AUTOMOTIVE EDGE COMPUTING CONSORTIUM

Digital Twin Use Cases for Automobiles

Version 1.0 • April 2025

Contents

Foreword

“Digital Twin Use Cases for Automobiles” was prepared by the Digital Twin Special Interest Group of the AECC’s Use Case Development Working Group.

The purpose of this document is to provide the AECC’s vision on the key use cases, enabled by the coupling of edge computing technology and digital twin solutions. In addition to the definitions and high-level descriptions of automotive digital twin use cases, it also offers discussion on technical requirements, reference deployment patterns and challenges to be addressed. This document is positioned as a supplement to the AECC’s “General Principles and Vision” white paper [1] and the “Digital Twins” white paper [2].

The AECC is a not-for-profit association of industry members, dedicated to promoting edge computing and communications technology in the connected vehicles ecosystem. For more information about the AECC and its publications, visit https://aecc.org/.

1. Terms and Definitions

Term	Definition
Twin Model	A model representing digital twin(s) needed for a target use case
Digital Twin (DT)	A digital representation of an observed entity
Observed Entity	A real-world entity that may include information on relevant processes, events and environment parameters
Digital Twin Mobility Services	A service that leverages DTs to enable the scenarios defined in the AECC's "General Principles and Vision" white paper [1]

2. Introduction

Accurate digital representations of real-world entities play a vital role in simulations and prediction of complex systems. Future automotive services will focus on creating safer, more enjoyable and more eco-friendly mobility experiences. Data representing the status of roads, vehicles and other road entities needs to be collected to cloud and edge computing platforms to enable these services. It is also important to systematically structure, organize and aggregate the data, collected from heterogeneous sources. It facilitates more efficient data interpretation, which in turn helps gain stronger value from it. The Digital Twin is an emerging paradigm that aims to structure a myriad of data as digital reconstructions of physical entities.

Efficient data collection and sharing are becoming increasingly important in many fields, including the automotive domain. Until now, automotive data has been distributed in different silos, which do not necessarily reflect the relationship between the physical entities the data describes. A digital twin represents an observed entity in the digital space with high fidelity. Multiple instances of digital twins can interact and/or be combined with each other to enable simulations of complex systems (e.g., road traffic), consisting of many actors. Data representation in digital twins shall be intuitive to make it easier to leverage them to empower business.

This document explores possible use cases of digital twins in the automotive domain. While the automotive industry deals with a broad scope of services, we are particularly interested in on-the-road use cases rather than production and marketing. We examine three use cases that utilize different kinds of digital twins: road traffic optimization using road event digital twins, personalized cruise assist based on driver behavior digital twins and vehicle resource sharing services by vehicle digital twins. While examining the use case behavior, we also identify relevant stakeholders to be involved and what each stakeholder wants to achieve. From there, we formulate service requirements for each use case. We also discuss possible edge computing technologies available to address those requirements and identify technological gaps we need to address.

2.1. The Digital Twin in the AECC Context

Many organizations, including standards bodies, are giving attention to digital twins. Each organization may define digital twins differently, depending on the assumptions of target use cases. Those definitions often consist of the following aspects: an observed entity (OE), a digital twin (DT), a connection between OE and DT, and connections between multiple DTs. In the AECC, we define those terms as follows (Figure 1.).

Observed Entity (OE): a real-world entity that may include information on relevant processes, events and environment parameters

In the automotive context, this may include static/semi-static entities such as roadworks, road properties, potholes and snow cover, as well as dynamic entities such as vehicle movement, traffic surges and weather conditions in a target area of interest.

Digital Twin (DT): a digital representation of the observed entity

Connection between OEs and DTs: upstream and downstream communication paths between observed entities and digital entities

Connection between DTs: communication paths between a pair of DTs

Digital Twin Mobility Services: mobility services that leverage DTs to enable the use case scenarios mentioned in the AECC's "General Principles and Vision" white paper [1]

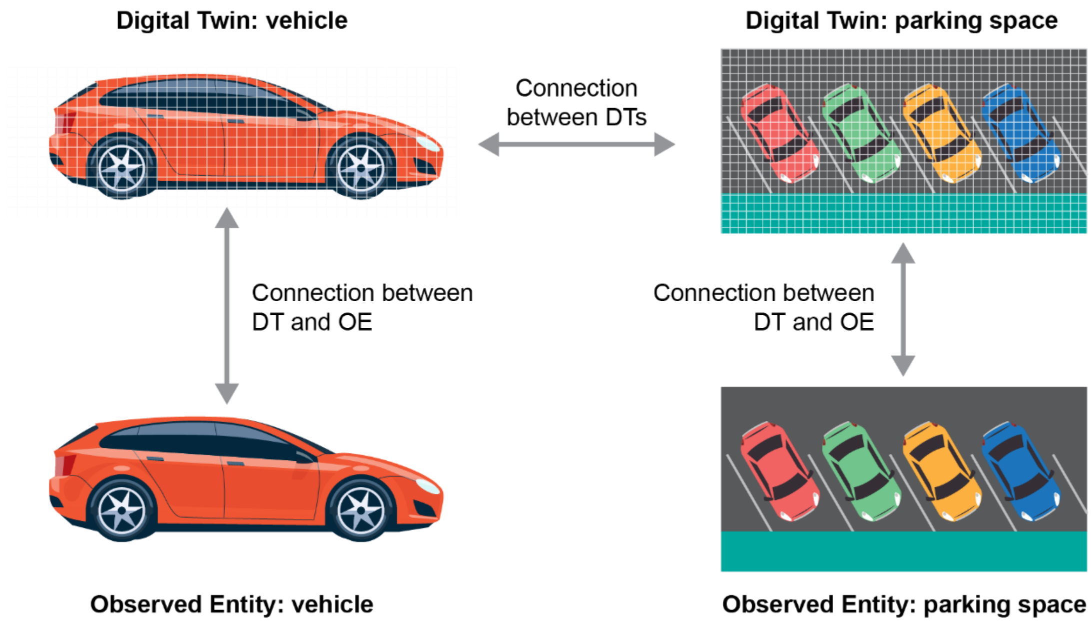

Figure 1. The AECC's Definition of Digital Twins

2.2. Objectives of This White Paper

In the AECC's vision, edge computing is positioned as a core driver of efficient big data processing. To realize the vision, the AECC has proposed a wide range of solutions, including the following items:

- Edge Data Offloading
- MSP Server Selection
- Vehicle System Reachability
- Access Network Selection
- Provisioning and Configuration Updates
- Opportunistic Data Transfer

These solutions were proposed and endorsed based on the needs that were identified through in-depth analysis of target service scenarios, defined in the AECC's "General Principles and Vision" white paper [1]. To further accelerate the implementation and deployment of edge computing, it is essential to create stronger awareness in different vertical industry sectors of the power of data processing and communications at the edge. Among a wide spectrum of service scenarios, this white paper particularly focuses on automotive "on-the-road" use cases.

In this paper, we want to inform stakeholders of the key challenges in implementing edge computing as a building block of their automotive digital twin solutions. This document analyzes the processes involved in on-the-road automotive use cases utilizing digital twins. By analyzing the processes involved, we aim to clarify the underlying infrastructure needed to implement the system. This document also describes the application processes involved in the use cases and the service and technical requirements that are derived from them. We will then describe possible deployment patterns and challenges in edge-based implementations of automotive digital twin use cases.

3. AECC Digital Twin Use Cases

In this chapter, we describe use cases that can represent the behavior of digital twins for connected vehicles. Three use cases are introduced to showcase different types of digital twins: road event digital twins, driver digital twins and vehicle digital twins. These types of digital twins have different behaviors, yielding separate sets of requirements on underlying edge computing platforms. The following sections describe interactions between the digital twins and each actor involved in the use cases.

3.1. Road Traffic Optimization by a Road Event Digital Twin

3.1.1. Use Case Description

In this use case, road events are collected, and digital twins of the streets are maintained to represent the current conditions of the roads. The data can also be accumulated to serve as a historical reference. A static map is often used as a basis to structure digital twins of road events. Road events are detected and collected using in-vehicle sensors as well as roadside sensing infrastructure. The types of sensors used for digital twin data collection include cameras, GPS, environment sensors (e.g., air temperature) and LIDARs. These events can be static, such as newly installed traffic signage; dynamic, such as road work; or ephemeral, such as street-parking spaces.

The interactions between digital twins of road events can be used to run a simulation for in-depth understanding of road traffic conditions. The result of this simulation can then be fed back to a traffic control system to realize a more favorable traffic condition by optimally guiding vehicles and/or controlling traffic lights. This use case requires creation and maintenance of digital twins, road traffic simulations using the road event digital twins and a mechanism to trigger various types of actuations to improve the road traffic. The overview of use case is presented in Figure 2.

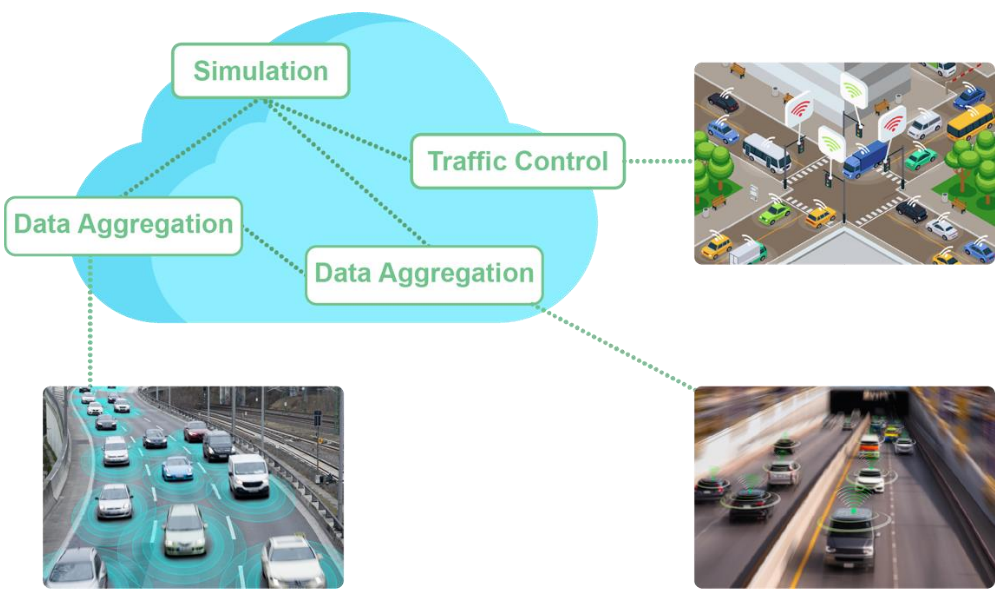

Figure 2. Overview of the Road Traffic Optimization Use Case

3.1.2. Actors

Road Traffic Equipment Operator (Client)

A road traffic equipment operator (client) wants to

- be able to monitor road traffic conditions when needed
- control road equipment (e.g., traffic light, digital speed signs, etc.) to actuate in time to enable safer and more efficient road traffic

Road Traffic Simulation Operator

The road traffic simulation operator wants to:

- simulate future traffic conditions on the fly to optimally control road traffic equipment
- be able to gather data from roadside devices and vehicles on the road while maintaining the privacy of road users

3.1.3. Expected Process Flow

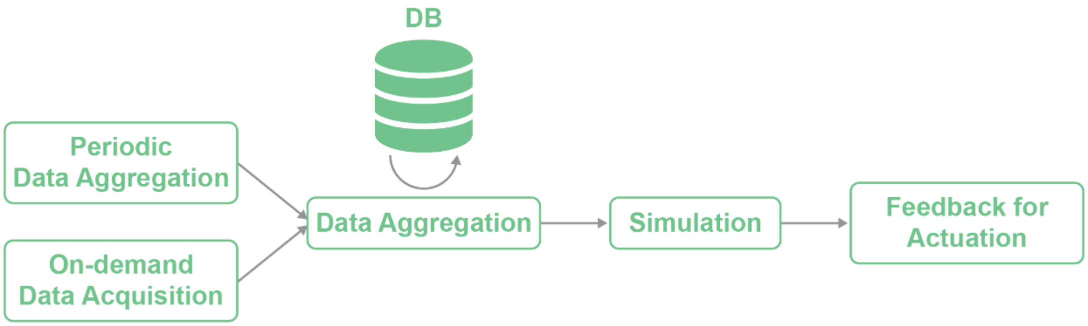

Figure 3. Process Flow of the Road Traffic Engineering Use Case

We show the digital twin process flow of use case in Figure 3. Each component is introduced as follows:

Data Acquisition

Table 1 shows the types of data collected to create digital twins of road events.

Table 1. Data Sources for the Road Traffic Optimization Use Case

Data Type	Data Source
Vehicle status (position, speed, acceleration, steering wheel angle, etc.)	• Onboard sensors (e.g., GPS, wheel speed sensors, accelerator/brake pedal positions, etc.)
On-the-road geolocation events (roadworks, accidents, potholes, etc.)	• Onboard sensors (e.g., cameras, radars, etc.) • Roadside cameras • HD map maintained in edge/center servers
Weather conditions	• Onboard sensors (e.g., windshield wiper status, temperature sensor, etc.) • External information sources (e.g., meteorological agency)

Two types of data acquisition schemes can be used in the road traffic optimization use case, the first one being *periodic* data acquisition, and the second option being *on-demand* data acquisition. The on-demand data acquisition requires bi-directional interactions between the road traffic optimization service and data sources.

Data Aggregation is a process of collecting data on road events from several vehicles and edge sites and accumulating historical data so that sufficient data can be attained to run a simulation of road traffic. Data aggregation from several edge sites can be challenging with respect to meeting latency requirements, as well as regarding data correctness when data comes from heterogeneous sources.

The **simulation** process creates an inference model that interacts with digital twins to find the outcome of a series of road events. This simulation will produce recommended actuation feedback to enable road traffic optimization. Simulations on distributed edge servers are particularly challenging as they should be completed in low latency to ensure timely actuation.

Feedback for actuation is a process of sending actuation control commands to relevant actuators. Timely delivery of the actuation feedback is particularly challenging if the actuator is a vehicle traveling on roads.

3.1.4. Data Characteristics

To characterize the communication requirements of the road traffic optimization use case, we assume that a vehicle, equipped with four cameras, constantly sends video streams to a center server and/or edge servers to keep the road event digital twins synchronized with actual roadway conditions. The video streams are analyzed on the server side to monitor the roads that the vehicle is traveling on. The vehicle's movement data is also sent to the server as part of the controller area network (CAN) data. In this requirement analysis, we assume that the data is uploaded to the servers without any preprocessing or compression.

Road traffic simulations take a series of the collected data as input to simulate future road traffic for a 10-minute time horizon. The actuation commands, however, may need to be sent to actuators more frequently in case of anomalous road traffic conditions (e.g., traffic accidents). The data characteristics are summarized in Table 2.

Table 2. Data Characteristics in the Road Traffic Optimization Use Case

Upload	Data Volume	CAN/GPS data: <1KB/data Camera data: <10MB/frame (from 4 cameras) [1]
	Frequency	5 seconds-10 minutes[2]
Download	Data Volume	Road traffic equipment control data: <1MB
	Frequency	5 seconds-10 minutes[2]
	Latency	<100ms

3.2. Personalized Cruise Assist by a Driver Digital Twin

3.2.1. Use Case Description

Adaptive Cruise Control (ACC) is a driver assistance system that automatically maintains a designated cruising speed while keeping a safe headway distance from a preceding vehicle. The personalized cruise assist use case is shown in Figure 4. The Distance to the preceding vehicle is measured by onboard ranging sensors (e.g., radars, stereo cameras, etc.) of the ego vehicle. If the headway distance drops below a set threshold, the ACC system automatically decreases the cruising speed to recover a safe gap. Typically, ACC systems offer just a few configuration options for keeping the headway distance (e.g., *short*, *medium* and *long*). However, some drivers may feel that even the longest headway distance option is too aggressive, while some others may feel that even the shortest option is too conservative. Also, even the headway distance preference of the same driver may vary depending on the driving conditions (e.g., weather, road traffic, behavior of the surrounding vehicles, etc.). The traditional ACC systems can barely consider such driver preferences.

The personalized cruise assist solves this problem by learning the headway distance preferences of individual drivers from the history of their driving behavior. The training data is collected when vehicles are manually driven by a human driver(s) without engaging ACC. The data is then analyzed by a machine learning framework to model the driver's habits on headway distance under various driving conditions. When ACC is engaged, the system observes the current driving conditions by onboard sensors and employs the personalized headway distance preference model to estimate the driver's preferred distance in the given context. The cruising speed is adjusted to maintain the preferred headway distance.

[1] [Assumption] Still image: 2Mpixels * 3Bytes (color) * 1/4 (Lossless JPEG), generated every second. One front-facing camera is used. Average total travel time per day: 30 minutes.
[2] Road traffic simulations are typically run to predict the future road traffic conditions for a 10-minute time horizon, but more frequent updates may be needed on anomalous road traffic events (e.g., traffic accidents).

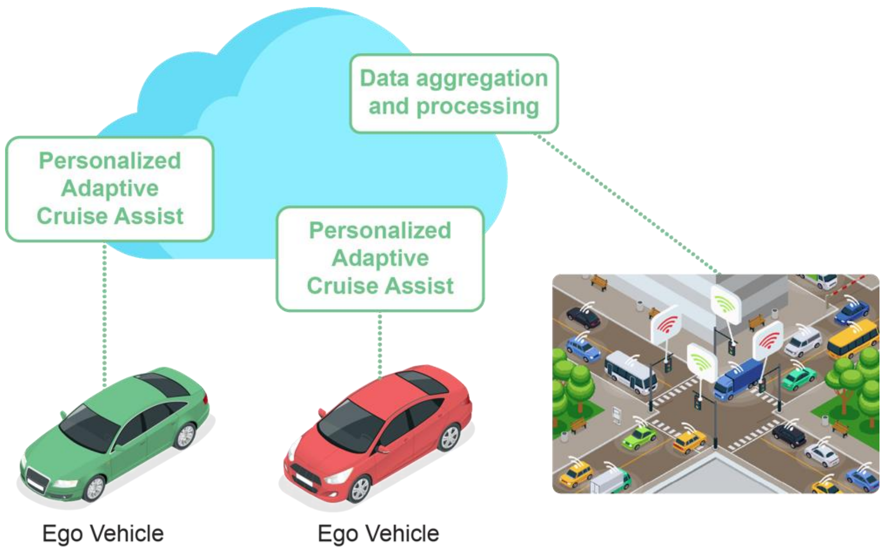

Figure 4. Overview of the Personalized Cruise Assist Use Case

3.2.2. Actors

Actors involved in the personalized cruise assist use case are a vehicle driver (client), mobility service provider and infrastructure provider.

Vehicle Driver (Client)

A vehicle driver wants to:

- experience a personalized cruising assist system that is convenient and reliable
- provide personal data to the mobility service provider in a secure manner to protect the client's privacy

Mobility Service Provider

The mobility service provider wants to

- offer a safe and reliable cruise assist system
- retrieve and process the client's data securely
- improve its service by updating its AI models without sacrificing its client's privacy
- collect and process the driver's behavior data, as well as transmitting the cruise assist guidance reliably and efficiently

Infrastructure Provider

The infrastructure provider wants to

- provide reliable and secure infrastructure to the mobility service provider and vehicle driver
- be able to provide the infrastructure required for the personalized cruise assist service to vehicles in motion

3.2.3. Expected Process Flow

Personalized cruse assist involves five main function blocks: Data Acquisition, Training, Driver Recognition/Classification, Inference and Vehicle Control. They are connected, as shown in Figure 5.

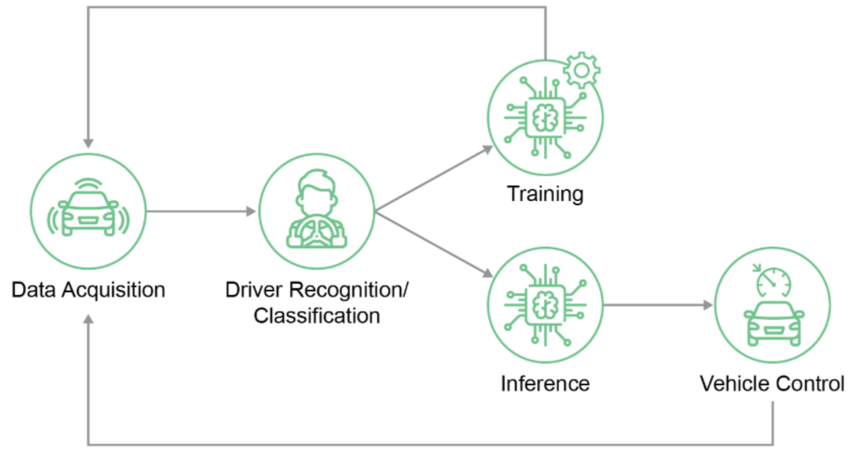

Figure 5. Process Flow for the Personalized Cruise Assist Use Case

Data Acquisition is a process of collecting sensor data from onboard sensors of a vehicle system and/or data from external sources at regular time intervals. Typically, the data is collected when a human driver manually drives a vehicle without engaging ACC. Examples of collected data are listed in Table 3.

Table 3. Data Sources for the Personalized Cruise Assist Use Case

Data Type	Data Source
Ego vehicle status (position, speed, acceleration, steering wheel angle, etc.)	• Onboard sensors (e.g., GPS, wheel speed sensors, accelerator/brake pedal positions, etc.)
Road traffic conditions (e.g., position, speed and orientation of the surrounding vehicles)	• Onboard sensors (e.g., cameras, radars, etc.) • HD map maintained in edge/center servers
Weather conditions	• Onboard sensors (e.g., windshield wiper status, temperature sensor, etc.) • External information sources (e.g., meteorological agency)

Training refers to a process of analyzing the collected sensor data by a machine learning framework to learn personalized headway distance preference models. The model can be trained per driver, but it may require a large amount of communication and computing resources. As an alternative solution, drivers with similar driving styles may be grouped, and the headway distance preference models may be trained on a per-group basis.

Driver Recognition/Classification is a process of recognizing a vehicle's driver so that the personalized cruise assist system can identify the headway distance preference model that fits the driver. If the headway distance preference model is trained on a per-group basis, the system classifies the driver into one of the predefined groups and applies the preference model trained for the selected group.

Inference is a process of estimating a driver's preferred headway distance in the current driving conditions. The Data Acquisition function block observes the driving conditions and is used as input for the inference.

Vehicle Control is a component to control a vehicle's acceleration and deceleration to maintain a set cruising speed while keeping the preferred headway distance to a preceding vehicle (if any).

3.2.4. Data Characteristics

We consider an example system configuration where headway distance preference models are trained on a center server and/or edge servers, while keeping the Inference function block on a vehicle system. When a vehicle is manually driven without engaging a cruise assist system, the vehicle system records CAN data (e.g., acceleration/brake pedal inputs, steering wheel angle, relative positions of the surrounding vehicles that are measured by radar(s), etc.) and camera images. The recorded data is uploaded to the servers in a batch via cellular or Wi-Fi networks. The data characteristics are summarized in Table 4.

The servers train each driver's headway distance preference models, using the collected data as input. The radar measurements and image data can be used to perceive the vehicle's surrounding roadway conditions, while other types of measurements from CAN are used to identify the maneuvers that the driver made in reaction to the given conditions. By combining the multi-modal sensor data, the servers train a model that estimates the driver's preferred headway distance in various roadway conditions. The servers may update the personalized models on a regular basis and distribute the latest models to the vehicle systems. The model update is a delay-tolerant process and can wait for up to a few days to complete.

Inference, on the other hand, is a latency-critical process, because the personalized cruise assist system must perceive the surrounding roadway environment based on the sensor outputs and adjust the headway distance settings of the ACC system in a timely fashion. Since inference is locally performed on the vehicle system in this example configuration, it does not require any data to be communicated with remote servers.

Table 4. Data Characteristics in the Personalized Cruise Assist Use Case

Upload	Data Volume	CAN data: 7.2MB/day/vehicle[3] Camera image: 2.7GB/day/vehicle[4]
	Frequency	Four times/day/vehicle[5]
	Latency	A few days
Download	Data Volume	Inference model: 1 to a few 100 MB
	Frequency	A few days to a few weeks
	Latency	A few days

[3] [Assumption] CAN data: 100 floating point numbers (4 bytes each), sampled at 10Hz. Average total travel time per day: 30 minutes.

[4] [Assumption] Still image: 2Mpixels * 3 bytes (color) * 1/4 (Lossless JPEG), generated every second. One front-facing camera is used. Average total travel time per day: 30 minutes.

[5] [Assumption] A vehicle makes four trips per day on average, with a total travel time of 30 minutes per day. The recorded sensor data is uploaded in a batch on each trip.

3.3. Vehicle Resource Sharing by a Vehicle Digital Twin

3.3.1. Use Case Description

In the vehicle resource sharing use case (Figure 6), we consider vehicle computing resource sharing, where vehicles offer computing services to the surrounding vehicles that fall short in their own computing power. The emerging Edge AI concept, where AI inference models are run on the edge computers, aims to improve the quality of services by using smaller, localized and/or personalized models. While the models are smaller, they still require a huge amount of computing resources. High-end vehicles may be equipped with powerful GPUs that can act as vehicle-based edge computers to assist the edge AI computation of other vehicles.

When a vehicle is able to share its resources, it informs an orchestration service, running on a center server or edge servers, about the available resources. The orchestration service is responsible for resource aggregation and orchestration among multiple serving vehicles, and it keeps the available vehicular computing resources in a resource pool. When the orchestration service receives a request for a computing service, it allocates required computing resources from the resource pool and deploys application services on the allocated resources. The service, hosted on the shared vehicle resources, can be accessed by other vehicles. The client vehicles are typically near the serving vehicle. When the serving vehicles need to take back their resources, they will inform the orchestration service. The orchestration service will then take the necessary steps to free and return the resources for private use.

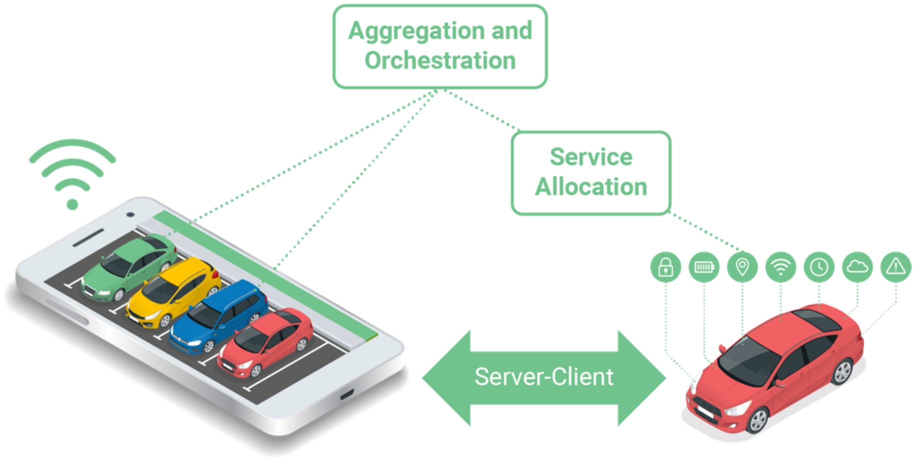

Figure 6. Overview of the Vehicle Resource Sharing Use Case

3.3.2. Actors

Infrastructure Provider

An infrastructure provider wants to:

- offer secure infrastructure without spending excessive capital to fulfill volatile market demands
- gather information on available vehicle resources that can be shared with other vehicles

Vehicle Owner

A vehicle owner wants to:

- lend the resources of the owner's vehicle in return for financial rewards
- lend the resources securely without compromising the safety of the vehicle
- be able to signal the infrastructure provider when the vehicle has computing resources available to contribute and when the driver needs to take back the contributed resources for the driver's own use

Mobility Service Provider (MSP)

An MSP wants to

- deploy its service on a reliable infrastructure at a reasonable cost without having to maintain the infrastructure
- uniformly deploy its service regardless of the physical configurations of the machines

Mobility Service User

A mobility service user wants to:

- have reliable and secure access to the mobility service

3.3.3. Expected Processing Flow

The process flow for the vehicle resource sharing is shown in Figure 7.

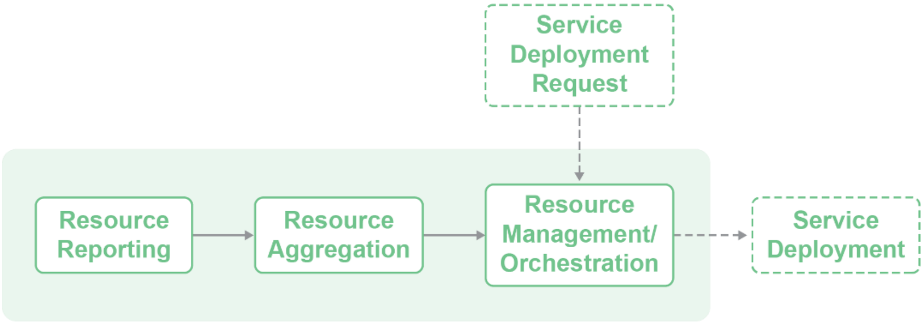

Figure 7. Process Flow for the Vehicle Resource Sharing Use Case

Resource Reporting refers to a process of reporting the computing resources available on the vehicle to the infrastructure provider. Vehicles also must communicate with the infrastructure provider when the resource will soon be unavailable so that the infrastructure provider can take steps to ensure continuity of service provisioning.

Resource Aggregation is a process to keep track of the available resources that can be used for resource sharing. This process utilizes digital twins. The information collected by this process includes:

- computing resource specification
- location of the serving vehicles
- available network connectivity
- available electricity/battery resources

Resource Management/Orchestration is a process to provide seamless infrastructure provisioning even when resources being used in the background change.

3.3.4. Data Characteristics

Table 5. Data Characteristics for the Vehicle Resource Sharing Use Case

Upload	Data Volume	Resource Data: <1MB
	Frequency	When a vehicle is parked and before it starts
	Latency	<100ms[6]
Download	Data Volume	Control data: <1MB Service image data: <100GB
	Frequency	Minutes-hours
	Latency	<100ms[6]

[6] The latency requirements are not stringent, as the vehicle is stopped and there will be a sufficient time buffer to gather information about available resources and to deploy application services.

4. Leveraging Edge Computing for Digital Twin Use Cases

In this chapter, we discuss possible deployment patterns of automotive digital twin use cases on edge computing platforms. We discuss the typical characteristics of the processes involved in each use case and typical ways to deploy them in a hierarchical edge computing architecture.

4.1. General Processing Flow

While each use case has different data characteristics, processing flows of digital twin use cases can be generalized as in Figure 8.

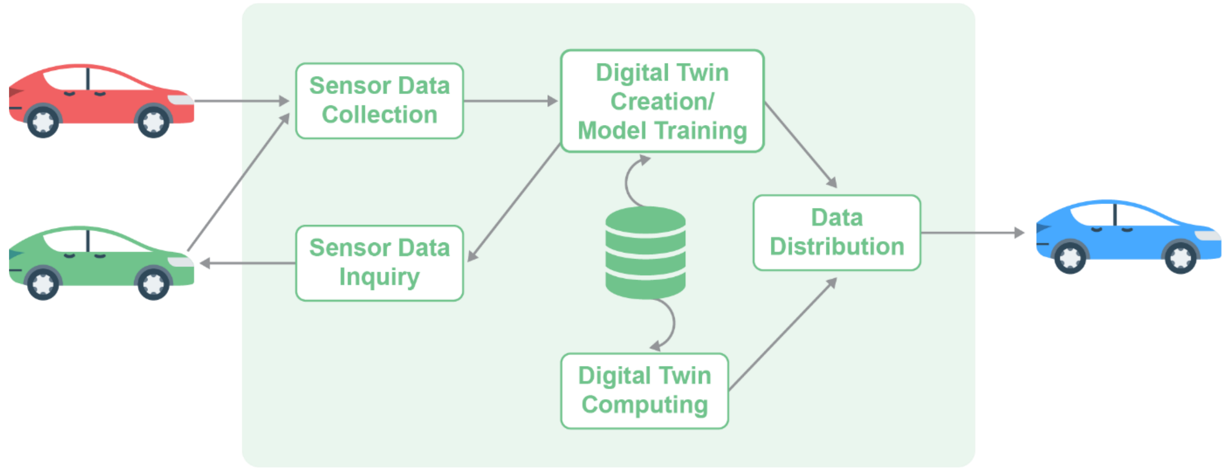

Figure 8. General Processing Flow of Digital Twin Use Cases

4.1.1. Sensor Data Collection

In automotive digital twin use cases, data is periodically collected from data sources. All use cases have their own versions of periodic data collection mechanisms. This process may include data verification as well as aggregation of several data sources. The process may also require low-latency communications between the User Equipment (UE; i.e., vehicles) and the processing site to ensure data freshness. Data processing will likely be hosted on vehicles or edge servers.

4.1.2. Sensor Data Inquiry

Sensor data inquiry refers to a process of collecting data by sending inquiries to vehicles that generate and keep the sensor data. This process requires that the vehicles be reachable by network. The request is typically made during the data aggregation process or when the digital twin model is being trained. This process is likely to be hosted on an edge server.

4.1.3. Digital Twin Creation/Model Training

This process takes the collected data as input to train a model constituting the digital twins. It typically requires a large amount of data processing power. The process is typically hosted on a center server, but it may alternatively be hosted on edge servers in consideration of privacy protection and regulatory compliance.

4.1.4. Digital Twin Computing

This process uses digital twins to make observations, insights and predictions required for individual use cases. Depending on the use cases of interest, this may be a simulation, inference or orchestration process. This process may use recently collected sensor data, historical data or a combination of both to predict future events. It typically requires a large amount of computing resources, while the simulation results may need to be generated in low latency for timely feedback of actuation commands. Depending on the resource requirements, the urgency of the simulation results and the number of actuation targets, this process can be hosted on center server(s), edge server(s) or vehicle(s).

4.1.5. Data Distribution

Data distribution is an additional process that may be needed if the results of simulations and/or inference need to be distributed to one or more actuators. This process is typically triggered by center server(s) or edge server(s).

4.2. Deployment Patterns

4.2.1. Deployment Pattern of the Road Traffic Optimization Use Case

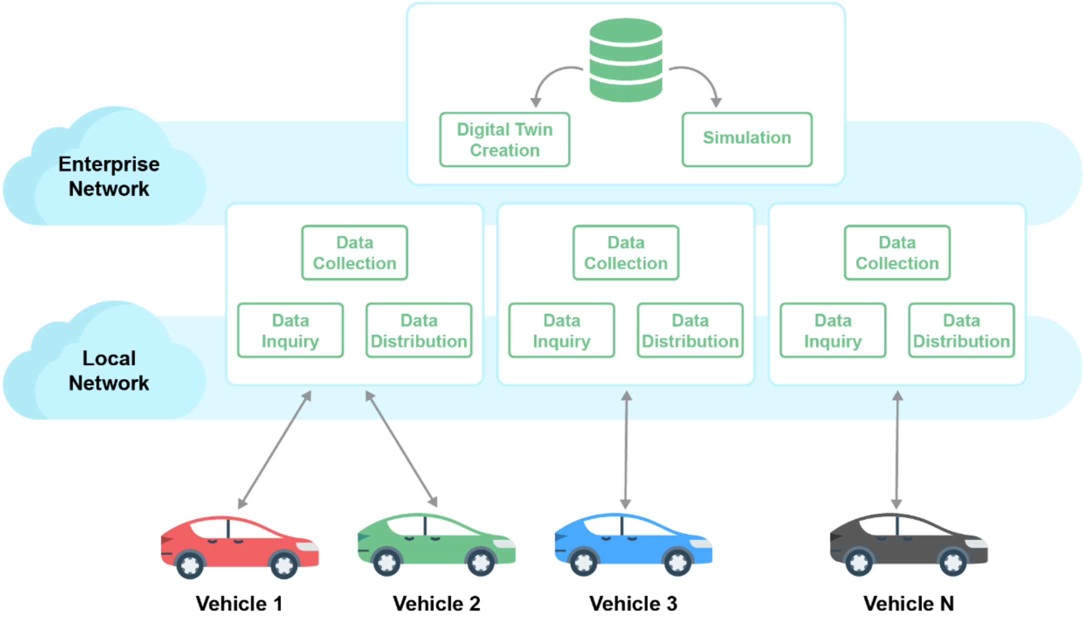

Figure 9. Deployment Pattern of the Road Traffic Optimization Use Case

Figure 9 illustrates a typical edge computing deployment pattern of the road traffic optimization use case. The use case requires aggregation of data in edge server(s) to enable road traffic simulations of a target area of interest. Processes that require networks to be reachable by vehicles, such as data inquiry and data distribution, are also hosted on the edge servers. Therefore, management of network traffic to/from vehicles is likely done in the edge servers that are responsible for the corresponding service areas.

A center server hosts processes that span several edge servers. In this sample scenario, digital twin creation and simulation processes are done on the center server as they will need data from multiple edge servers. Typically, these processes also require more significant computing resources, which is a requirement that the center server is suitable to fulfill. These processes also trigger data inquiry and distribution to request and send data from/to vehicles.

4.2.2. Deployment Pattern of the Personalized Cruise Assist Use Case

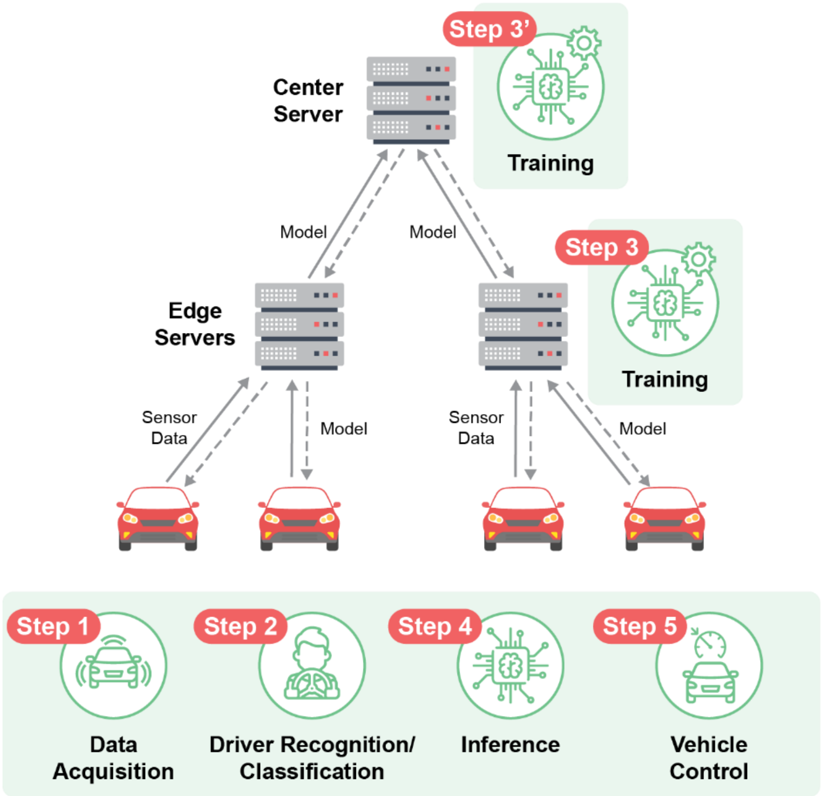

Figure 10. Deployment Pattern of the Personalized Cruise Assist Use Case

A typical deployment pattern of the personalized cruise assist use case is illustrated in Figure 10. The steps of deployment pattern are as follows:

- **Step 1**: The vehicle systems acquire sensor data.
- **Step 2**: The analysis of the series of measurements to recognize/classify a driver.
- **Step 3**: The vehicle system uploads the acquired sensor data and the identity/class of the driver to an edge server that trains/updates the headway distance preference model. Each edge server manages the model training of vehicles in a specific geographical region. After the model is trained/updated, the edge server distributes the latest model to the corresponding vehicle.

  - **Step 3'**: Optionally, the edge servers may upload the trained models to a center server. The center server may aggregate the collected models for fine-tuning and send the refined models back to the edge servers. Such model aggregation across multiple edge servers would facilitate model improvement.
- **Step 4**: The vehicle system then loads the headway distance preference model that is personalized for the driver and takes the sensor data as input to the selected model to estimate the driver's preferred headway distance under the current roadway conditions.
- **Step 5**: The vehicle system automatically controls the acceleration and deceleration such that the selected headway distance is kept.

Distributed training across multiple edge servers facilitates efficient handling of computation and communication workloads and is a possible solution for complying with regulatory requirements on data privacy. Since the sensor data uploaded by the vehicle systems is processed on an edge server in the same locality, privacy-sensitive data will not be transferred across the border of regulatory domains.

4.2.3. Deployment Pattern of the Vehicle Resource Sharing Use Case

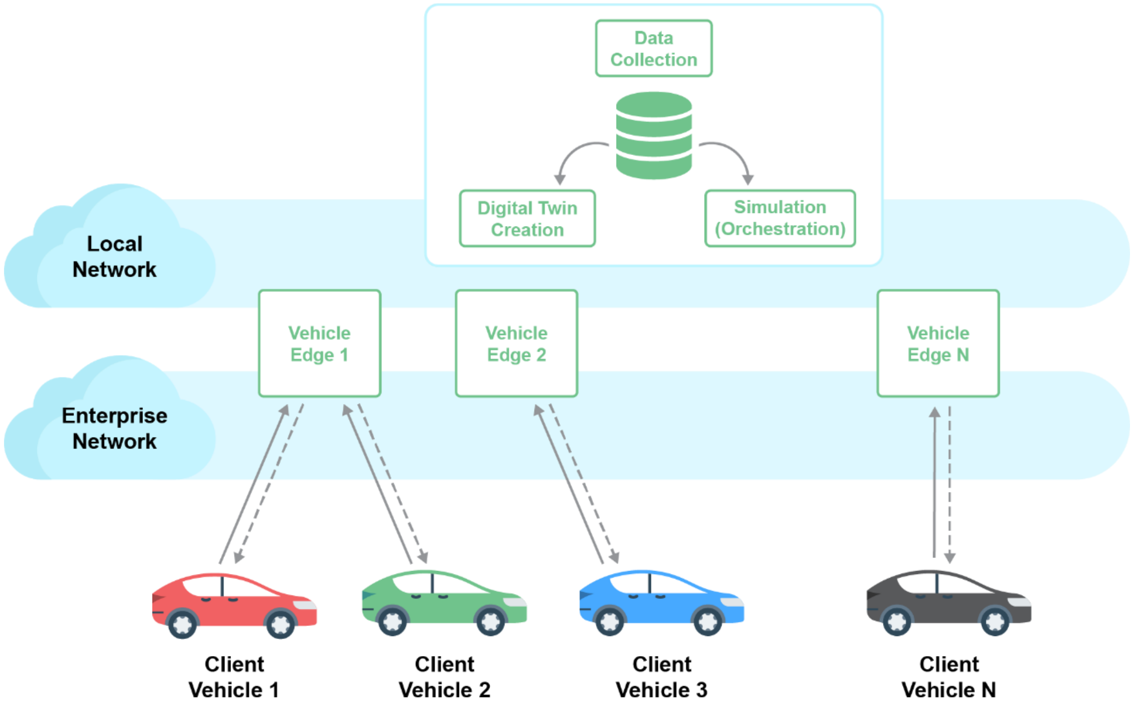

Figure 11. Deployment Pattern of the Vehicle Resource Sharing Use Case

The vehicle resource sharing use case is unique in that it does not employ typical hierarchical process distribution across center servers and edge servers, as vehicles themselves serve as edge computing resources in this case. It is assumed that the resource aggregation and orchestration will be done in a central server that connects to the vehicle edges via networks. Once services are deployed on the vehicle edge, the client vehicles utilizing the service will access the vehicle edge by way of local or enterprise networks. An overview of this deployment pattern is shown in Figure 11.

An overlay network could be deployed on the enterprise network to allow the client vehicles to discover and access the services deployed on the vehicle-based edge servers. A mechanism to host a service on resources distributed across multiple vehicle-based edge servers is also essential to handle resource-intensive services.

5. Challenges in Edge-based Implementation of Automotive Digital Twin Use Cases

5.1. Network Architecture

Latency and Fidelity

The use cases described in this document require timely collection of sensor data that is periodically generated at data sources. The network must be agile enough to autonomously recover from potential failures (e.g., due to natural disasters) to meet the latency requirements with minimal downtime.

Vertical and Horizontal Network Integration

Managing end-to-end network fidelity is additionally complicated due to edge computing architecture's hierarchical and distributed nature. It is essential that networks be reachable between nodes, and the network should allow seamless connection and coordination between vehicles, gateways, servers and other nodes.

5.2. Computing and Storage Architecture

Heterogeneity of Edge Computers

Managing edge computers will be a complex task, as edge computers will come in many forms and specifications. Especially for the vehicle resource sharing use case, where fleets become not only consumers and producers of data but also computing resources, computing and storage architectures should be redesigned.

Seamless Data Sharing and Processing

As computation takes a major part of end-to-end latency in automotive digital twin systems, the computing architecture shall be designed to minimize latency overhead for data handling, such as by utilizing in-memory processing. Resources from multiple vehicles may need to be used in unison to achieve the computing power and latency requirements.

5.3. Orchestration and Application Development

Application Development for Personalized AI

AI processing is the core of creating digital twins. However, AI models often take up a large amount of storage, memory and processing resources; hence, they may need to be adapted to fit into the new edge computing architecture. While application offloading has been the center of edge computing application engineering, with the growth of AI, new paradigms, such as federated learning and more efficient model fine-tuning methods, are needed.

Orchestration of Hierarchical Applications

With the distributed and heterogeneous nature of edge computers, orchestration of edge computing resources is a challenging problem. The need for a seamless computing environment on a myriad of edge computers calls for new paradigms of application design and resource orchestration.

Interoperability of Digital Twins from Different Parties

There can be a variety of service providers (e.g., automakers, road infrastructure operators, navigation service providers, etc.), each maintaining its own versions of digital twins. A typical example is the road traffic optimization use case, where multiple service providers may employ sensor data from different sets of vehicles and roadside sensors to monitor the latest road traffic conditions. Let's say that service provider A collects sensor data from the vehicles of a certain brand, and maintains the road event digital twins on the edge servers that are operated by mobile network operator X. Another service provider, B, may collect sensor data from the vehicles of a different brand, and host its road event digital twins on mobile network operator Y's edge servers. To improve the fidelity of the digital twins, the service providers A and B may want to partner with each other to exchange their own information about road traffic status. To facilitate such interoperation of digital twins from different parties while meeting the requirements on communication and data processing latencies, edge servers from different edge computing infrastructure providers (i.e., mobile network operators X and Y) should have a capability of being paired with each other. There should also be a mechanism to absorb the difference in data format, data collection schemes and many other system configurations adopted by each party.

6. Conclusion and Next Steps

In this white paper, we introduced three automotive digital twin use cases that leverage edge computing.

1) Road traffic optimization by a road event digital twin
2) Personalized cruise assist by a driver digital twin
3) Vehicle resource sharing by a vehicle digital twin

Each use case is enabled by a different type of digital twin, each having different data processing characteristics. This white paper has elaborated the use-case-dependent data characteristics, processing flows and possible deployment patterns. We hope this serves as reference for future adaptation of edge computing for automotive digital twin applications with similar characteristics.

It is important to note that there are still challenges in implementing these use cases, leveraging the full potential of edge computing. On the network side, it is important to have a resilient and dynamic network that is able to provide seamless communication across computing nodes to maintain the fidelity of the digital twins. On the computing side, a data sharing and processing paradigm that enables seamless sharing of digital twin data across heterogeneous computing nodes constitutes a critical component. Lastly, a new application development and orchestration framework is needed to enable distributed services to run smoothly on edge computers.

The AECC will continue discussions of these topics to cultivate a cross-industry consensus on best practices to tackle implementation challenges for these use cases.

7. References

[1] Automotive Edge Computing Consortium, “General Principles and Vision (Paper) Version 4.0.4,” 2024. Available online: https://aecc.org/resources/publications/.

[2] Automotive Edge Computing Consortium, “Digital Twins (White Paper) Version 1.0,” 2024. Available online: https://aecc.org/resources/publications/.