbody>
</table>
The **GNSS** module as part of the CCU provides positioning information for the Prediction Function (PF) Client as well as a very high precision synchronization of the CCU internal system clock. The Pulse-per-second (PPS) trigger allows using the clock for one-way latency measurement.

The **PF Client** on the CCU is tasked with requesting a QoS prediction from the Prediction Function Application Server (AS) and making it available to other components in the vehicle. Therefore, it has to obtain the planned route from another component in the vehicle. Once a prediction is received, it is then sent to the Car Personal Computer (PC) to be available for all components there. Figure 3-4 displays the connection of the PF Client and the Car PC. Triggers of the prediction are given in section 3.1.5.1 and message contents of the prediction are given in Deliverable D3.2 [3].

Furthermore, the PF Client can make CCU measurements available to the prediction function in order to improve the prediction. This could include signal strength measurements, as well as observed data rates.

Predictive Quality of Service (pQoS) Usage (Car PC): On the Car PC, pQoS Usage is not a single component but part of all components which adapt their behaviour depending on the predicted QoS. Especially the component responsible for the configuration of the video stream requires an according interface. Additional adaptations of the ToD function besides the change of video stream parameters, e.g., change of the vehicle speed, are possible. However, these have not yet been included at this point. More details on pQoS for ToD are provided in Section 3.1.5.

As shown in Figure 3-4, the vehicle hosts an **RTSP Server** for video streaming. This server allows the RTSP client in the VCoC to request video streams and start the transmission.

The **VCoC Client** is implemented on the Car PC and enables the connection to the VCoC Services. This connection is established via MQTT Protocol. The VCoC is further described in section 3.1.3.2.2.

### 3.1.3.1.1 Volkswagen AG Vehicle

The following components are only allocated in the Volkswagen AG vehicle.

**Radar**: Volkswagen AG implementation pass the radar information via the Ethernet switch that directly feeds information to the ToD and AD functions.

**AD and Safety modules**: Volkswagen AG provides an AD unit equipped with the corresponding hardware and software elements dedicated to implement safety services. Such services are part of the core functionality of the vehicle and are in charge to keep control of the actuators and the resulting manoeuvres.

### 3.1.3.1.2 BOSCH Vehicle

The following components are only allocated in the BOSCH vehicle.

---

5Under the condition that the same high precision clock synchronization is available in the backend
Radar Sensor and Positioning: These components are different compared to the Volkswagen AG setup. For the radar sensor, the main difference is the interface used to communicate with the Car PC component. In BOSCH case, this interface is BroadR-Reach and thus the data must be transferred via the Ethernet to a BroadR-Reach Router to the Car PC. For the positioning component on BOSCH side, Ethernet is used as interface via which it is connected over the Ethernet switch to the Car PC.

AD Function: On BOSCH side, the AD functionality will be realised on the Car PC while the AD function on Volkswagen AG side is hosted on a separate component.

Safety Gateway: The connection to the actuators (Brake, Engine, Steering) is decoupled from the Car PC component via a so-called Safety Gateway. This component cares for safety treatments of the actuation commands coming from the AD component on the Car PC as actuation might cause safety relevant activity of the car.

Human Machine Interface (HMI): The BOSCH vehicle allows the safety driver to control the vehicle via the steering wheel and the pedals. This unit will be used as HMI to the safety driver and the Safety Gateway will take this input with a higher priority than the AD or ToD requests. The vehicle cockpit unit will display the safety driver the status of the vehicle.

3.1.3.2 Backend
In this section, the two system components in the backend, pQoS and the VCoC, are described.

3.1.3.2.1 pQoS
According to Deliverable D3.2 [3], different deployment options could be considered, regarding the place where the PF (i.e., generator of prediction) could be hosted.

Application Function (AF): The AF interacts with the network to request and/or receive QoS prediction / network information. The PF can be also placed at the AF or receive QoS prediction by the network (e.g., according to the 3GPP architecture, the Network Data Analytics Function (NWDAF) has the PF role and the AF receives QoS prediction via N33 interface). The AF can be also MEC-deployed and serves as interface towards the network.

AS: The AS can be hosted in the MEC or public Internet. Therefore, the MEC one is marked as optional. It interacts with the AF to request and/or receive QoS prediction. If not provided directly to the AF, the AS can also provide measurements from the vehicle to it. The AS can be accessed from the vehicle and/or the VCoC.

3.1.3.2.2 VCoC
In the following, the components of the VCoC shown in Figure 3-3 are described.

VCoC Service with MQTT Broker and RTSP Clients: The VCoC service provides the interface to the network, and thus the vehicle. It is transmitting and receiving data to and from the vehicle via the MQTT broker. The video streams from the vehicle are received by the RTSP clients, connecting to the RTSP server in the vehicle. All received data is being passed on to the output
device. Partially some processing of the data is required. For instance, the video data needs to be decoded before they can be displayed.

**Output Device:** The output device displays the vehicle and sensor data that is received in the VCoC to the tele-operator. It is a set of three monitors, mounted in a circular shape, as shown in the picture in Figure 3-5. Some aspects of the interface for the operator are also visible. On the left, there is the GUI, which is provided to set the operator state. Furthermore, the video streams, as are transmitted from the vehicle, are shown.

![Figure 3-5: VCoC Setup (Note: Not visible, but there is a total of three monitors)](image)

**Input Device:** The VCoC input device is used by the tele-operator to generate the control commands. In the case of the direct control concept, this is a steering wheel with pedals. For the indirect control concept, a mouse is used to specify the desired waypoints.
### 3.1.4 Tele-operated Driving Session

Figure 3-6, taken from Deliverable D2.2 [1], provides an overview of the time schedule of a ToD session between the AD vehicle and the VCoC for User Story 1 (see Table 2.1). Furthermore, the timeline is divided into three phases: initiation, execution and termination process.

![Figure 3-6: Time Schedule of Use Case 1 in User Story 1](image)

#### 3.1.4.1 Initiation phase

The initiation phase of a ToD session takes place between $t_1$ and $t_3$, as depicted in Figure 3-7. It starts after the vehicle detects a traffic situation it cannot resolve on its own. Once the vehicle stands still the initiation phase commences.

The phase starts in $t_1$ by requesting to the CCU module to establish a communication link between the vehicle and the VCoC. Upon the establishment of the communication link over the 5G network, the VCoC receives the vehicle registration information. This initial step allows the VCoC to determine the ITS station type of vehicle requesting ToD support. VCoC acknowledges the request of the vehicle by assigning a tele-operator to it. Further exchange of data permits the tele-operator to know the configuration of the vehicle, e.g., obtaining the number of cameras accessible by the VCoC or the vehicle position.

This data allows the construction of a Graphical User Interface (GUI) for the tele-operator to display the vehicle’s environment.

At point $t_2$, the tele-operator identifies the vehicle state, ego data, position and odometry information via its GUI. In addition, the tele-operator receives the different video streams provided by the multiple vehicle cameras.

The vehicle receives the assignment of a ToD session and a corresponding tele-operator from the VCoC. The tele-operator then analyses the border case situation of the vehicle and selects the control method to be executed, i.e. direct control (User Story 1 and 3) or indirect control (User Story 2 and 4). At $t_3$ the tele-operator selects the control method and transmits it to the vehicle.
27

Figure 3-7: Initiation Phase (t₁ to t₃)

3.1.4.2 Execute

The execution phase starts when the teleoperator informs the vehicle through the VCoC about the selected type of control method. Two control modes are considered: direct or indirect control. In addition, two approaches for the transmission modes of the environment of the vehicle are considered: video streaming (User Story 1 to 3) or Slim Uplink (User Story 4) [1].

After receiving the control type at time t₃ the vehicle state is set to “active” and control is handed over to the operator in the VCoC, as depicted in Figure 3-8.

From this point onwards, the VCoC transmits to the vehicle the respective control and operation state messages. The vehicle is sending its actual state, position, velocity, etc. to the VCoC. Thus, both together form a data exchange loop. The termination phase starts at t₄, when the loop is interrupted by one of the conditions, as described in the Section 3.1.4.3.

Across the entire time period between t₃ and t₄, both entities are able to receive and verify the network conditions. During the execution phase, the entire system imposes strict requirements on the communication. High reliability, low latency and a minimum throughput is required. However, vehicle is able to deal with a predicted variation of the network conditions by adapting the configurations within certain limits. An application adaptation depending on the pQoS therefore helps to avoid interruptions during the ToD session.
Direct Control Mode

This control method directly steers the vehicle actuators. The message contains values of the desired steering wheel angle position and the vehicle acceleration (Volkswagen AG vehicle) or velocity (BOSCH vehicle). In addition, secondary actions can be realized by the tele-operated vehicle such as wipers, direction indicators or the horn. This allows the VCoC to help the vehicle to overcome border case situations.

In general, during the execution of this type of control method, the tele-operator needs to maintain all safety precautions. However, certain safety features in the vehicle remain active during the ToD session.

Indirect Control Mode

This control method is defined by the transmission of a preselected vehicle path designated by the tele-operator. Indirect control refers to no direct lateral nor longitudinal control such as steering angles. The VCoC instead provides a series of way points to the vehicle. The routing and execution of the path is done by the vehicle itself.

3.1.4.3 Terminate

The last phase in the ToD session is the termination phase. Figure 3-9 shows the termination phase, which takes place between $t_4$ and $t_5$. We investigate two triggers to start the termination phase.
phase. They are selected based on the origin of the condition and lead to disengagement of the ToD session.

**Termination based on the VCoC conditions**

A termination of the ToD session based on the VCoC conditions is performed when the tele-operator determines that the situation requiring ToD is solved. When this occurs, the tele-operator transmits the request to the vehicle to terminate the ToD session; with this information the vehicle initiates the sequence to disengage the communication link with the VCoC. The termination phase ends when the vehicle ends the communication with the VCoC and is able to continue its ride automated.

**Termination based on the vehicle conditions**

A Termination based on the vehicle conditions is performed when the vehicle is able to detect one of the two following kinds of conditions. The first one is related to safety aspects; the condition in this case is the interaction of the safety driver with the primary actuators in the vehicle, such as steering wheel, vehicle pedals or the hand brake. This interaction is captured by the safety module, as shown in Figure 2.5. The vehicle requests the VCoC to terminate and proceed to close the communication link.

The second condition is based on the monitoring of the VCoC communication. The ToD application monitors the reception of all messages sent by the vehicle. Whenever any messages exceed a time out limit, the ToD session is terminated. This functionality is indicated in Figure 3-9 via the green boxes.
3.1.5 Predictive Quality of Service

In the following, a description is given of triggers how the QoS predictions should be supplied to the vehicle from an application point of view. Furthermore, concepts for application adaptations are presented. The content of prediction messages exchanged between the PF Client and the PF (that is placed in the network and/or the AF), is given in Deliverable D3.2 [3]. Involved components are described in Section 3.1.3.2.1.

3.1.5.1 Trigger

The prediction shall be obtainable by a pull-mechanism, where the vehicle requests a prediction from the prediction function. This request is repeated regularly and contains the upcoming route up to a certain length. The length is depending on the duration for which a prediction shall be available, called “min. prediction duration” (specified in seconds). However, this pull-mechanism shall be combined with updates, pushed by the prediction function in case the prediction changes. Necessary messages are “prediction request” and “prediction response”. The prediction response can either contain a complete prediction for the request or an update, in case the predicted QoS changes.

The PF Client has to assure that a QoS prediction is always available for at least “min. prediction duration”. Therefore, it has to request a new prediction when the remaining prediction is not long enough to cover this duration of the upcoming route. Requesting a new prediction can also be triggered by a change of the vehicle’s route, for which no prediction was obtained so far.
Different user stories and scenarios might require different values for the “min. prediction duration”. For example, ToD on a highway at high speeds requires a high duration for the available prediction, since finding a place to safely stop the vehicle when predicted QoS is not sufficient anymore will be difficult. Section 3.1.5.3 gives an estimation of the required duration of the prediction.

When updates for the prediction shall be received, threshold values have to be defined. This determines whether a change in the QoS prediction, which might occur frequently, is actually communicated or not. For example, a threshold of 10 Mbit/s for downlink data rates means that updates for predictions of this parameter are only communicated if the current or the updated data rate prediction is below 10 Mbit/s. These thresholds could also be used to determine whether an additional prediction for an alternative route shall be requested. If the current prediction shows entries where the latencies are above the latency threshold, the Packet Error Rate (PER) are above the PER threshold or the data rates are below the data rate threshold, the operator could decide based on alternative routes whether he wants to re-route the vehicle.

### 3.1.5.2 Concepts for Possible Reactions

Depending on the QoS prediction response, different reactions could be triggered at the ToD application and/or the vehicle side in order to allow the efficient operation of the ToD service, regardless of a potential QoS degradation. The decision whether a reaction has to be made is determined based on the QoS prediction and the current driving situation. In the following, two concepts for reactions are given. If one of them is activated, a notification should be transmitted from the vehicle to the VCoC to inform the operator about the changed behaviour. Further notifications have to be supplied to the operator in case the prediction will deteriorate too strong to continue ToD. In this case, it is the operator’s task to select an appropriate counter measure. For example, he could change the route of the vehicle, if an alternative with suitable connectivity is available.

**Reduce video quality:** If the predicted minimum uplink data rate is not high enough to transmit video at the current quality, the video quality has to be reduced. This can be done for example by reducing the resolution, reducing the amount of transmitting video cameras or increasing the compression. Alternatively, or in addition, the operator could prioritize the different camera streams. Potentially, any reduction in video quality has to be done together with a reduction of the maximum vehicle speed. In this project, only the resolution and the bitrate will be reduced for the adaptation.

**Reduce maximum vehicle speed:** If the predicted QoS is not high enough to allow the maximum ToD speed, the maximum speed has to be reduced. Whether this is necessary depends in general on the maximum latency (uplink and downlink), the available data rate (uplink and downlink), the Packet Error Rate as well as the video quality. In the scope of this project only the main influencing factor, the predicted latency, will be used to limit the vehicle’s speed.
3.1.5.3 Estimated Required Prediction Duration

The required prediction duration defines, how long predictions must be known in advance by the vehicle in order to achieve smooth adaptations when QoS changes. Influencing factors for this time depend on the considered adaptation, so we structure the influences according to the adaptation.

For the “reduce video quality” adaptation, the values depend on the used video codec, its implementation and its parametrization. Based on observations with the current state of the implementation, the duration until an adaptation of the bitrate or the resolution takes effect is less than one second. Therefore, the required duration for predictions in this case is one or very few seconds. The reliability of the prediction must be very high, in order to avoid dropped video frames. In addition to reducing the video quality, a reduction of the number of cameras is considered above. But this does not require an additional prediction duration, since stopping the transmission of a video feed should be possible instantaneously.

When considering the adaptation “reduce vehicle speed”, the main influence for the duration of the adaptation is the deceleration. In literature, different deceleration values which are deemed comfortable can be found. One of the lowest values is 2.0 m/s² [12], which will be used for the consideration here. If we assume a maximum ToD speed of 80 km/h and, in the most extreme case, a reduction down to 5 km/h, this leads to an estimated duration of the adaptation of about 10 seconds. So, for this adaptation, a prediction of latencies with high reliability is required for at least 10 seconds in advance.

Overall, considering these two adaptations, the maximum required duration for predictions is expected to be in the order of 10 seconds.

3.1.6 Security

Below, security measures applied between different components of the TOD Use Case are described. It is not intended to enable them all during trials as this complicates result analysis. Instead, it will be inspected which security measures can have an impact on measured Key Performance Indicators (KPIs), especially the Application Level Latency and tests will be conducted to quantify this impact. But even those results will have to be taken with care as the prototypes used might not benefit from hardware acceleration typically applied in this field.

In the communication stack for the TOD Use Case different protocols are used for the interface between vehicle and VCoC.

MQTT over TCP is used to distribute the general messages up- and downlink communication as described in section 3.1.2. This allows to use security measures that are part of the MQTT specifications and ones that can be applied with TCP, namely Transport Layer Security (TLS). A comprehensive description of these measures is provided in the security section of Deliverable D3.2 [3]. In the project MQTT authentication with unique username and password per vehicle is intended to be used to connect to an MQTT broker. Furthermore, TLS as a further security layer is available.
The video streams from the vehicle to the VCoC are realized via GStreamer using the H264 Codec. Transmission is implemented using the three protocols RTSP, RTP and Real-time Transport Control Protocol (RTCP) via UDP. Here the appropriate security measures are described in RFC2326 [13] for RTSP, RFC3711 [14] for RTP and RFC3550 [15] for RTCP. First, as defined in RFC2326, unicast can be used such that the video is only streamed to one destination. This receiver can only be set by a request sent through an encrypted and authenticated MQTT message. This way it becomes impossible for third parties to request video streams. Second, the communication channels can be encrypted. For RTSP the appropriate method is also defined in RFC2326. For RTP the encrypted variant called Secure Real-time Transport Protocol (SRTP) and for RTCP the encrypted variant with the same name can be used, which are defined in RFC3711 respectively RFC3550. The relevant algorithms are available as part of GStreamer. With this approach, all parts of the video data streams can be protected from third party access, also called man in the middle attacks.

### 3.1.7 Real-world Architecture

Previous sections provide an architecture consisting of functions and components jointly enabling them. Within 5GCroCo, this allows to demonstrate the ToD Use Case and evaluate KPI performance in trials. The systems used might not be present in a real-world deployment of the use case. In this section, these differences will be explained.

The architecture of a real-world deployment of the ToD Use Case is shown in Figure 3-10. A key difference is that in reality the VCoC for ToD would not be provided by an academic institution. Depending on the business model, it is more realistic that instead an OEM would deploy its own VCoC. Also conceivable is that another service provider like a city or specialized companies, would serve vehicles with ToD in certain areas. In general, a real-world setting would involve many more than only two OEMs as it is the case within 5GCroCo. Also, in contrast to maximum performance specifications of the ToD system, it is expected that a real-world system setup will aim more at cost saving components for series production. Finally, another significant difference when deploying ToD will be the consideration of safety. Within the 5GCroCo trials, there is always a safety driver located in the vehicle, ready to take over.

Regarding pQoS, an Over-the-Top (OTT) approach could be used in addition or as alternative to the network supported approach considered here. For a pure OTT solution without network support, the OTT PF receives measurements from the PF Client in the vehicle and can be used to obtain pQoS information. Further adaptations can be considered as well, like planning new routes if the predicted QoS is not sufficient.

The messages defined for teleoperation based on Protobuf are a good foundation towards standardization of a common OEM vehicle controller interface. Basing the signals used in this messaged on the signals defined for the Collective Perception Message, as currently in the standardization process at ETSI- ITS [4], will probably support the standardization. In particular, it is important that well thought-out definitions of these signals are available. In addition, the implementation effort for a manufacturer of a teleoperation system is reduced if a signal remains the same over several functions. However, currently the vehicle architectures are developing very
at a fast pace. Thus, it will have to be evaluated if this format is suitable to be used in interfaces that are expected to be used for longer periods, i.e. many years or even decades.

Figure 3-10: Real-world ToD Architecture
3.2 Use Case 2: HD Mapping

3.2.1 Overall Trial Architecture and Main Components
The architecture for realization of the HD Mapping Use Case contains two main blocks: in vehicle and backend deployed services. The backend deployed services are to a large extent deployed both in MEC and public Internet to compare performance. Figure 3-11 shows the implementation view of this setup.

![Diagram](image)

Figure 3-11: Components of the HD Mapping Use Case Trial Implementation Including Backend Deployment and Mapping to Responsible Partners (Legend in Appendix B)

3.2.2 Vehicle Side Architecture Components
This section contains more detailed descriptions of the vehicle components outlined in Figure 3-11 in Section 3.2.1.

3.2.2.1 Sensor Array
Sensors provide a semantic view of the vehicle surroundings.

3.2.2.2 Global Navigation Satellite System receiver
This is the sensor that receives GNSS signals to determine the vehicle position.

3.2.2.3 Prediction Function Client
The Prediction Function Client (PF Client) is responsible for retrieving predictions on network quality of service along the path the vehicle will likely travel as produced by the Map Tile Manager.
3.2.2.4 Detection Manager
The Detection Manager compares sensor readings from the Sensor Array with current map data and produces confirmed deviations (detections) when they differ from the downloaded map and provides them to the Uploader.

3.2.2.5 Static Detection Upload
The Static Detection Upload component provides support for triggering detection uploads also for cars without the sensor array, this component supports manual triggering. This is only used when performing trials while not connected to the car sensor array.

3.2.2.6 Uploader
The Uploader uploads geo-referenced differences in the HD map when advised to do this by the Detection Manager.

3.2.2.7 Map Tile Manager
The Map Tile Manager (see Figure 3-11 in Section 3.2.1) is responsible for keeping the set of currently needed map tiles in sync between the backend and the local Map Tile Cache. It uses the GNSS sensor to keep track of the cars progress and uses that to predict the likely path the car will travel. It also listens to notifications from the Map Update Tracker on when new map data is published. This component is responsible for planning the download sequence and timings based on information from the Prediction Function Client (PF Client).

A backend-initiated download flow (see Section 3.2.4.1) is triggered from the Map Update Tracker that keeps track of delivered tiles and tile updates pending for each individual car.

Client-initiated downloads use information from the PF Client but triggers for requesting it are changes in position.

3.2.2.8 Map Tile Cache
The Map Tile Cache provides local transient storage of map data and the interface to retrieve map data from the cache.

3.2.2.9 Map Tile Client
The Map Tile Client downloads new map tiles when it is needed based on planning from the Map Tile Manager.

3.2.2.10 Communication Control Unit
The Communication Control Unit provides network access to car applications.

3.2.3 Backend Side Architecture Components

3.2.3.1 Prediction Function – Application Function (MEC)
The PF AF (MEC) is a network-deployed service interacting with the network and potentially further components to calculate and provide the network information needed by the Prediction Function Application Server (PF AS) (described in Section 3.2.3.2) to satisfy the prediction
requests from the PF Client. Interfaces to the network and other components supporting the prediction are not yet fully agreed, as also described in Section 3.1.5 for ToD.

3.2.3.2 Prediction Function - Application Server
The Prediction Function Application Server (PF AS) interacts with the PF AF to provide the QoS prediction to the PF Client in the vehicle.

3.2.3.3 Map Update Tracker
The Map Update Tracker component keeps track of which versions of which map tiles that have been delivered to which vehicles and provides notifications to vehicles when there is a critical update for a specific tile.

3.2.3.4 Map Tile Server
The Map Tile Server (see Figure 3-11,3.2.1) provides map data to the vehicle Map Tile Client. It constructs and updates map data based on input both from the vehicles as well as other sources such as map suppliers.

3.2.4 Main Application Flows
This section describes the main application flows as message sequence charts with the components from the above sections.

3.2.4.1 Client-Initiated Download Flow
The primary mode for the use case is centered on downloading map data and measuring the client experienced network performance. The client consumes HD map data along the driven route for high precision positioning and navigation. The goal with this use case is to determine the benefits given by upgraded network infrastructure to the map download planning function in the car. With improved network performance, the client is given more freedom to plan when and/or where it downloads map content and can thus provide better performance to the functions (driver assists functions as well as Automated Driving) that rely on up-to-date map content. Figure 3-12 shows the client-initiated download flow where the Map Data Manager in the vehicle plans and executes map download based on different input decision criteria, e.g., where the vehicle travels.
Figure 3-12: Client-Initiated Download Flow

3.2.4.2 Backend-Initiated Download Flow

To support server-initiated map updates, the secondary flow enables the backend to notify the vehicle client about map tile updates for tiles that the client already has downloaded. This is described in the message sequence diagram in Figure 3-13.
The Sensor Array in the vehicle continuously scans the environment and feeds data into the Detection Manager which compares the perceived view with the one from the HD map data. If the Detection Manager finds a deviation, this is sent to the Uploader and from there to the Map Tile Server where it is processed and fused with other detections of the same deviation. The result is sent to the Map Update Tracker which determines whether updates to already delivered tiles are needed. This flow is described in Figure 3-14.

Figure 3-13: Backend-Initiated Map Download

3.2.4.3 Detection Upload Flow
The Sensor Array in the vehicle continuously scans the environment and feeds data into the Detection Manager which compares the perceived view with the one from the HD map data. If the Detection Manager finds a deviation, this is sent to the Uploader and from there to the Map Tile Server where it is processed and fused with other detections of the same deviation. The result is sent to the Map Update Tracker which determines whether updates to already delivered tiles are needed. This flow is described in Figure 3-14.
Predictive Quality of Service

pQoS is utilized in HD map download to provide the map manager component with input to its planning of download and upload activities. Planning in the map manager is done both to get optimal coverage of map data at the right point in time and pQoS will provide the network view of congestion and coverage on the road ahead. This will be used in User Story 4 (see Table 2.1, and Deliverable D2.2 [1]) for HD map content download time scheduling having two distinct objectives in mind: prediction of network throughput to conduct the download at suitable locations or prediction of cost to conduct the download where the Mobile Network Operator (MNO) would charge less. The motivation for the latter is that the MNO would suggest download locations where the network experiences low congestion and awards the user with lower cost if it follows this suggestion.

For pQoS, a PF AF is used to interact with network and potentially further components to provide the information needed by the HD map download application. pQoS interaction between vehicle and backend is realized by two components: one PF AS, deployed in public Internet or MEC, which interacts with the PF AF to request and retrieve the QoS prediction, and one vehicle deployed PF Client which communicate with the PF AS through a prediction Application Programming Interface (API) for retrieving predictions. The corresponding network architecture interfacing with the PF AF is provided in Deliverable D3.2 [3]. Its realization for the second round of trials will be provided with the results in Deliverable D4.3 [16].
3.2.5.1 Prediction API
On a high level the API is realized by the vehicle providing information about its planned route and in return receiving predicted downlink throughputs along the route (performance prediction) or a suggestion where to do the download (cost reduction, if suggestion is obeyed). For the trial implementation the vehicle would ideally provide a set of waypoints as GNSS latitude-longitude pairs together with estimated times of arrival. The HD Mapping Application Client does not have the required information for this, so a different approach will be taken. The Tile Horizon Length (see Deliverable D4.2 [17]) defines how many tiles ahead should be fetched. The vehicle will provide this list of tiles, according to application-internal representation of tiles, to the PF Application Server to request estimated throughputs or guidance where to do the download in order to benefit from reduced cost. The vehicle will also provide information about its average speed used to determine a rough and estimated time of arrival at each tile.

Section 3.2.7 provides information how this trial architecture could be evolved to a real-world one.

3.2.6 Security
To ensure Information Technology (IT)-security attributes Confidentiality, Integrity and Availability the communication solution for the trial is based on industry standard mechanisms:

- TLS 1.3 [18] is used for encrypting the data stream as well as providing authentication between server and client. Authentication is mutual with both map server and vehicle client providing credentials during handshake.

- X509 [19] certificates are used to provide identification of both server and client. The certificates are issued from a private Public Key Infrastructure (PKI) specific to the trial. Server and client will only accept counterpart certificates from this PKI.

To provide application logic the trial is using industry standard application protocols:

- Hypertext Transfer Protocol HTTP [20] is used for tile download, tile upload and Round-trip time (RTT) measurements.

- Websockets [21] is used for tile update notifications.

As the trial has not yet started, the evaluation or deployment of the pQoS function have not been performed, therefore no security mechanisms have yet been decided.

3.2.7 Real-world Architecture
This chapter contains a proposed deployment for a more production-like setup. The approach of course differs a lot between OEMs, some would for example use the map provider in the critical flow for uploads and downloads and others will not. It all depends on what level of control and strategy the OEM has. The component selection will also of course differ a lot depending on the complete vehicle and backend architecture and strategy of the OEM. Figure 3-15 shows how a possible deployment view of a more production like implementation could look.
Figure 3-15: Generic Application Architecture of the HD Mapping Use Case and Mapping to Possible Solution Providers

The security solution used in the trial implementation reflects the current state of art for this type of applications. The trial implementation uses a small private PKI setup, in a real-world implementation this would likely be a more upscaled industrial implementation. This will be further elaborated in Deliverable D3.3 [22].

Section 3.2.5 describes options for interfaces between the vehicle and backend for QoS prediction. These options are mostly motivated by the capabilities of the HD Mapping client application to indicate the driving direction. For the real-world architecture the interface described in Section 3.1.5 for ToD could also be applicable for HD Mapping for performance and/or cost prediction. In the following, mostly performance prediction is discussed but it is similarly applicable to prediction to reduce cost. The HD Mapping Use Case does not require such a precise spatial resolution as ToD and could therefore also apply an interface where the vehicle does not expose its path to the network. It might even not expose its position beyond information about its serving cell. With such an interface the network would inform the vehicle about expected performance in

---

6 The serving cell is known to the MNO serving the vehicle, but this information is normally not exposed to backend servers. Means to expose it to servers to which an according trust relation exists are possible. Alternatively, the vehicle can determine the serving cell ID through its CCU and provide it to backend servers it trusts with this information.
the current serving cell and potentially in surrounding cells. This would require solving the following challenges:

- Performance can even differ within the same cell as the achievable throughput decreases with distance from the cell tower. This can be compensated by providing more resources to the respective vehicle but would also mean to withdraw these resources from other vehicles.

- Without knowing where the vehicle is going the network would have to provide predictions for all surrounding cells with an indication about the area they serve. The vehicle would have to determine when a considered cell is serving it, but it can determine a set of candidate cells that will likely serve it next if it was information about cell locations. In some cases, even this might be difficult if different cells serve the same area e.g. on different frequency bands. The difficulties mentioned in the above bullet point also apply for surrounding cells that might also not be able to provide the same predicted capacity on the entire area covered by the corresponding cell.

Those challenges come on top of the general challenges of QoS prediction. The less prediction accuracy is needed, the easier they can be solved. The simplest case would be to predict where no acceptable communication is possible where “acceptable” can be defined by a lower bound throughput threshold that could be also set to zero. Event with that the HD Mapping application client learns that it might need to trigger a transmission now as later the network performance might be insufficient.

---

7 There are no clearly bounded serving areas for each cell as handover locations might differ among vehicles and even within the same vehicle.
3.3 Use Case 3: ACCA

In ACCA Use Case, there is a Geoservice which builds a real-time view of the road hazards and which delivers the information about the hazards which are on the trajectory of the vehicles. By this means, the vehicles can anticipate the risk of collision and the driver or the vehicle can take appropriate decisions to avoid accidents. It is described in detail in D2.1 [2] and D2.2 [1]. For this purpose, the vehicle interacts with a Geoservice which runs in the backend, usually hosted on MEC hosts or in the public Internet backends, when MEC hosts cannot be reached. Vehicles cooperate with the Geoservice to build the view of road hazards. When the Geoservice is hosted in a MEC host, it is done at local scale, according to an area served by the MEC host. The Geoservice instances and the Traffic Management Service (TMS) cooperate to build the view of road hazards at larger scale. This Geoservice delivers information about road hazards to the registered and authenticated vehicles which are interested by a given Region of Interest (ROI).

Besides just consuming the service, the vehicles also contribute to build this view of road hazards by sending the road events they have detected and their own status to the Geoservice. Moreover, dedicated applications, hosted in the backend as subcomponents of the Geoservice, challenges the confidence level of the road events, computes other potential road hazards and insert them into the view of road hazards.

The events of interest for ACCA are stationary vehicle, accident, and traffic condition.

Lastly, the received information about hazards in front of the vehicle is used by the vehicle Advanced Driving Assist System (ADAS) for anticipation and supervision means, e.g. adjusting the maximum speed or slowing the vehicle down.

3.3.1 Overall Trial Architecture

Figure 3-16 shows the application architecture, intended for the trials, in the vehicle and backend with the network connecting them.

The Geoservice in the backend is distributed between MEC hosts and public Internet. Besides functions which are essential for hazard warning, it also includes authentication and registration services.

Three different vehicle architectures are shown in Figure 3-17. The main difference are the partners responsible for the components with the top one being CTTC, the middle one RSA, and the bottom one PSA. PSA and RSA vehicles have the same components but will mostly differ in communication protocols they use, as further described in Section 3.3.2.1 and how the ADAS system uses the Hazard Notifications. However, for safety and security on the trials, the ADAS is not allowed to use this information to control the actuators on RSA cars. Instead, the hazard information is shown on the vehicle display. The CTTC vehicle implements both RSA and PSA protocol flavours, but vehicle sensors and actuators are emulated.

Interaction between components is shown in Figure 3-17. The components in the vehicle are:
- The Communication Control Unit (CCU) connecting to the network. It is integrated into the hosting Car PC for the CTTC vehicle and a separate physical component for the RSA and PSA vehicles,
- The Geoservice client interacting with the Geoservices in the backend,
- Detection and Trigger to detect hazards from sensor readings or from vehicle status (emulated in the CTTC vehicle),
- Sensors providing information to the Detection and Trigger component (not present in CTTC vehicle),
- Optionally ADAS System adjusting the speed based on Hazard Notifications (not present in CTTC vehicle) or showing an alert on the vehicle display.

Figure 3-16: ACCA Application Architecture in Vehicle and Backend (MEC and Public Internet) with Network in between
In the next sections, the message flows between the vehicles and the backend are described with high level of details. It highlights the differences between the architecture flavours used by the RSA and PSA vehicles. The interaction between different Geoservices on different MEC hosts with and without support of the TMS is described in more detail in Section 3.3.2.6.

In case the vehicle cannot reach a MEC-hosted Geoservice, a Geoservice hosted on the public Internet can be used, as further described in Section 3.3.2.1, in order to guarantee a continuity of the service with possibly degraded performances.

### 3.3.1.1 Hazard Reporting Message Flow and Interpretation

Vehicles can detect that they are hazardous thanks to their own vehicle status, e.g., when the vehicle is stopped, and the warning lights are turned on. In this case, a Hazard Report is sent to the Geoservice e.g. as Decentralized Environmental Notification Message (DENM) by RSA vehicles. The vehicles can also report their position and speed in CAM to the Geoservice. Additionally, the ADAS vehicle sensors can be used to detect dangerous events in their proximity and contribute to the collective perception of road hazards by sending CPMs to the Geoservice. The Geoservice hosts applications which interprets CAM and CPM information to detect hazards like a stationary vehicle or a traffic jam.
The Geoservice also processes the received information to determine if it applies to a new hazard, if it can be used to update the information of a previously detected hazard, e.g. by increasing the detection confidence or by improving the hazard location or if it is a duplicated information which is then discarded.

Figure 3-18: High-level Message Sequence Chart (MSC) Showing Hazard Reporting Message Flow with One Geoservice Involved

Figure 3-18 shows the message flow for Hazard Reporting in case only one Geoservice on one MEC host is involved. Vehicle 1 provides information about the detected hazard to the Geoservice. There, the information is used to determine if this is a previously unknown (new) event or a previously detected one. In this example, it is determined to be new. The TMS is informed about the hazard. Later, Vehicle 2 reports the same hazard. The Geoservice detects that it is a known hazard, updates its own information about this hazard and forwards the information to the TMS.

3.3.1.2 Hazard Notification Message Flow

Each vehicle has a ROI. Information about hazards within their ROI is obtained by vehicles using a subscribe-notify communication pattern, as shown in Figure 3-19 and Figure 3-20, respectively.

The Geoservice provides information about detected hazards within the ROI of the vehicle, using DENM format, with one DENM per hazard.

The vehicle provides its ROIs upon subscription either periodically or when its location has significantly changed according to criteria which are proprietary to the ADAS system. In Figure 3-20, the vehicle is responsible of defining its ROI and its attributes. For RSA flavour, the ROI shape is a geometric form while for the PSA flavour it is a set of unique keys representing to tiles which are translated to MQTT subscription topics.
Notifications are sent when a new hazard occurs in the ROI, a previously existing one is updated\(^8\) or is not present anymore, or when there are hazards in the ROI after a new subscription is sent to the Geoservice. Typically, information about only one hazard is included in the `ListOfProcessedHazardInformation`. But cases might exist where information about more than one hazard is provided. Typically, when updating the ROI, the ROI may include several hazards, and the vehicle did not yet get notified about them since they were not present within its previous ROI. It then gets notified about all hazards, either in a single notification or in a series of notifications.

\(^8\) E.g. its location information or detection confidence is updated

![Diagram](image-url)

**Figure 3-19: Obtaining Hazard Information using Subscribe-Notify Design Pattern with Vehicles Regularly Providing their ROIs**
Application Message Formats
The PSA and RSA vehicles will support different application message formats for communication. The CTTC vehicle will support all formats since it combines the capabilities of both vehicles.

The RSA vehicle uses DENMs for Hazard Reporting and receives DENMs as Hazard Notification. They are fully compliant with the ETSI specifications, i.e., Basic Transport Protocol (BTP) [23] and GeoNetworking Protocol [24].

The PSA implementation uses JavaScript Object Notation (JSON)-encoded message fields present in DENMs, CAMs and CPMs for Hazard Reporting and Notification. The messages do not include information from BTP or GeoNetworking protocols. Authentication and privacy are provided by Transport Layer Security (TLS) as further described in Section 3.3.4.

Role of MEC and Related Challenges
It is assumed that every geographical location has an optimally serving Geoservice deployed on the MEC host close to the Packet Data Network (PDN) Gateway (P-GW) of the cell serving. There is no unique mapping of serving cells to areas since handovers are performed at slightly different locations.

---

Figure 3-20: Obtaining Hazard Information using Subscribe-Notify Communication Pattern with Vehicles Providing Regular Location Updates Enabling the Geoservice to Determine the ROI
area. This is shown in Figure 3-21 where the example area is divided among five MEC hosts close to five different P-GWs. Furthermore, it is not necessarily the case that radio handovers automatically trigger change of serving MEC host and/or P-GW.

In this example, the vehicle is located in an area optimally served by MEC3 according to its P-GW. The ROI is controlled by the vehicle and fully decoupled of the areas which are served by the MEC hosts. It basically depends on its trajectory and is adapted by the ADAS system. At a given time, the vehicle is interested only in hazards within its ROI. In this example, the ROI ranges through the area of MEC1, MEC4, MEC5, and MEC3. The hazard in front of it, e.g. a traffic jam, ranges across the area covered by MEC1 and MEC4 and was accordingly reported to Geoservices deployed on respective MEC hosts. Several challenges are illustrated by this example:

- In the uplink, Geoservices in MEC1 and MEC4 must obtain a consistent view of the hazard
- In the downlink, the information must be available in the Geoservice hosted on MEC3 and later by MEC5 when the vehicle moves

![Figure 3-21: Relation Between Vehicle Location, MEC Hosts and Areas they Serve, Hazard Areas, and Regions of Interests (ROIs)](image)

As shown in Figure 3-18, every hazard is also reported to the TMS, assuring it has a global view. For the MEC-hosted Geoservices, several challenges exist to obtain a consistent picture of existing hazards and assure all vehicles that have the hazard within their ROI get notified.
The requirements for Hazard Reporting are:

- The same hazard could be reported to different Geoservices because it spans across the area of multiple ones and/or because vehicles detecting it are served by different Geoservices.
- All involved Geoservices and the TMS must consistently update their information about the same hazard, including the case of determining when a hazard is not present anymore.
- The hazard area can evolve slowly, e.g., for a traffic jam, and therefore new Geoservices can get involved long after the first report of this hazard.

The requirements for Hazard Notification are:

- Vehicles might have a hazard within their ROI while being served by a Geoservice that never received a report about this particular hazard.
- When Vehicles update their ROI, some hazards which occurred earlier might be present in this new ROI. Therefore, it is needed to receive notifications about hazards that occurred earlier in the new ROI.

### 3.3.1.5 ROI Criticality and Mapping to Network QoS

Vehicles can have a “critical” and an “informational” ROI. The notification of hazards in a “critical” ROI must have higher QoS than the notification of hazard in an “informational” ROI. Hazard Reporting in the uplink is always considered critical since it must be assumed that the hazard may occur in an area considered critical by some vehicles. It means that notification of hazard in a critical ROI and Hazard Reporting shall be prioritized over other data in the network. Prioritization in the network is achieved through dedicated bearers. Informational Hazard Notifications are transmitted on the default bearer while critical ones are transmitted over a dedicated one.

### 3.3.2 Communication between Vehicle and Backend

#### 3.3.2.1 Function, Interaction and Composition of Geoservices

Section 3.3.1 provided high-level information on the role of the Geoservice. This section provides a refined description. Figure 3-22 presents the sub-components of the Geoservices and their interaction. It is a logical subdivision and an implementation subdivision. On the vehicle side, the RSA and PSA vehicles implement different solutions. The Geoservice is composed of an RSA-flavour Geoservice instance, a PSA-flavour Geoservice instance, common services and applications. The RSA-flavour and PSA-flavour Geoservice same MEC host communicate together so that an event which is reported by a PSA car can be known by an RSA car and vice-versa. In cases where no MEC deployed Geoservice can be reached, e.g., due to a lack of MEC hosting capabilities of the serving network, Geoservices deployed on the public Internet can be used instead.
Figure 3.22: Sub-Components of the Geoservice in the Vehicle, MEC and Public Internet

Geoservices on different MEC hosts can be able to communicate with each other through the Inter-MEC interface, but this is not always the case. It depends on the MEC deployment of the MNO.

Geoservices can always reach the TMS on the public Internet. There can be more than one TMS instance, e.g., different ones for different countries. They can always exchange information over the public Internet.

In the following, the sub-components are described. Components with the same name within the Geoservice can have different implementations for PSA- and RSA-flavours. The CTTC vehicle
supports both flavours but, in the following, for simplicity, only “RSA” and “PSA” will be used as label where the two flavours diverge.

The order of the next sections reflects the message and packet flow starting from Hazard Reporting from the vehicle (see Section 3.3.1.1) and then Hazard Notification from the Geoservice (see Section 3.3.1.2).

### 3.3.2.2 Hazard Reporting

#### 3.3.2.2.1 Hazard Report Sender

The Hazard Report Sender creates a Hazard Report when triggered by the Detection and Trigger component in the vehicle. Message formats are described in Section 3.3.1.3. RSA-flavoured vehicle clients pass the message to the V2X Message Manager while for the PSA-flavour it is passed to the MQTT Client.

#### 3.3.2.2.2 V2X Message Manager in Vehicle (RSA-flavour only)

The V2X Message Manager attaches a BTP and GeoNetworking header to the message received from the Hazard Report Sender. Target Geoservice address and port number are determined from the Edge Server Selection sub-component. If no entry for the location exists, a default Geoservice hosted on the public Internet is used instead.

The V2X Message Manager sends Keep Alive messages in UDP packets to the Geoservice in parallel, preventing the radio network to switch the modem in the vehicle to a non-active state. The Keep Alive messages in UDP packets to the Geoservice also establish a downlink connection from the Geoservice to the vehicle and assure that Network Address Translation (NAT) gateways along the data path keep a mapping table entry allowing Hazard Notifications to reach the vehicle.

There are two ports for sending the Keep Alive. When the ROI is critical, Keep Alive messages are sent on port 1; when the ROI is informational, Keep Alive messages are sent on port 2. The source port numbers correspond to the ones where critical and informational notifications should be received.

The message from the RSA vehicle with BTP and GeoNetworking header is transmitted over UDP. There are multiple reasons for preferring UDP to TCP: the same V2X stack is used for short-range communications and long-range communications. When sending a Hazard Report of stationary vehicle in short-range, it is transmitted 15 times along the validity duration of the event. These message repetitions improve the reliability of the transmissions to the Geoservice, at least for the Hazard Report and reduce the value of TCP.

#### 3.3.2.2.3 MQTT Client (PSA-flavour only)

The MQTT Client is only present in the PSA solution flavour. The message from the Hazard Report Sender is sent to the currently used Geoservice, according to the current subscription (see Section 3.3.2.3.3). The MQTT topic to be used, corresponding to the position of the hazard, which is contained in the message, maps to the logical area or areas determined through the Georeferencing Service described in section 3.3.2.3.4.
3.3.2.2.4 Operating System (OS) Network
The operating system enables the service to send and receive data over IP networks using TCP or UDP as transport layer protocol. Furthermore, it provides Domain Name Service (DNS) name resolution to translate Fully Qualified Domain Names (FQDNs) to IP addresses.

3.3.2.2.5 V2X Message Manager in Edge Geoservice (RSA-flavour only)
Hazard Reports from RSA vehicles are received by the V2X Message Manager in the Geoservice. The GeoNetworking and BTP header is removed and the message is passed to the Hazard Report Receiver.

Furthermore, the V2X Message Manager receives Keep Alive messages from the vehicle. They are only intended to allow Hazard Notifications to be correctly routed through NAT gateways and to map the Hazard Notifications on the port corresponding to the criticality of the ROI.

3.3.2.2.6 MQTT Broker (PSA-flavour only)
In the PSA solution flavour implementation, the Hazard Report Receiver has subscribed to all currently relevant\textsuperscript{10} topics of the MQTT Broker. It receives the message together with meta information from the removed MQTT header, e.g., the topic under which the message was published by the MQTT Client.

3.3.2.2.7 Hazard Report Receiver
The Hazard Report Receiver uses the Hazard Database to determine if the report is a duplicate report of a known hazard (then it is discarded), if the report applies to a new or updated report (then the Hazard Database is updated) or if the report applies to a known hazard issued from another vehicle. In the latter case, the Data Fusion sub-component is involved to update the information of the existing hazard in the Hazard Database. Updating includes the special case when the received Hazard Report indicates a hazard does not exist anymore.

The received Hazard Report usually triggers a Hazard Notification, as described in Section 3.3.2.4. Furthermore, the Inter-MEC sub-component shown by the black arrow, is used to disseminate information to other Geoservices, if needed. The Geoservice to TMS MQTT Client disseminates the information to the TMS. This is further described in Section 3.3.2.6.

3.3.2.3 Hazard Notification Subscription Management and Authentication
Many subscription and authentication related tasks are handled by the MQTT Client and Broker for the PSA solution flavour, while the RSA one has own methods and responsible sub-components to handle it.

3.3.2.3.1 Authentication
RSA vehicles authenticate towards the Authentication sub-component in the Geoservice.

\textsuperscript{10} It is for further study how to determine those. As a starting point, all topics can be monitored using wildcards
Login name, password, Media Access Control (MAC) address, brand name, and device name are submitted for authentication using HTTPS POST with HTTP authorization header. Login name and password are unique per vehicle. Upon successful authentication, the vehicle receives a HTTP 200 OK status with a JSON Web Token (JWT) over the “x-access-token” header of HTTP with a current lifetime of 7 days. If authentication is not successful, HTTPS 403 status is returned to the vehicle.

PSA vehicles authenticate through the means provided by the MQTT Client and Broker using TLS certificates. Login name and password are provided upon first connection to a Geoservice using an MQTT CONNECT-message. After evaluation of the credentials, the MQTT broker returns the CONNACK-message containing return code 0 for successful connection, 4 for bad login name or password, or 5 if the user is not authorized to use the service. The procedure must be repeated when the serving Geoservice changes, but the same credentials can be used on all Geoservices.

The Authentication procedure is done on the default bearer.

3.3.2.3.2 Edge Server Selection
It can be assumed that for every given location and serving mobile network, there is an optimal MEC host. The Geoservice hosted on it should be used for communication. As trial solution, a lookup table is used. When receiving a position change from the GNSS sensor, the lookup table is queried. The vehicle then decides if the current MEC host can still be used or if a better one should be reselected. In case the MEC-hosted Geoservice must be changed, or a Geoservice hosted on the public Internet must be used, unsubscription and subscription is triggered through the Subscription sub-component.

3.3.2.3.3 Subscription
The RSA flavour uses a dedicated Subscription sub-component while the PSA one relies on means of the MQTT Client.

The RSA flavour uses a HTTPS POST message with parameters in JSON format and also including the JWT obtained during authentication, see Section 3.3.2.3.1. Following parameters are included:

- ROI: Geometric shape parameters representing the ROI,
- Event type: Corresponds\(^\text{11}\) to that of DENM messages,
- Time to Live (TTL): Time in minutes for which the subscription request is valid,
- MAC address of the vehicle used during authentication. If the MAC address changes for any reason, the vehicle must re-authenticate itself.

\(^{11}\) Setting event type to 0 means all even types are considered, but initially only DENM is considered
There is only one valid subscription per vehicle: any new subscription overwrites the previous one. When changing Geoservice, as described in Section 3.3.2.3.2, an explicit unsubscription should be sent. If not done, the subscription will eventually time out according to its TTL.

The PSA vehicle uses the MQTT subscription mechanism to subscribe to the topics corresponding to its ROI. When changing to another Geoservice, the vehicle must unsubscribe from all topics of the previous one.

The subscription procedure is done on the default bearer.

3.3.2.3.4 Georeferencing
The RSA solution flavour allows any geometric shape to be used to describe ROIs to the Geoservice. For every detected hazard, the Geoservice will check which vehicles have this hazard within their ROIs.

The PSA solution flavour requires that Geoservices in the backend and in the vehicles must use a common mapping of positions to logical areas\(^{12}\). The reason is that each logical area translates to a related MQTT topic.

3.3.2.3.5 ROI Database
The ROIs are submitted by vehicles upon subscription. It is used by the Hazard Notification Sender to determine the recipients of Hazard Notifications.

3.3.2.3.6 ROI Management
Either triggered by passed time or by position updates or a mix of both, the ROI Management sub-component invokes the Subscription sub-component to update the ROI.

For the RSA flavour, the ROI will be periodically updated. The PSA vehicle will do its updates according to the used quadtree georeferencing (see Appendix A). When the ROI is covering a new logical area or not covering a previous one as the vehicle moves, the MQTT Client is invoked to perform the respective subscriptions and unsubscriptions from the topics mapping to those logical areas.

3.3.2.4 Hazard Notification

3.3.2.4.1 Hazard Notification Sender
Hazard Notifications are triggered when the Geoservice receives new or updated Hazard Reports either directly within the same Geoservice or through the Inter-MEC components. It also sends a Hazard Notification when an Application component of the serving Geoservice detects a new or updated hazard based on a newly received CAM or CPM. Furthermore, a new or updated ROI subscription by a vehicle can trigger the transmission of Hazard Notifications for this ROI towards the vehicle. In the latter case, the Hazard Database is queried for hazards within the respective

\(^{12}\) The word “tile” is often used as synonym, but it is commonly associated with adjacent rectangles forming the logical areas. We therefore use the more generic term “logical area”. 
ROI of the new or updated subscription. If there are hazards which are in the new ROI and which were not present in the previous ROI, a notification with these hazards should be sent.

For the other cases, e.g., when the Geoservice receives a new or updated Hazard Report, when its Data Fusion component updates the hazard information or when an Application detects a new event, it updates the Hazard Database. Then the ROI Database is queried for the RSA solution flavour to determine all vehicles that have the hazard within their ROI. A Hazard Notification message is constructed based on the hazard information in the Hazard Database, including the most recent update from the just received Hazard Report information, and passed to the V2X Message Manager together with a list of vehicles that should receive it.

For the PSA solution flavour, the Quadtree Georeferencing sub-component is used to determine the smallest logical area covering the hazard position and from there the respective MQTT topic for publication. As for the RSA solution flavour, a message is constructed. The message is published to the MQTT Broker with the devised topic.

The Hazard Database is periodically cleaned-up so that events which are not valid anymore are deleted.

3.3.2.4.2 V2X Message Manager in Edge Geoservice (RSA-flavour only)
The V2X Message Manager in the Geoservice adds a BTP and GeoNetworking header to the message. It retrieves the connection endpoint of each vehicle and passes the packet for each destination vehicle. Since the connection endpoint of each vehicle is derived from the Keep Alive message, the Hazard Notification is sent on a port which depends on the ROI criticality.

3.3.2.4.3 MQTT Broker (PSA-flavour only)
The messages are published to the MQTT topics representing the logical areas covering the hazard location. The MQTT Brokers pass the messages to the different vehicles which have subscribed to the topics overlapping the hazard location.

3.3.2.4.4 OS Network Service
The OS Network Service transmits the packets using UDP (RSA solution flavour) or TCP (PSA solution flavour).

3.3.2.4.5 V2X Message Manager in Vehicle (RSA-flavour only)
The message is passed to the Hazard Notification Receiver after the BTP and GeoNetworking headers were stripped from it.

3.3.2.4.6 MQTT Client (PSA-flavour only)
The message is passed to the Hazard Notification Receiver.

3.3.2.4.7 Hazard Notification Receiver
The information from received Hazard Notifications is passed to the ADAS Decision component for PSA solution flavour vehicles and logged or displayed on an HMI. For PSA solution flavour
vehicles, the ADAS supervisor can display the message on an HMI (if the received message is only informative).

### 3.3.2.5 Hazard Detection Application

Applications are also developed to detect hazards in real time. The processes are based on already existing Hazard Notifications, CAMs and CPMs. When such a new hazard is detected, it is put in the Hazard Database.

The applications mainly use the position (latitude, longitude) and the mobility information (speed, acceleration and heading) available into CAM and CPM message, to detect stopped vehicles and traffic jams. If such an event is detected, they create an event in the Hazard Database and inform both flavours that a new Hazard is present. Information of these events is updated in real time when a new message relative to these events is received.

#### 3.3.2.5.1 Stationary Vehicle Detector (SVD)

When the application receives multiple consecutive CAMs of a vehicle with a speed close to null or multiple CPMs reporting a vehicle with a speed close to null at a given position, the application checks if this hazard for the position exists on the Hazard Database:

- If it does not and no other stopped vehicle in the neighborhood is known, a new hazard “Stationary Vehicle” is inserted in the Hazard Database.
- If it is already present in the Hazard, the application uses the new messages to improve the level of confidence and the position accuracy of the hazard.

When the application receives a CAM or a CPM with a speed which is not null, for a vehicle which is tagged as stationary, it deletes this hazard from the Hazard Database. The vehicles whose ROI covers this hazard are notified that the hazard has disappeared.

#### 3.3.2.5.2 Traffic Jam Detector (TJD)

When the application receives a CAM or a CPM with a speed close to zero, it checks if another slow or stopped vehicle is detected in the neighbourhood. If there is, it inserts the hazard “traffic condition” in the Hazard Database.

When the application receives a CAM or a CPM for a vehicle in the traffic jam with a speed above a given threshold, it deletes this hazard from the Hazard Database and the vehicles whose ROI covers this hazard are notified that the hazard has disappeared.

### 3.3.2.6 Interaction among Geoservices and with TMS

#### 3.3.2.6.1 Inter-MEC Communications

In Section 3.3.1.4, the challenges for the need of information exchange of MEC-deployed Geoservices among each other and with the TMS are described. Briefly summarized, the challenges arise from the fact, that the Geoservices receiving and processing a Hazard Report are not necessarily the same ones serving the vehicles that need to receive notifications about the hazards. Furthermore, it cannot be assumed that all MEC-hosted Geoservices can interact directly, e.g. in the case when deployed on MEC hosts operated by different MNOs. The proposed
architecture assures all vehicles to be notified of a new or updated hazard even it does not occur under the coverage of their serving Geoservice. For the sake of simplicity, it is assumed for the trials that all Geoservices are associated to the same TMS.

3.3.2.6.1.1 Purely Rely on TMS
According to Figure 3-18, any new or updated hazard information received at a Geoservice is forwarded to the TMS. Similarly, if a Geoservice detects a hazard by processing CAM or CPM messages, it also forwards this hazard to the TMS.

In parallel, every Geoservice subscribes to geographical areas which are outside of its coverage area but close to it; the close area is to be defined and is indeed a topic of the real-world implementation. By this means, it is notified about a new hazard information which occurs under the coverage of another neighbour Geoservice, through the TMS, as explained by Figure 3-23. And thus, a vehicle whose ROI exceeds the coverage area of the serving Edge Geoservice is notified about hazards which occur in its ROI, but which are out of the coverage area of the serving Geoservice. The Edge Geoservice subscription is static and is fully decoupled of the vehicle subscription and ROI.

Both Edge Geoservices and Central Geoservice discard any notification which is already in their Hazard Database.

3.3.2.6.1.2 Edge-to-Edge Geoservice Direct Communication
We can assume that at least Geoservices deployed on MEC hosts of the same MNO can directly interact with each other. The serving Geoservice subscribes to the other Geoservices to be informed about any new or updated information of an hazard which occurs in a geographical area out of its coverage area but under the coverage area of one of these reachable Geoservices (Figure 3-24).

Since not all Geoservices might be reachable, this solution can be combined with previously described ones involving the TMS.
Previously described solutions assume that Hazard Reports are received by the Geoservice serving the reporting vehicles. However, some hazard may not be reported by the vehicles, e.g., if the hazard type is not supported by the vehicle. But the road operators can inform the TMS about such hazards. It is up to the TMS to dispatch this information to the interested Geoservices.

### 3.3.2.6.2 Intra-MEC Communications

The RSA and PSA vehicles implement different solution flavours. The Interexchange component allows both solution flavours inside the same MEC host to communicate, so that a Hazard Reported by a PSA car is known by the RSA car and vice-versa.

When a new or updated Hazard report is received by one flavour, the hazard information is forwarded to the other flavour thanks to the adaptation/translation into a JSON file sent through a software broker that allows the exchange of information among the two protocols. When CAM or CPM messages are received by any flavour, the information is forwarded to the Hazard Detection application.

### 3.3.3 Onboard Software Architecture

#### 3.3.3.1 CTTC Vehicle

The vehicles provided by the CTTC are non-automated vehicles equipped with a Car PC. These vehicles are not equipped with smart sensors, e.g., camera, radar, LiDAR, etc., hence they are not capable of detecting hazards in their proximity. The idea is to emulate the detection of hazards from the Car PC software in order to trigger the transmission of Hazard Reports in the form of DENM on the uplink to the Geoservice, allowing to collect many measurements in the trials. Similarly, Hazard Notifications in the form of DENM can be received on the downlink and subsequently such notifications are displayed on the Car PC’s HMI.
The external communications of the Car PC with the Geoservice are established over LTE cellular connectivity. Fundamentally, two types of ETSI ITS-G5 standard-compliant messages are used for information exchange with the Geoservice over the UDP protocol (in RSA flavour) or MQTT protocol (in PSA flavour): periodic Cooperative Awareness Messages (CAM) and event-triggered Decentralized Environmental Notification Messages (DENM). The exchanged information between the Car PC and the Geoservice is displayed on the HMI of the Car PC, providing an OpenStreetMap with the current position of the vehicle and Hazard Notifications received in the form of DENMs from the Geoservice, i.e., detected events within vehicle’s ROI.

Figure 3-25 shows the basic hardware components of the Car PC. The Car PC consists of a Linux Laptop computer and a software application developed on top of an ETSI C-ITS protocol stack. The Car PC integrates a positioning subsystem (e.g., (GPS)/GNNS receiver), an LTE cellular modem and an interface to the Controller Area Network (CAN) bus of the vehicle by means of an On-Board Diagnostic (OBD) interface adapter or a Peak Controller Area Network (PCAN)-USB adapter.
3.3.3.2 RSA vehicle
RSA ACCA architecture relies on the following modules shown in Figure 3-26:

- CCU: It sends information about ego-events in DENM messages. It also receives information about events of interest from the Geoservice in DENM messages.
- CAN bus: it provides vehicle status which is read by the Detection & Trigger component.
- ADAS: The V2X Perception Module puts the received V2X events on the local map and provides events of interest to the Decision-Making Module which is responsible of the driving policy.
- HMI: It does a map-matching of the received events and displays them on the vehicle HMI.

![Figure 3-26: ACCA Architecture on RSA vehicle](image)

From a functional perspective, the embedded software plays two roles:

- Transmission: The CAN bus provides the vehicle status. Under some triggering conditions, this status is used to build a Hazard Report and to be transmitted to the Geoservice.
• Reception: It receives Hazard Notifications in DENM to enhance its in-advance perception thanks to the V2X Rx Data Processing module. The V2X Connection Manager controls the Geoservice and the Hazard Notifications.

The V2X connection manager includes the following sub-components described in 3.3.2.3: Edge Server Selection, Authentication, ROI Management and Subscription. This component is responsible of updating the ROI according to specific criteria. When it detects a loss of coverage or a change of MNO (at cross-border for example), it searches for the best Edge server and reconnects to the Geoservice hosted on the reselected server. It also configures the Keep Alive port according to the ROI criticality so that the Hazard Notifications are sent to the bearer of the corresponding QoS.

In addition, the received V2X Objects (or Hazard Notifications) are pre-processed: they are merged with the V2X objects which have been received on the short-range technology and they filtered out if they are not relevant or critical for the user. The position of the objects is also checked with respect to the locally available map. When the data is considered as relevant for the vehicle, it is sent to the vehicle HMI via the automotive someIP protocol and shown on the vehicle HMI. In parallel, the data is sent to a decision-making policy which decides the driving strategy or manoeuvre to take.

3.3.3.3 PSA Vehicle

Each vehicle provided by PSA group for the 5GCroCo project is equipped with a CCU that makes the communication interface between onboard functions (like ADAS functions) and communicating nodes outside of the vehicle (e.g., the network backend). Figure 3-27 shows the embedded architecture of PSA cars for the ACCA Use Case.
The main modules of PSA vehicles used in the project are described below and shown in Figure 3-27:

- **CCU**: It is capable to send information acquired from onboard sensors, as well as receive and combine information received from the backend. The CCU communicates with the backend through the Uu interface of the 3GPP network.
- **Sensors**: All vehicles are equipped with onboard sensors capable to sense the environment in which the vehicle is operating. PSA vehicles are equipped with cameras and radars.
- **ADAS Supervisor**: It receives data from sensors, interprets such data as hazard events or road perception before transmitting them through the CCU. The ADAS supervisor uses standardized DENM-like message or CPM messages (both JSON-encoded), in order to transmit hazard events and road perception, respectively. CAM messages are also supposed to be sent to the Geoservice in the backend. Moreover, the ADAS supervisor module is responsible to receive data from the network infrastructure (by CCU), verify its authenticity and send it to the vehicle core system.
- **HMI**: It is integrated in the vehicle dashboard and enables the vehicle core system to display messages from local sensors, as well as the ADAS supervisor to display maps and Hazard Notifications received from the Geoservice.
- **Actuators**: When receiving a Hazard Notification from the Geoservice, the ADAS supervisor determines if the reported event leads to a risk of collision and potentially adapts the longitudinal behaviour of the vehicle accordingly.

**Figure 3-27: Embedded System of PSA Vehicles for the ACCA Use Case**

The main modules of PSA vehicles used in the project are described below and shown in Figure 3-27:
• Vehicle Core System: It is responsible to combine data from onboard sensors and from network infrastructure, in order to make the decision to trigger the actuators.

3.3.4 Security
Below, security measures applied between different components of the ACCA Use Case are described. It is not intended to enable them all during trials as this complicates result analysis. Instead, it will be inspected which security measures can have an impact on measured KPIs, especially the Application Level Latency and tests will be conducted to quantify this impact. But even those results will have to be taken with care as the prototypes used might not benefit from hardware acceleration typically applied in this field.

For the RSA-flavour, an ETSI-ITS compliant protocol stack is used for the Hazard Report and Notifications, where classically on Geo Networking layer message signing is performed based on certificated issued from a PKI. This enables receivers to validate the authenticity of the message content. It means that the Geoservice will accept reports from trustable vehicles. Certificates issued by the PKI can be revoked: if a vehicle is identified as being malfunctioning or sending malicious data, its PKI can be revoked by a certified authority. By this means, the Hazard Reports of such a vehicle are discarded by the Geoservice. Technically, message signing could be done at other layers than the Geo Networking layer. The ETSI-ITS specification allows to optionally enable this mechanism per service. It has been implemented in the project but is usually not enabled to relax the dependency on a PKI. In parallel, subscription procedure is protected with an authentication mechanism.

The PSA-flavour and communication across backend components uses MQTT transported over TCP. This allows to use security measures that are part of the MQTT specifications and ones that can be applied with TCP, namely TLS. A comprehensive description of these measures is provided in the security section of Deliverable D3.2 [3]. In the project MQTT authentication with username and password is used to connect to an MQTT broker. Furthermore, TLS is available between vehicles using the PSA-flavour and corresponding backends. TLS enables authentication of servers (MQTT brokers) towards clients and encryption.

On top of these measures, the Geoservice can implement some plausibility checks: it can check the consistency of the different fields of a received Hazard Report. It can also check the plausibility of a received Hazard Report with the reports of other vehicles in the same location.

3.3.5 Real-world Architecture
Previous sections provide an architecture consisting of functions and components jointly enabling them. This allows to demonstrate the ACCA Use Case and evaluate KPI performance in trials. Besides this, also a KPI measurement system is in place that might not be present in a real-world deployment of the use case. Further things can be different in a real-world deployment than what

---

13 All components in the backend are considered servers but when communication among each other they take the roles of clients or servers and it is then always the backend component that has the server role (MQTT broker) that authenticates towards the backend component in client role (MQTT client).
is described in the trial-architecture above. This section explains these differences. Its main focus is on the fact that in reality the ACCA Use Case is embedded in a setting with many more than the two OEMs involved in 5GCroCo trials of this use case and also more Road Traffic Authorities (RTAs)\(^{14}\) and similar entities with their individual Intelligent Traffic Systems (ITS) and Information and Communication Technology (ICT) systems have to be considered.

Figure 3-28 shows the envisioned real-world architecture in which ACCA, together with other Cooperative, Connected, and Automated Mobility (CCAM) services of this kind, is embedded. Terms used for stakeholders follow the application reference architecture from 5GAA [25], as further described in Deliverable D3.2 [3]. Each component can have different suppliers and subcontractors involved so the focus when mapping stakeholders to components is on who provides the service in a sense of being responsible\(^{15}\) for its operation and content. While the terms “MNO” and automotive “OEM” should be clear, “RTA” and “Service Provider (SP)” require further clarification: RTAs are entities that are either public or private bodies. In the latter case the respective private company received a mandate from a responsible public authority, e.g., in case of privately-operated motorways. SPs are all other private entities with no such mandate, e.g., Google Maps. When explicitly mentioned the term “MNO” refers to the entity providing connectivity and MEC-hosting services. This should not preclude that MNOs can offer further services, but they are then in the role of an SP.

---

\(^{14}\) We use RTA as generic term covering all stakeholder, public or private, involved with the road transport system and its environment. Considered private entities usually got a mandate from a public authority.

\(^{15}\) This does not imply any liability. Content is assumed to be provided with no guarantees if not stated differently. Such “guarantees” are not subject of this architecture discussion.
The Geoservice is composed of many sub-components, as described in Figure 3-22 in Section 3.1.3. The actual communication to the vehicle can be within the responsibility of the corresponding vehicle OEM. Besides this, all stakeholders and the OEM can deploy further sub-components to enable new features for ACCA or other use cases. For the ACCA Use Case to work communication with the TMS is essential to provide inter-MEC communication where this is not directly possible and to provide the Geoservice in cases where a MEC-hosted one is not available. TMSs can also serve as gateway to other backend components provided by other stakeholders, but it is also possible they directly communicate with the MEC-hosted Geoservice.

There is also OEM-SP and RTA-SP interaction shown but direct communication between OEM and RTA is considered not common as there are thousands of different RTAs which would require an immense integration effort by the OEM. But it shall not be precluded that such integration is done, e.g., for RTA systems covering large areas like one or more countries and/or for OEMs with tight bounds to the geographical region covered by the RTA system.

### 3.3.5.1 Cross-OEM and OEM-RTA Interoperability

This section first presents the current situation regarding cross-OEM and OEM-RTA interoperability and challenges emerging from it. It then points at potential solutions.
Figure 3-29: Short-range Interoperability

Figure 3-29 shows the short-range communication technology system. It includes ITS-G5, also known as IEEE 802.11p, and C-V2X PC5, also known as sidelink, as radio access technologies. A vehicle that wants to communicate with a Roadside Unit (RSU) of one of these radio access technologies, shown in purple and orange, needs a corresponding On-Board Unit (OBU). The protocol layers above the radio access technology have been clearly specified in ETSI Technical Committee Intelligent Traffic Systems (ETSI-ITS) for Europe and very similar specifications exist in other parts of the world, e.g., IEEE 1609 (also known as Wireless Access in Vehicular Networks). Any set of protocols / message formats can be used.

\[16\] It is not shown but shall not be precluded that the same RSU supports both radio access technologies.
Environments (WAVE)) is the United States (US) and China GB/T 31024). The specifications allow several degrees of freedom, so profiles were created to assure interoperability throughout the protocol stack and the information conveyed in the message. In Europe this was done in the C-Roads Platform for V2N and the Car2Car Communication Consortium for V2V the so-called C-ITS Day 1 use cases. Both cooperated were necessary. This assures that RSUs and OBUs, of the same radio access technology, regardless of the supplier, can exchange messages, as represented by the yellow shapes. For many V2N services the RSU needs to communicate with one or more RTAs responsible for the geographical area where it is placed. This is done over wide area networks (WANs) including the public Internet and/or ITS networks related to an RTA using the Internet Protocol (IP) and typically UDP or TCP as transport layer protocol. Such IP-networks are depicted in red.

Commonalities exist between the different RTA backend systems, like e.g., the use of Data Exchange (DATEX) II, but they are usually not as precisely profiled as for the short-range protocols, as described above. Integration between the RSUs and RTA backends is therefore required. The RSU vendor would usually receive the necessary information to integrate towards one or more RTA backends relevant in the area where the RSU gets deployed or the RSU has to be customizable so the RTA or a third party on its behalf (integrator) can do this. Changes to the interfaces at an RTA can require corresponding adjustments to the RSUs using these interfaces. Different RTAs and interfaces used are depicted with dark red geometrical shapes. Dark red lines depict different protocols above the transport layer (typically UDP or TCP) and message formats that just have in common that they can be transmitted over IP-networks.

---

17 There are many technical means to virtualize and/or secure a physical communication links and many operational and ownership models possible. None of them shall be precluded but it is not relevant for these architecture topics which ones are used.
Figure 3-30: Vehicle-related Internet of Things (IoT) Ecosystem

Figure 3-30 depicts what is referred to as “IoT ecosystem” in this document. For each OEM a corresponding backend is shown. Each vehicle uses OEM-specific protocols and message formats. It is likely that they follow some common IoT practice, but they should not be considered interoperable across different OEMs. Beside OEM, also SP backends are shown. The upper one represents any kind of backend which can provide any vehicle-related service. It can use any protocol and message format that can be transmitted over an IP-network, as indicated by the dark red line. The lower backend represents a special kind of services called Data Marketplaces. They usually collect and sell information but providing it free of charge shall not be precluded here. The icons inside indicate that it can provide information from four different RTA backends as well as from OEM2 (green) or OEM4 (brown). The OEM-specific data would typically be obtained through the respective OEM backend, but there can be cases where it comes directly from the vehicle, e.g., if there is a corresponding cooperation between the OEM and the SP. The Data Marketplace exposes its data through some protocol and message format that can be transmitted.

---

18 This shall not preclude that all OEMs need to have their individual backends.

19 E.g., the SP also supplies corresponding hard- and/or software in the vehicle
over IP-networks and would usually provide according documentation or even a machine-readable interface description. Consumers of the data can access it by client software supporting the interface. Changes to this interface could require respective changes on the client side. In order to obtain the information from different sources, two ways exist. Either the data providers, e.g., OEMs or RTAs, provide it through the interfaces exposed by the Data Marketplace or the SP of the Data Marketplace needs to integrate towards the interfaces exposed by the OEMs or RTAs. Also, here a change of interfaces can require adjustments by others. The TMS can be seen as example for a Data Marketplace.

Also, on the RTA backend side a special kind of service exist. It is called National Access Point (NAP) and mandated by EU legislation for all EU Member States [26]. They assure “data is accessible on a non-discriminatory basis, in accordance with the necessary standards for exchange and reuse.” [26]. Similar like for the other RTA backends, commonalities like use of DATEX-II exist, but it is not completely aligned and profiled. The corresponding information flow is therefore also considered to be of any kind that can be transmitted over an IP-network (dark red line). NAPs and Data Marketplaces both serve the purpose of collecting data from different sources, potentially with different interfaces, and exposing it through one or few well-documented interfaces, potentially also defined in machine-readable format. NAPs use the same means as SPs running Data Marketplaces to integrate towards other RTAs. It is not precluded that NAPs also obtain data from Data Marketplaces. The main difference between NAPs and Data Marketplaces is that NAPs fulfil a legal obligation in EU Member States.

An overview of solutions for V2X and V2N services [27] over mobile radio networks was collected mid of 2019 by the EU CONCORDA20 project.

20 https://ertico.com/concorda/
Figure 3-31: C-Roads IP-Based Interface Introduced to Vehicle-related IoT Ecosystem

Figure 3-31 shows a new component within the RTA backend domain implementing the C-Roads IP-Based Interface [28] which is technically an Advanced Message Queuing Protocol (AMQP) message broker. It enables exchange of information across different backend components through a publish-subscribe communication pattern. AMQP is very similar to MQTT and plugins enabling MQTT for AMQP brokers and translating between the two solutions exist. MQTT encodes context information in the “topic” which is part of the Unified Resource Locator (URL) separated by slashes. This URL is used to subscribe to the corresponding topic and in case of the PSA-flavour of ACCA the topic mostly reflects a geographical ROI encoded as Quadtree. AMQP allows to use “attributes”, being key-value pairs to attach context information to a message. This information can be used internally to filter messages to certain queues. The names of these

21 https://www.rabbitmq.com/mqtt.html
queues do not necessarily have to reflect the attributes. A message client requires additional information which queue contains the information of interest, e.g., according to certain values of the attribute representing the geographical location. Currently, the IP-Based Interface specification only defines how certain fields from selected ETSI-ITS messages should be used to populate corresponding attribute key-value pairs. This could in the future be extended to other message formats than ETSI-ITS e.g., by defining equivalent fields, as done between ASN.1 encoded RSA-flavour following ETSI-ITS DENM specifications for Hazard Reports and Notifications and the PSA-flavour using mostly the same information elements but encoding them with JSON. The C-Roads IP-Based Interface uses Quadtree to reference the area the message content relates to, e.g., where a hazard happened. The IP-Based Interface specifications define a Basic Interface, which is the actual message transmission and population of key-value attributes, as described above and the Improved Interface used to convey control information. It is e.g., used to identify the right queues, as queue names are not directly related to the attributes used to filter and route messages towards them. It is also intended for information sources to detect the right backend instance where they should inject their information.

Compared to previously discussed Data Marketplaces and NAPs. The C-Roads IP-Based Interface offers the advantage of having a well-defined interface specified in within the C-Roads Platform. Software vendors can provide compliant server and client components that stay compatible to each other as the specifications evolve. The communication is therefore depicted with a yellow dashed line in Figure 3-31 rather than a dark red one as used for the NAPs and Data Marketplaces. On the other hand, this imposes the limitation of following the C-Roads processes to upgrade the interface, e.g. to overcome the limitation that only for a subset of ETSI-ITS defined messages the specifications cover how to create attribute key-value pairs from the message content to use them to route the messages to corresponding queues and/or for client-side filtering. Alternatively, implementation specific extensions could be added to e.g., also support JSON-encoded hazard messages very similar to DENMs. This way faster innovation cycles could become possible, but it also bares the risk of diverging solutions as it is the case for NAPs and Data Marketplaces. But as for those, well-documented, or even machine readable, interface specifications allow clients to integrate towards components using extensions beyond what is specified for the C-Roads Basic Interface.

The yellow stars in Figure 3-31 represent components that either receive from or send to an AMQP broker-based C-Roads IP-Based Interface. In this example the NAP and Data Marketplace are both attached, some other RTA backend components and two of the four OEMs.
Figure 3-32: Vehicle-related IoT Ecosystem with MEC-hosted Backend

Figure 3-32 introduces the option of MEC-hosted backends to the vehicle-related IoT ecosystem described before. MEC-hosting should not be just considered as another location to deploy a backend application server to e.g., reduce the Application Level Latency (see Deliverable D2.1 v3 [2]). Similar like short-range RSUs discussed above, a MEC-hosted Geoservice is supposed to serve a certain area, as further described in Section 3.3.2.6. Just as for RSUs, MEC-hosted Geoservices must integrate towards RTAs serving the area. In addition, there is a multitude of protocols and message formats possible between the MEC-hosted Geoservices and vehicles of different OEMs. Even when same protocols and message formats are used, OEMs might insist to remain responsible for their OEM-specific components in the Geoservice, e.g., for security reasons.
Figure 3-33: Baseline Interoperability Challenge within an RSU and Possible Solutions

Figure 3-33 reuses the RSU example to visualize the challenge of integration which will similarly be discussed for MEC-hosted Geoservices, where the challenge is even more complex. In case of RSUs the left side, representing protocols and message formats used when communicating with the vehicle, is always the same. The right side can be different for each RTA. For the sake of simplicity, only one RTA interface is shown. The RTA interface is shown inside the RSU in a sense of “this is how it is exposed towards the RSU”. The right side of the yellow ETSI-ITS box represents the interfaces provided by the RSU vendor. The top left figure presents a baseline situation where the RSU cannot interoperate with the RTA to deliver a V2N service. The top right figure shows the solution where the RSU vendor integrates towards the RTA. The RSU vendor is of full control of the RSU software and therefore able to do that if the RTA provides the required information. The bottom solutions assume that the RSU can be customized, e.g., through plugins. In case of the bottom left solution an adapted is introduced that translates between the RSU and RTA interface. This adapter is a piece of software and can be developed by the RTA or a third party. The bottom right solution, where the RTA adapts to the RSU interface, is different in the sense that the RTA interface is changed. If others, e.g., RSUs from different vendors, already used that interface, interoperability would break when it is changed. It also becomes obvious that any changes in the interface can require to again adapt. This can be prevented if new interfaces, or interface versions, are introduced while keeping old ones operational.

---

22 Internally, the top right solution, where the RSU vendor did the integration, could also use an adapter. This is not the main scope as the main scope is who does the integration.

23 “Developed by” in a wider sense including the option to outsource the task to a subcontractor.
Figure 3-34: Baseline Interoperability Challenge with MEC-hosted Geoservices and How it was Solved in the 5GCroCo ACCA Use Case Trial Implementation

Figure 3-34 shows the interoperability challenge in case of MEC-hosted Geoservices. In this case it cannot be assumed that a common agreement on protocols and message formats exists and even if it does OEMs might want to keep control over the backend part directly communicating with the vehicle. In the RSU case it is very likely that the RSU vendor will do the adaptation towards the RTA as the RTA is often the entity purchasing the RSUs. In case of MEC-hosted Geoservices the OEM-specific parts are ideally reused world-wide, or at least in large geographical regions. Just looking at Europe this would mean adapting to hundreds, even thousands (when including municipalities) of RTA systems. The right side of the figure shows how the challenge was solved in the ACCA Use Case trial implementation, which can also be applied in the real-world. RSA- and PSA-specific backend software was refactored and combined to obtain OEM-specific and common parts. In this tight collaboration it was also possible to slightly adjust existing interfaces, as shown by the change of shape of some interfaces. This way integration towards the TMS\(^{24}\) was achieved and integration among the two OEM-specific solution flavours. The details of the split between common and OEM-specific parts are described in Section 3.3.2.1.

---

\(^{24}\) The figure shows the TMS in dark red as it took the role of an RTA backend. This should not preclude that the TMS could also be operated by an SP making it a Data Marketplace.
Figure 3-35 shows the MEC-hosted Geoservice in context of the vehicle-related IoT ecosystem described earlier. It does not further show how integration towards the OEM-specific parts can be done, but the solution described before as depicted on the right side of Figure 3-35. Figure 3-34 is one option. It can be trivially assumed that the OEM-specific parts of the Geoservice can also interact with their respective OEM backends, as the same entity, in this case the OEM, is responsible for the components. Now only one RTA is shown intentionally assuming that in order to limit integration complexity. The corresponding RTA could be equivalent to the entity operating the NAP for the given country. This would reduce integration complexity but will still require individual adjustment at least per EU Member State. Below RTA1 it is shown how also Data Marketplaces, with their individual interfaces, and the C-Roads IP-Based Interface can be directly used in the MEC-hosted Geoservice. The example in the figure, of which countless variations exist, shows that OEM4 (green) is making use of the Data Marketplace to e.g. obtain RTA information in that Data Marketplace or even provide information to it. OEM1 (black) does not interact with that Data Marketplace but it does make use of the C-Roads IP-Based Interface. At
least for ETSI-ITS message format it can seamlessly send and receive e.g. DENMs which are also used within the RSA-flavour of the ACCA Use Case. RTA1 and the Data Marketplace also make use of the C-Roads IP-Based Interface in this example. This shows several options for V2N use cases assuming an RTA has the required information on the backend (N) side or would like to receive it. The latter would be true for all three user stories of the ACCA Use Case where the RTA should also be informed about hazards. Despite others, the RTA should get the information for the backend detection variants of User Story 2 and 3 so it can be disseminated to all OEMs. So, the following options exist for V2N/N2V:

- RTA1 has the information through the NAP and provides it to the OEM-specific components in the MEC-hosted Geoservice, if the according integration (upper blue question mark) was done
- Same as above, but RTA1 got the information from somewhere else than the NAP, e.g. the Data Marketplace (not shown in the figure) or the C-Roads IP-Based Interface
- The OEM backend on the public Internet obtains the information through the C-Roads IP-Based Interface and forwards it to the MEC-hosted Geoservice, and from there to the OEM-specific component (figure shows this for OEM1); different options exist how the hazard information was injected into the IP-Based interface:
  o From another OEM backend
  o From a MEC-hosted Geoservice integrated towards it (lower blue question mark)
  o From a Data Marketplace
  o From a NAP
  o From other RTAs than the one operating the NAP
- Same as above, but the information comes from the Data Marketplace (shown for OEM4); different options exist how the Data Marketplace could have obtained the information in a similar way as listed for the IP-Based interface

What is hardly covered so far, besides with the option of having a common component connecting the two OEM-specific ones, is the case of V2V communication across different OEMs. The options listed for V2N above can all be used for V2V if the hazard information is propagated to the OEM backend on the public Internet. Integrating all possible OEM-specific components to each other is likely too complex. There are not as many OEMs as RTAs, but it can still result in too many combinations to be feasible.
Figure 3-36 shows three options how interoperability for V2V communication can be achieved within one MEC-hosted Geoservice. The left option corresponds to what was done in 5GCroCo for RSA and PSA. In 5GCroCo the interfaces of the OEM-specific components were adjusted a little bit. When strictly following this solution the Geoservice software provider would have to integrate towards the OEM-specific interfaces. In the middle a solution is shown where the blue message forwarding component provides a common interface and the OEMs integrate their components towards it. It could be the same or a similar interface as used in the public Internet hosted backend, e.g., for C-Roads IP-Based Interface or a Data Marketplace, but other options are possible. It is not necessary that only one such interface exist. Having a few can still result in less effort than having to deal with all combinations of OEM-specific interfaces. The solution on the right depicts the option where OEMs agree on a common interface Geoservice software provider can integrate towards. It is again possible that more than one such interface exists when several OEMs agree on a common one. This is still reducing the number of combinations.
4 Summary and Way Forward

This deliverable has presented the final application architectures for the three 5GCroCo use cases. Its main purpose is to describe what is implemented in order to conduct trials for KPI measurements. Besides that, it also contains discussions for each use case if anything would have to be different in a real-world architecture than how it is done in the trials and who would be potentially responsible for the components needed to operate the use case.

Substantial parts of the architecture have been already implemented and are being evaluated with results published in Deliverable D4.2 [17] which is being updated as further trials are conducted. Deliverable D4.1 [11] describes which features are now being implemented to be available for the second round of trials planned for mid of 2021.
5 References


A Methods of Georeferencing and Mapping to MQTT Topics

A.1 Fixed Quantified Latitude and Longitude
A simple method of georeferencing works with latitudes and longitudes and a fixed quantization, e.g., to full degrees and five-minute steps. Each longitude and latitude degree are then divided into 60 / 5 = 12 parts numbered from 1 to 12. Corresponding MQTT topics would be:

.../.../[0…180]/[1…12]/[W|E]/[0…90]/[1…12]/[N|S]

Depending on geographic location, this maps into rectangle sizes of few kilometres.

A.2 Quadtree
The principle is to calculate keys that represent tiles in a quadtree grid. This system is used by Bing Maps [29] and Here [30] under the name of quadkeys and also mentioned in the C-Roads Platform hybrid communication specifications [28]. A Java implementation is publicly available.\(^\text{25}\)

As shown in the following figure, each tile of the quadtree grid has a unique quadkey. The length a quadkey corresponds to the zoom level, and the quadkey of a tile always starts with the quadkey of its parent tile. In Figure A-1, tile 2 is the parent of tiles 20 to 23, and tile 21 is the parent of tiles 210 to 213.

\(^\text{25}\) https://github.com/passchieri/Hybrid-IF2
These properties are compatible with the hierarchical pattern of MQTT topics, which renders the integration of quadkeys directly into MQTT topics very easy, allowing publishing and subscribing into specific tiles.

For example, to subscribe to all DENM messages in the tile with quadkey number 12022, the following topic extension can be used:

```
... / ... / 1 / 2 / 0 / 2 / 2
```

If a DENM is published in the tile with the quadkey 120220 (one of the child tiles of tile 12022), then it will be received by users that have subscribed to it and all its parent tiles as well (12022, 1202, 120, 12 and 1).

### A.3 Other

Below is a list of further methods of georeferencing that can be applied. Other methods are not precluded.

- **Civic Addresses**: Logical areas can be referenced by their civic addresses. Motorways e.g. can be referenced by their identifier, usually a unique number within a country, and a quantified offset, e.g. 10 km long segments. Different zoom levels can be applied to support segments of different sizes.

- **S2 Cells**: S2 cells provide an alternative to Quadtree to uniquely reference the whole world and have unique area identifiers at different zoom levels. More information is provided here [31].
# B Common Legend for Architecture Figures

## Table B-1: Legend of Architecture Figures

<table>
<thead>
<tr>
<th>Icon / Graphical Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>——</td>
<td>Transport link(s) within a local or wide area network using IP and typical, unmodified lower layer protocols like Ethernet</td>
</tr>
<tr>
<td>———</td>
<td>Control plane interface specified by 3GPP</td>
</tr>
<tr>
<td>———</td>
<td>Proprietary interface not defined by normative 3GPP specifications. It can be defined by other bodies or commonly used software implementations.</td>
</tr>
<tr>
<td>— — —</td>
<td>Optional transport links</td>
</tr>
<tr>
<td>——</td>
<td>Crossing links are connected</td>
</tr>
<tr>
<td>— — ——</td>
<td>Crossing links are not connected but there was no other feasible option to draw it without this overlap</td>
</tr>
<tr>
<td>APP</td>
<td>Application or application component hosted deployed in public Internet or vehicle</td>
</tr>
<tr>
<td>APP (MEC)</td>
<td>Application hosted in MEC</td>
</tr>
<tr>
<td>Device</td>
<td>(Embedded) hardware device that can optionally host application components</td>
</tr>
</tbody>
</table>