AUTOMOTIVE EDGE
COMPUTING CONSORTIUM

Data Management Systems in the Distributed Environment

White Paper

Version 1.0 • October 2023

Contents

Foreword

The automotive industry is facing upheaval around connected vehicles and mobility services, and it seeks a way to properly manage data to build intelligent services and improve customer experiences. However, it is not easy. Simultaneous connections and data flows -- that are not only very large, but also fluctuating -- need to be handled, while maintaining target response time for each service and achieving a good cost-performance ratio, in order to process such massive data flows and stored big data so as to provide the desired intelligent services.

In that context, the AECC assumes a distributed environment, because it significantly decreases the data transfer burden over the network, and processes a huge amount of data more quickly in an environment closer to the running vehicle than in the centralized environment. However, it is apparent that good data management needs to be implemented when doing so, in order to guarantee global data consistency, streamline data movement and synchronization across environments, and support secure and rapid access to distributed data.

The problem of finding available parking spaces is used as an example in this document. That use case collects and analyzes data from running vehicles to detect available parking spaces and informs subscribers about them. Since there is an analysis step involved, existing solutions, such as content delivery networks, cannot be used. This sample use case is generalized so that high-level system behaviors can be discussed, with relevant data management requirements and recommendations to industry players who may try to build similar mobility services.

The AECC is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability in the connected vehicle ecosystem. For information about the AECC and AECC white papers, see http://www.aecc.org/.

1. Terms and Definitions

Term	Definition
AECC System	A logical system composed of one or more AECC services, mobility services, computing platforms and the networks that connect them to vehicle systems and other clients
AECC Service	A service that provides configuration and control of the components of an AECC system
Central Cloud	A cloud that provides a coordination service for distributed computing
Cloud	A logical, location-independent computing platform that hosts services to store, manage and process data, which is implemented on a set of remote servers
Computing Facility	A physical facility that houses computing infrastructure
Computing Infrastructure	Resources and services on which systems and services are built. *Note: This may include but is not limited to power, cooling, computing, networks and storage.*
Computing Platform	An environment in which data is processed. *Note: This refers to the hardware or the operating system (OS), even a web browser and associated application programming interfaces or other underlying software, as long as the application code is executed with it.*
Distributed Computing	Computing that divides an application into many tasks that can be served by many computers. *Note: Tasks in this context could be microservices.* *See also: ISO/IEC TR 23188:2020.*
Edge	Boundary between pertinent digital and physical entities, delineated by networked sensors, and actuators. *Note: "Pertinent digital entities" means that the digital entities that need to be considered can vary depending on the system under consideration and the context in which those entities are used.* *Source: ISO/IEC TR 23188:2020.*
Edge Cloud	A cloud providing edge computing

Term	Definition
Edge Computing	Distributed computing in which processing and storage take place at or near the edge. where the nearness is defined by the system's requirements. *Source: ISO/IEC TR 23188:2020.*
Mobility Service	A service provided to the passengers or the driver of a vehicle (e.g., telematics, traffic, map, ride-sharing, insurance) and external clients
Mobility Service Provider	A platform-independent provider that provides customers with one or more mobility services
OEM	It stands for "Original Equipment Manufacturer," which means a vehicle manufacturer in the automotive industry.
Service	Means of delivering value to clients by facilitating desired outcomes
Vehicle System	A system composed of a computing platform, applications, services and other components residing in the vehicle

2. Introduction

To make driving safer, traffic flow smoother, energy consumption more efficient and emissions lower, communication between running vehicles and data center services, efficient and timely data processing and dynamic information delivery from data center services to running vehicles are becoming more and more important. For instance, it is expected that the cost of high-definition (HD) map creation will be reduced by gathering the required information from commercial vehicles, and the arrangement of mobility services will be optimized by knowing a precise traffic and mobility demand situation.

With the rise of such information utilization scenarios comes a great need to collect and process an unprecedented amount of data, in a timely and low-cost manner. To realize such technological innovation, the AECC is studying distributed data processing by leveraging emerging edge technologies. It is expected that the network data transfer cost and response time to process data will be reduced as a result.

2.1. AECC Data Management Vision

To manage and process a tremendous amount of data from running vehicles effectively and efficiently, multiple geo-distributed computing platforms -- the edge cloud and the central cloud -- will be used. These platforms can potentially be provided by multiple vendors. In any case, with AECC solutions, distributed data is accessed seamlessly and managed in a unified manner, and the system resources and workloads can be easily orchestrated in a cost-effective manner. This should remain true even when workloads vary largely and when the platforms to be used are changed significantly.

Figure 1 below shows an image of such geo-distributed data management.

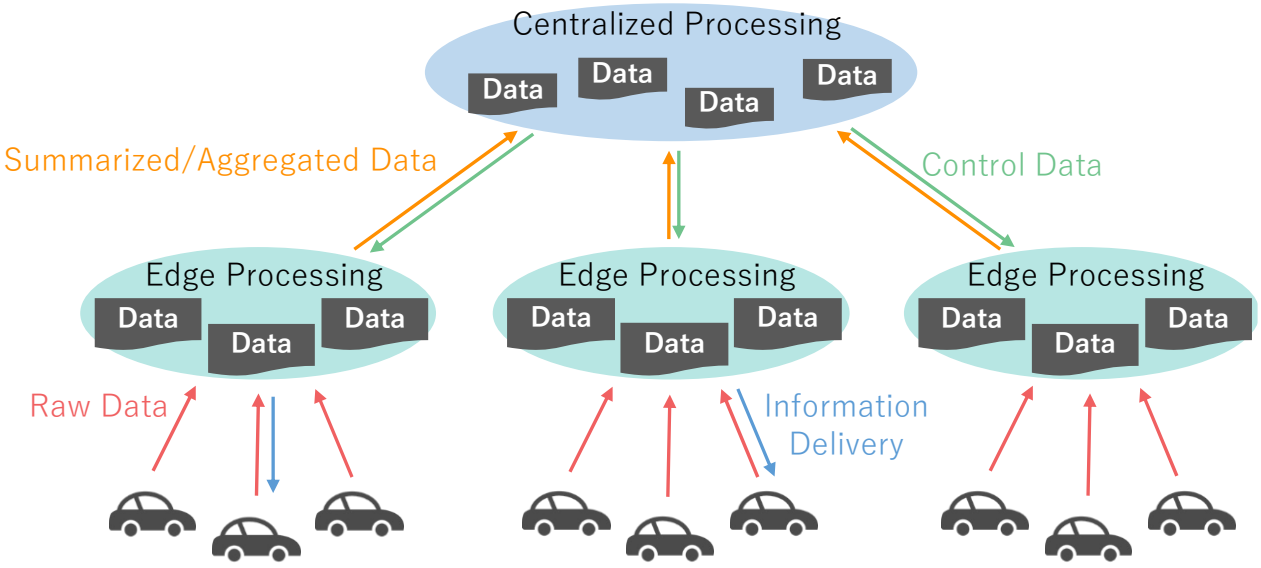

Figure 1. Distributed data processing in the AECC system

2.2. AECC Data Management System Context

As described in Figure 1, the AECC Data Management System processes and manages the data in three layers:

- **Vehicle Systems:** This layer is positioned at the bottom of the AECC architecture. Vehicles embed more sensors and electronic computing units for software-based control. All these sub-systems gather data locally and communicate to the outside world through 5G/4G and Wi-Fi connectivity.
- **Edge Processing**: This layer is positioned in the middle of the AECC architecture and supports the geo-distributed deployment of mobility service applications. These applications can use data from vehicle systems or other mobility services to provide various connected functionality to vehicle systems. These can also pre-process and relay data collected from vehicle systems before sending it for centralized processing. The edge layer can be of various natures and capabilities, depending on where they sit in the network.
- **Central Processing**: This layer acts as a control unit that coordinates distributed edge processing. For that purpose, aggregated/summarized data might be collected from the edge processing layer, and corresponding control data is delivered back from the central processing layer.

Note: With respect to the separation of the edge and vehicle systems layers, while processing capabilities embedded in vehicle systems can be considered as an edge processing capacity, these systems are mostly managed by OEMs and are out of the AECC's scope.

This tree-like structure can be mapped on top of many physical or logical deployments of the edge and cloud layers. At first glance, this seems to work well; however, there are various points to consider that do not create issues in centralizing all data processing tasks in only one central cloud.

Connected and autonomous vehicles are expected to generate a minimum of 300 GB of data per hour and vehicle. For fully autonomous vehicles, these amounts could go up to several terabytes per hour.

Assuming a fleet size of a million vehicles results in a required ingress capacity of several thousand terabytes per second, which is far above the capacity of a single data center or cloud provider. This huge data flow can be reduced by using multiple strategies:

1. Edge filtering by using data processing capabilities embedded in connected vehicles
2. Spreading the data transfers over multiple intermediate edge computing platforms with significant data processing capabilities

The focus of this white paper is on the second part. We need a specific analysis of the data management requirements for this large infrastructure because of the scale of the data and the complexity of the supporting infrastructure.

2.3. Objectives of This White Paper

Data on moving vehicles will be distributed on multiple edge computing platforms, so how can support for vehicle owners be provided? A regional HD map will be managed in the edge computing platform of a region, but how can data integrity over a wide area be ensured? These questions must be answered. The AECC is studying data management mechanisms to solve such problems and develop the necessary architecture. Accordingly, the objectives of this white paper are to describe the characteristics and a high-level structure of the AECC data management system, clarify the data management requirements and explain the expected solutions required for the AECC data management system.

In order to properly describe such an AECC data management system, this document first analyzes some application examples in Chapter 3 and generalizes them with respect to system structure in Chapter 4. Then, data management requirements are set forth in Chapter 5, and the core components of the AECC data management system are explained in Chapter 6.

3. AECC Application Examples

To provide a pertinent analysis for mobility services providers, we describe a set of application examples, which have in common the need to collect, store and process large amounts of data from vehicle systems or roadside units.

Such an example, described below, is an available parking space detection application. The goal of this service is to automatically locate available spaces that are suitable for vehicle parking and report these locations to vehicle drivers who subscribe to or request the service, by utilizing image sensors installed in inexpensive general-purpose vehicles. The available parking space detection service is particularly suited for dense urban areas where it might be difficult to locate and drive to an available parking space without distraction for the driver.

To detect available spaces, it is necessary to collect image data from vehicles sent to the remote computing platform and run visual analysis against it. As there are many roads in large cities, the amount of image data collected is also very large. In addition, it is desirable to detect available spaces and notify the vehicle drivers who are looking for them as soon as possible; therefore, the end-to-end latency should be minimized. We have taken the available parking space scenario as a starting point to see the typical requirements that must be met in the distributed environment.

Note: A fully automated driving vehicle will have a lot of computing resources, so it might be able to detect available parking spaces locally and just send the results to the remote computing platform. However, automated driving controls do not allow delays, so it may be difficult to have the extra image analysis process for other purposes. The additional power consumption also reduces the vehicle's range. In addition, automated driving vehicles are expensive, and it will take some time before they become widespread. Therefore, it is also necessary to design services that take into account manual driving vehicles and vehicles with simple advanced driving support systems.

In the subsequent sections, three deployment scenarios for the available parking space detection service are described.

- The available parking space detection service is operated by a single OEM (Section 3.1):
  In this scenario, we assume that a standalone available parking space detection service operated by a certain OEM leverages the connected vehicle fleet belonging to that OEM. This provides an opportunity to show the core functions of the service and how they are distributed over the architecture tree.
- The available parking space detection service relies on other services (Section 3.2):
  In this scenario, we assume that there is an available parking space detection service that relies on other available parking space detection services, which may be operated by other OEMs.
- Available parking space data is reused by other services (Section 3.3):
  The last scenario describes extended services to leverage available parking space detection data, such as an urban congestion report and roadside condition report.

3.1. Available Parking Space Detection Service Operated by a Single OEM

3.1.1. Scenario

In this scenario, it is assumed that there is an available parking space information service operated by a certain OEM that leverages its connected vehicle fleet and provides information about available parking spaces in a given area designated by the vehicle driver.

To provide the available parking space, the available parking space detection service gathers image data about the targeted parking spaces from its connected vehicle fleet, analyzes it to detect the status of parking spaces and updates the available parking space detection information as background processes. In urban areas during peak hours, there will be repeated available parking space detection requests; as a result, such data collection and analysis processes will be executed very frequently to provide rapid responses. On the other hand, in off-peak hours, this will be done in a request-based manner, to avoid meaningless data collection and analysis. Such frequency or timing of the background process is managed by the available parking space detection service itself.

The available parking space detection service then provides the available parking space information in the specified area to the driver in the foreground process, when it is requested by the driver, and before the driver arrives at the specified area.

Figure 2 shows the high-level behavior of such an available parking space detection service.

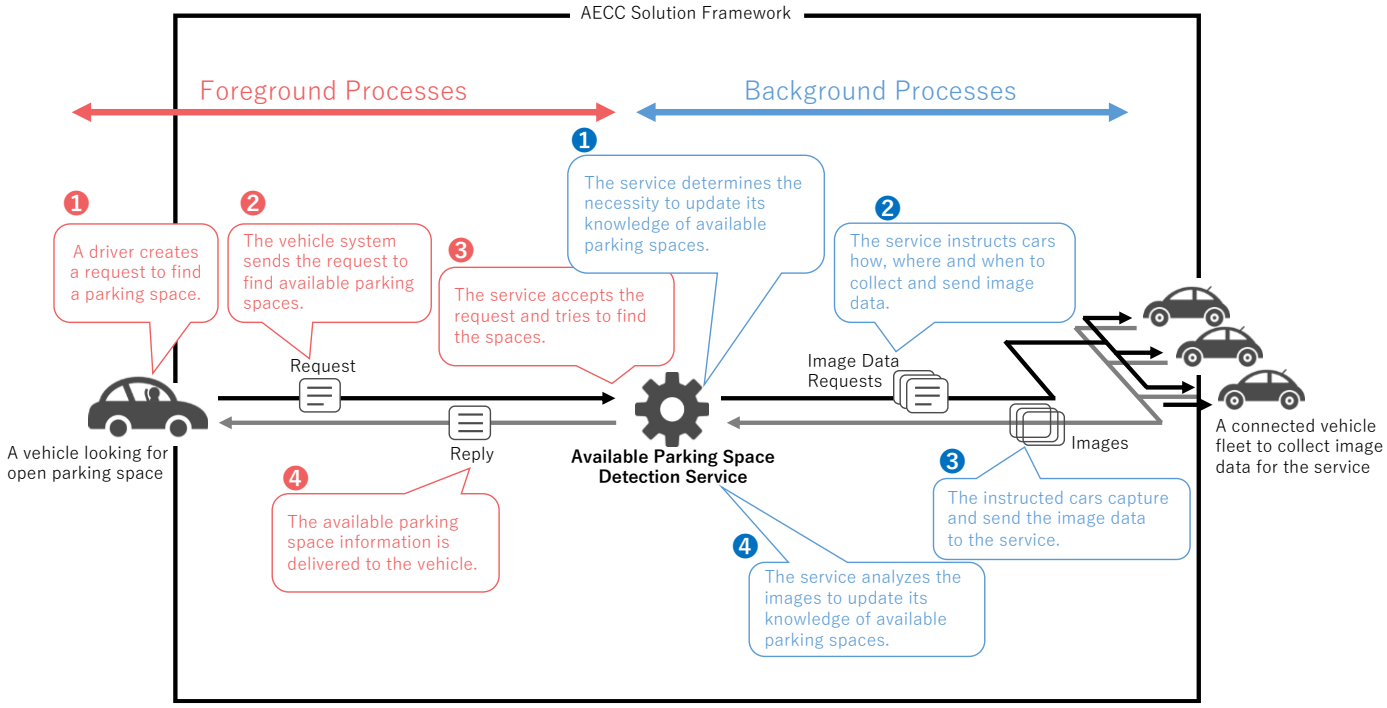

Figure 2. Service image of the available parking space detection service operated by a single OEM

A more detailed explanation of the system's behavior appears below:

Pre-conditions:

- The user has a valid registration and set of credentials to access the available parking space detection service.
- The connected vehicles are configured to interact with the available parking space detection service.

System Behavior:

Foreground Process

1. Drivers specify an area in which they want to find available parking spaces, and the time they want to park their vehicles.
2. The vehicle system sends a request to provide the available parking space information, specifying the area and time to the available parking space detection service.
3. The available parking space detection service checks available parking spaces in the specified area at appropriate times -- e.g., 30 minutes, 15 minutes, 5 minutes and 1 minute before the specified time -- or continuously, if the driver has already arrived at the area but the service could not yet locate a space.
4. The available parking space detection service responds to the driver with the identified available parking space information.

Background Process

1. The available parking space detection service manages the freshness of available parking space information and determines the necessity to collect image data to keep the available parking space information up to date and/or to respond to the driver's request.
2. The available parking space detection service selects the connected vehicles that will pass by target parking spaces and instructs them to send image data.
   Note: The parking space locations are pre-registered in the available parking space detection service.
3. The vehicle systems of the requested vehicles capture images in front of the target parking space and send them to the available parking space detection service.
4. The available parking space detection service receives the image data sent from the vehicles and analyzes it, then updates the status of the parking spaces of interest.

3.1.2. Deployment Model

To determine the deployment model for the scenario described above, it is necessary to collect and analyze a large amount of image data frequently to maintain high-freshness parking space information in the urban area and to respond to drivers with available parking space information as soon as possible when requested. However, today's cloud-based centralized data processing is expected to be problematic regarding increased data transfer costs and overall latency. Therefore, the AECC assumes that the analysis and processing of image data are performed in a very distributed manner, leveraging regional computing facilities, which eliminates communication bottlenecks and shortens the end-to-end latency.

However, such a massively distributed environment poses challenges in controlling data flow as well as protecting private information. Therefore, the AECC recommends implementing a gateway and interface that will:

1. Hide location and implementation details of the distributed services from the vehicle system and its users, because:
   a. It is very hard to keep configuration information for the distributed processing up to date for a massive number of connected vehicles.
   b. If the system is configured so that data analysis servers are assigned by location for the sake of data processing efficiency, eavesdropping on the data flow to the server is equivalent to leaking personal information.

2. Hide vehicles and other personally identifiable information (PII) data from the parking space application and its users, because:
   a. Many users do not want their personal information used to find parking spaces.
   b. The parking space detection application does not need to know individual vehicle and personal information.

Figure 3 shows an example of such a deployment model of an available parking space detection service operated by a single OEM.

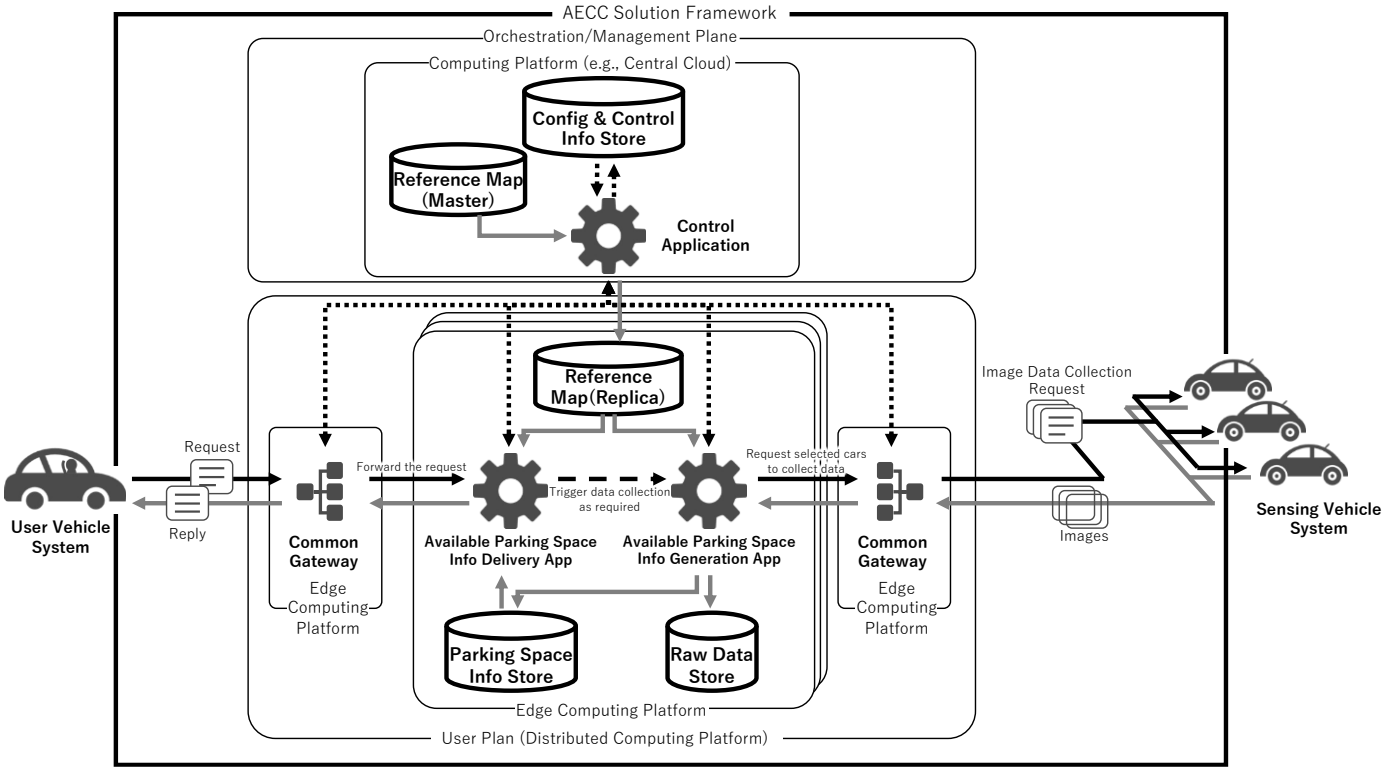

Figure 3. Deployment model example of an available parking space detection service operated by a single OEM

As described in the above figure, we assume that the available parking space detection service consists of the following system elements:

- **User Vehicle System:** A connected vehicle system subscribing to the available parking space detection service. It sends an available parking space information request to the available parking space detection service and receives the available parking space information from it.
- **Sensing Vehicle System:** A connected vehicle system configured to collect and send image data that has potentially captured an available parking space(s) based on the requests from the available parking space information generation app. To minimize the amount of data transfer, certain filtering mechanisms will be operated in the vehicle system before it sends data to the common gateway, such as confirmation of space using millimeter waves, etc. However, due to the resource limitations of the vehicle system, it will not be comprehensive and quite a lot of information will be sent without being filtered. In this scenario, we assume that the maker of the sensing vehicle system and the owner/operator of the available parking space detection service are the same entity.

- **Common Gateway:** This element sits in front of the available parking space information delivery app, the available parking space information generation app and other mobility service applications, if any, running on the edge computing platforms, to abstract the physical implementation; that is, to hide the execution environment of the available parking space detection service, and to protect privacy. Protecting privacy entails preventing the driver's location from being identified to the system by the connection information, and also preventing it from being exposed to the available parking space information delivery and generation apps, by replacing the driver or vehicle system ID with an ephemeral ID. All of the available parking space information requests sent from the user vehicle system and image data sent from the sensing vehicle systems must pass through this gateway. The placement of this common gateway needs to facilitate meeting service level requirements, such as response latency.
- **Available Parking Space Information Delivery App:** An application that provides the available parking space information to the subscribed drivers when requested. This application is configured to provide the available parking information in a timely manner to drivers who need it quickly. For this reason, this application might be executed in the edge computing platforms to provide short-latency services as well as to support a server-push notification for drivers who arrived at their destinations but could not locate a place to park the vehicle. This app also manages the freshness of the available parking space information, and if the information is stale, this app requests that an available parking space information generation app update the information.
- **Available Parking Space Information Generation App:** An application that collects image data from the sensing vehicle systems, analyzes them and updates the available parking space information. It is executed in the edge computing platforms in a distributed manner to minimize the amount of data transfer and perform the available parking space information updates with very low latency.
- **Parking Space Information Store:** It stores the availability status of all registered parking spaces.
- **Raw Data Store:** It manages the raw image data collected from the sensing vehicle systems. To guarantee privacy protection, all private information, such as vehicle registration plates and human faces, is mosaicked when the raw data is stored. It should be noted that, for the sake of completeness, a human process to confirm the mosaic processing may be in place before mosaicked raw data is exposed to external parties.
- **Reference Map Store (Master):** It manages the master data of the road structures and the targeted parking space locations.
- **Reference Map Store (Replica):** It manages a subset of the road structure data and the targeted parking space location data, which are required for the associated available parking space information delivery and generation apps. The content is replicated from the reference map (master).
- **Control Application:** An application manages and controls physical deployment; e.g., instances that run the common gateway, available parking space information delivery and generation apps and related responsibilities -- that is, the areas assigned to them -- and relevant data distribution. Such configuration information is delivered to the common gateway so that it can forward the available parking space information requests and/or the image data collected to the instances that run the available parking space information delivery or generation apps.
- **Configuration and Control Information Store:** It manages up-to-date overall system structure, such as placement of data processing, dependencies between data processing, data distribution and data flows, as well as policies and parameters to control structural changes to the system. This information is used to provision the available parking space detection service and to control its behavior.

3.1.3. Overview of Data Flows

A parking space is defined as an obstacle-free rectangular space at least 5.00 meters long and 2.6 meters wide. This free surface has to be located in an area where parking is allowed and should not obstruct any regulated or private access to nearby buildings (e.g., parking entrance/exit, firefighter access ramp, etc.). Information about authorized

parking areas could be obtained from local authorities and/or agencies and stored near the available parking space information delivery and generation apps as part of the reference map information.

By capturing images of the roadsides during cruising, a vehicle system can run an image processing algorithm to detect such free surfaces. While the definition of such image processing algorithms is beyond the scope of this white paper, it is assumed that a single vehicle cannot confirm the detection of a parking space with sufficient precision.

Therefore, we assume that the sensing vehicle system uploads image data that has potentially captured an available parking space(s) to the available parking space information generation app, and then a more advanced algorithm is used there to confirm the available parking space by cross-validating it with other images uploaded from multiple vehicles and checking whether it is really located in an authorized parking area and conforms to the various regulations.

An alternative method is to let the sensing vehicle system detect the parking space availability and upload its result with estimated probability. By collecting such information from multiple vehicles, the available parking space information can be updated correctly. However, it should be noted that some computation resources need to be allocated for such detection in the vehicle system, while it is still necessary to increase computation resources to realize autonomous driving vehicles. In addition, such additional processes will further increase power consumption, although there is an expected problem of reduced vehicle driving range due to autonomous driving power consumption. Therefore, such a method is not further considered in this document.

After collecting the image data, it is contextualized against the map, or parking space location information, and then analyzed as quickly as possible to determine parking space availability status. In New York City, it is said that there are 3.4 to 4.4 million on-street parking spaces [NYC-Parking]. Thus, a great deal of image data collected needs to be linked to one of the possible parking space locations for analysis to update the available parking space information.

The assumptions on which we base the data volume estimate are summarized below:

- The service is operated in urban areas, such as Los Angeles, New York, London and Tokyo.
- Only peak hours are considered.
- 1 million user vehicles subscribed to the available parking space detection service are running simultaneously at peak hours.
- Each subscribed user vehicle requests the available parking space information every 2,000 seconds (33.3 min) on average.
- The available parking space information delivery app queries the available parking space information multiple times for each request (15 minutes, 5 minutes and 1 minute before arrival, and periodically in case it cannot locate the space for the requesting driver).
- 1 million sensing vehicles are running simultaneously at peak hours.
- The number of parking spaces is 5 million.
- To scan each of 5 million parking spaces every 25 seconds on average, 20% of 1 million sensing vehicles need to send an image with an average size of 100 KB every second at peak hour. This corresponds to the data rate of 160 Gbps 200,000 frames per second.

Data flows with data ingestion/processing velocities of the available parking space detection service in typical urban areas are summarized in the following table:

Table 1. Data flows, volumes and velocities of an available parking space detection service operated by a single OEM

Data Flow	Data Type	Data Transfer Frequency and Speed at Peak Hours
Foreground Process-relevant		
Available parking space information requests from the User Vehicle Systems to the Common Gateway, and the Common Gateway to the Available Parking Space Info Delivery App	Semi-structured	500 requests per second Each request may contain several KB of data representing the search areas and conditions.
Available parking space information requests from the Available Parking Space Info Delivery App to the Parking Space Info Store	Structured	2,500 queries per second
Available parking space information replied from the Parking Space Info Store to the Available Parking Space Info Delivery App	Semi-structured	2,500 replies per second Each reply may include several tens of records or nothing.
Available parking space information generation requests from the Available Parking Space Info Delivery App to the Available Parking Space Info Generation App	Structured	Very rarely at peak hours as the available parking space information is kept up to date as much as possible
Background Process-relevant		
Image data collection requests from the Available Parking Space Info Generation App to the Sensing Vehicle Systems through the Common Gateway	Structured	20,000 instructions per second
Image data uploaded from the Sensing Vehicle Systems to the Available Parking Space Info Generation App through the Common Gateway	Image	160 Gbps and 200,000 frames per second in total
Image data from the Available Parking Space Info Generation App to the Raw Data Store	Image	Same as above
Available parking space Information from the Available Parking Space Info Generation App to the Parking Space Info Store	Structured	200,000 updates (or inserts) per second at peak hour. Note: There are at least 5 million records in total, even if no historical data is managed. Each record may contain 100-200 B of data, assuming no history data is managed.
Control Plane-relevant		
Control signals between Control Application and other elements	Structured	Various signals will be exchanged, such as configuration change requests, resource utilization status (telemetry), heartbeat, etc.
Map Information replication data from the Reference Map (Master) Store to the Reference Map (Replica) Store through the Control Application	Semi-structured	Geospatial information that represents the road structure and parking space location information

3.2. Available Parking Space Detection Service That Relies On Other Similar Services

3.2.1. Scenario

In this scenario, we expand the previous use case of an available parking space detection service so that its underlying system is operated by multiple providers. Vehicles running on the roads are manufactured/sold by various OEMs and served by various connected service providers. Accordingly, if a certain degree of interoperability can be established between the data collected from the connected vehicles and the services built on the data, it will be possible to operate the services at a lower cost and with higher quality. As an example of such an interoperability scenario, we deal with an available parking space detection service that also relies on other available parking space detection services.

An available parking space detection service will rely on other available parking space detection services if it cannot confirm the required available parking space information by itself, which means that its own available parking space information store does not hold the up-to-date information, or if it cannot find enough vehicles around the targeted area to collect the image data to update the available parking space information. Other available parking space detection services may have the necessary information in such situations, and service quality can be improved by establishing interoperability to leverage such information.

The following figure shows such a high-level interaction scenario between two available parking space detection services.

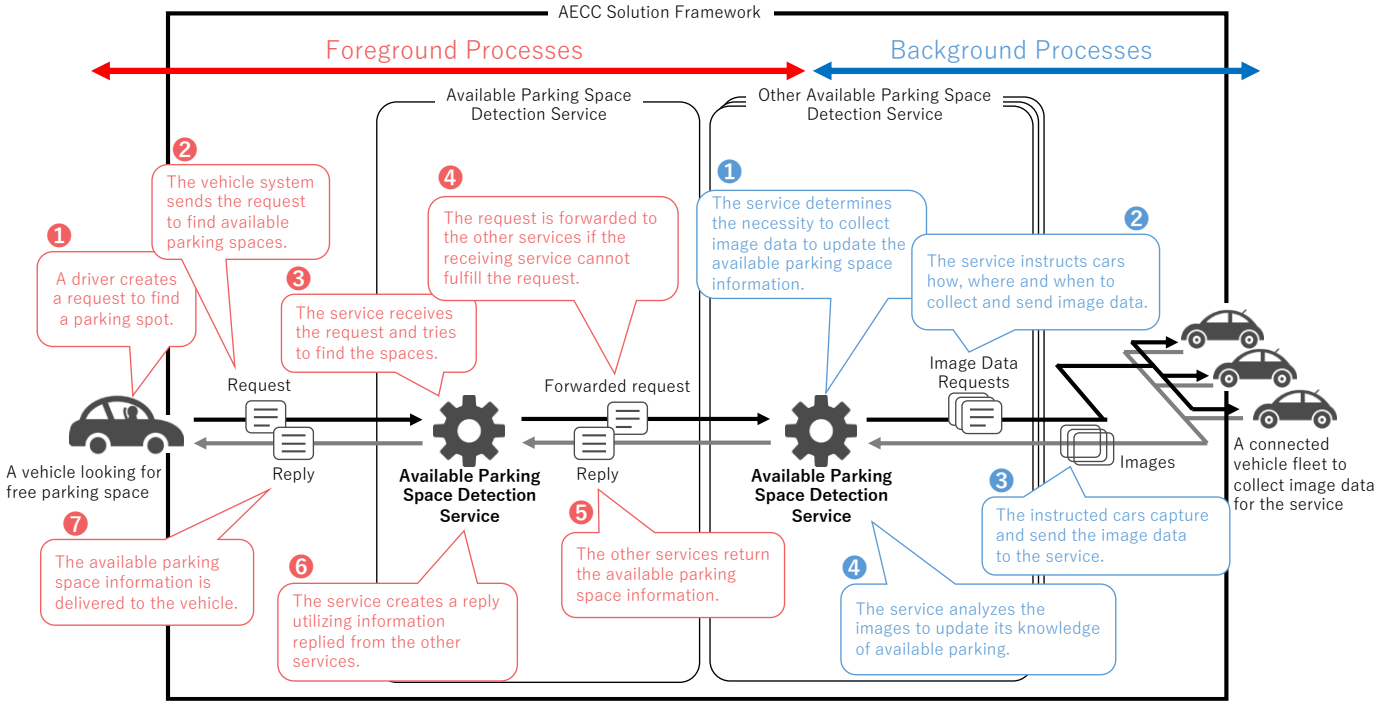

Figure 4. Service image of an available parking space detection service that relies on other services

A more detailed explanation of the system's behavior appears below:

Pre-conditions:

- Pre-contracts are made between available parking space detection services that interact together.
- API schemas are published in machine-readable formats in advance and updated if changes happen, to easily establish/maintain interoperability between available parking space detection services.

System Behavior:

Foreground Process

1. Drivers specify an area in which they want to find an available parking space, and the time they want to park their vehicle.
2. The vehicle system sends a request to provide the available parking space information, specifying the area and time to the available parking space detection service.
3. The available parking space detection service checks available parking spaces in the specified area at the right timings -- e.g., 30 minutes, 15 minutes, 5 minutes and 1 minute before the specified time -- or continuously if the driver has already arrived at the area but the service could not yet locate a space.
4. If the available parking space detection service cannot locate the available parking spaces by itself, it decides to interact with other available parking space detection services to find available parking spaces.
5. Each of the requested available parking space detection services checks whether it has fresh information on available parking spaces in the requested area, and if so, it returns the information to the requesting available parking space detection service.
6. The available parking space detection service checks and aggregates the available parking space information delivered by other available parking space detection services.
7. The available parking space detection service responds to the driver with the identified available parking space information.

Background Process

1. The available parking space detection service manages the freshness of the available parking space information and determines the necessity to collect image data to keep the information up to date.
2. The available parking space detection service selects the connected vehicles that will pass by target parking spaces and instructs them to send image data.
   Note: The parking space locations are pre-registered with the available parking space detection service.
3. The vehicle systems of the requested vehicles capture the images in front of the target parking space and send them to the available parking space detection service.
4. The available parking space detection service receives the image data sent from vehicles and analyzes it, then updates the status of the parking spaces of interest.

3.2.2. Deployment Model

When it comes to the deployment model, interactions among independent services must be considered. Again, we assume that the interaction goes through the common gateway to hide detailed configuration information and protect privacy. The APIs to be called and API schemas are also exposed at the common gateway.

Figure 5 shows an example of such a deployment model of an available parking space detection service that relies on other available parking space detection services.

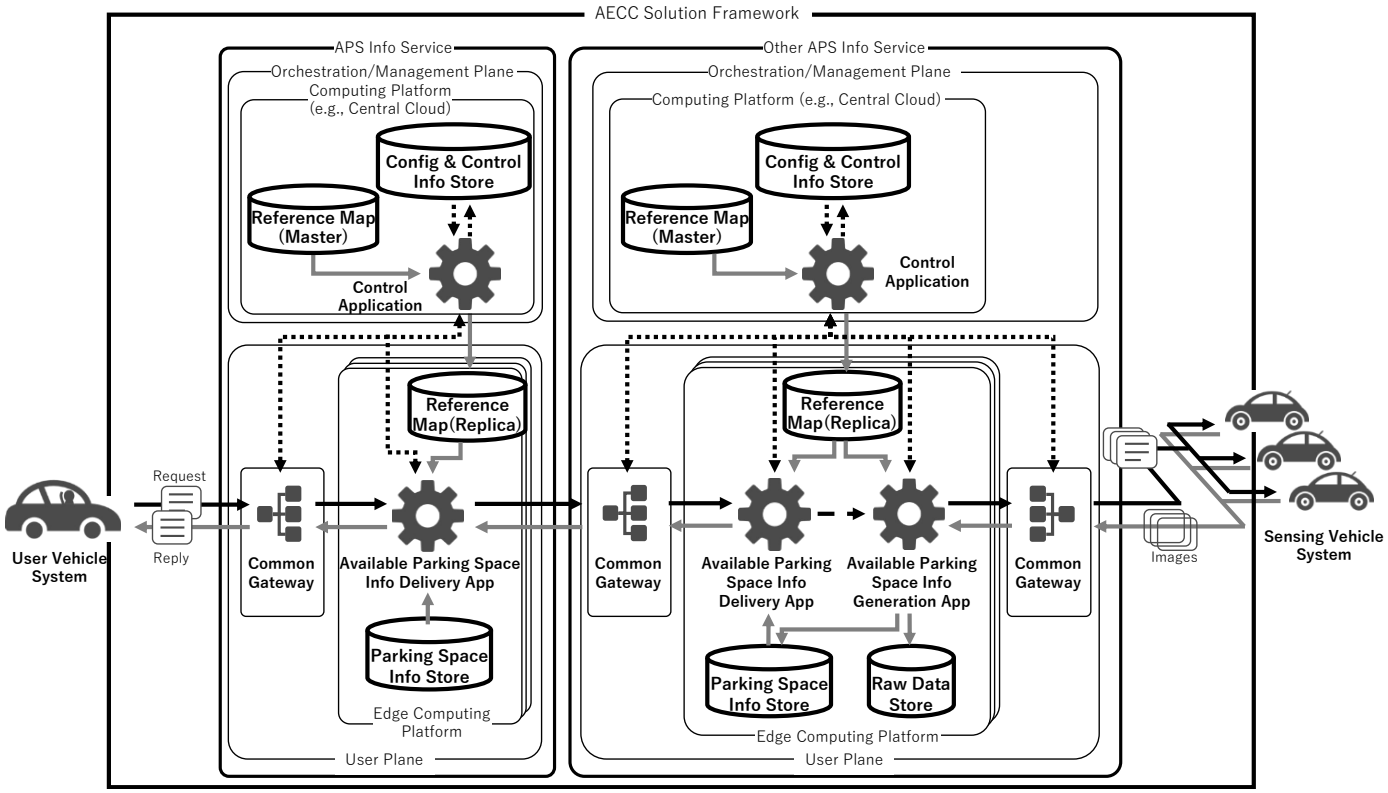

Figure 5. Deployment model example of an available parking space detection service that relies on other services

The system elements considered are basically the same as in the first use case, but a little more extended for the common gateway, which is described below:

- **User Vehicle System**
- **Sensing Vehicle System**
- **Common Gateway**
  Note: In addition to the basic roles that are described in the first use case, here it also exposes APIs and API schemas for external parties.
- **Available Parking Space Information Delivery App**
- **Available Parking Space Information Generation app**
- **Parking Space Information Store**
- **Raw Data Store**
- **Reference Map (Master)**
- **Reference Map (Replica)**
- **Control Application**
- **Configuration and Control Information Store**

3.2.3. Overview of Data Flows

To estimate data flow volumes and velocities, we assume that:

- An average of 20% of available parking space information requests are forwarded to other available parking space detection services.
- When requested, the available parking space detection services requested by another service only check their own available parking space information store to respond but do not collect the image data from their sensing vehicle fleets upon request.
- Request-Reply is a one-time operation; that is, it does not send the replies multiple times in an asynchronous manner.

Most of the data flow with volumes and velocities within each available parking space detection service is basically the same as described in the first use case. However, some additional data flow and data amounts need to be considered due to additional requests coming from external available parking space detection services.

Data flows with data ingestion/processing velocities of an available parking space detection service that relies on other available parking space detection services for the typical urban area are summarized in Table 2 below:

Table 2. Data flows, volumes and velocities of an available parking space detection service that relies on other available parking space detection services

Data Flow	Data Type	Data Transfer Frequency and Speed at Peak Hours
Foreground Process-relevant		
Available parking space information requests from the User Vehicle Systems to the Common Gateway, and from the Common Gateway to the Available Parking Space Info Delivery App	Semi-structured	500 requests per second Each request may contain several KB of data representing the search areas and conditions.
Available parking space information requests from the Available Parking Space Info Delivery App to the Parking Space Info Store	Structured	2,500 queries per second
Available parking space information replies from the Parking Space Info Store to the Available Parking Space Info Delivery App	Semi-structured	2,500 replies per second Each reply may include several tens of records or nothing.
Available parking space information requests from the Available Parking Space Info Delivery App serving the User Vehicle Systems to the other Available Parking Space Info Delivery Apps through the Common Gateway	Semi-structured	100 requests per second x number of external available parking space detection services
Available parking space information replies from the other Available Parking Space Info Delivery Apps to the Available Parking Space Info Delivery App serving the Use Vehicle Systems through the common Gateway	Semi-structured	100 replies per second Each reply may include several tens of records or nothing.
Background Process-relevant		
Image data collection requests from the Available Parking Space Info Generation App to the Sensing Vehicle Systems through the Common Gateway	Structured	20,000 instructions per second

Data Flow	Data Type	Data Transfer Frequency and Speed at Peak Hours
Image data uploaded from the Sensing Vehicle Systems to the Available Parking Space Info Generation App through the Common Gateway	Image	160 Gbps and 200,000 frames per second in total
Image data sent from the Available Parking Space Info Generation App to the Raw Data Store	Image	Same as above
Available parking space Information from the Available Parking Space Info Generation App to the Parking Space Info store	Structured	200,000 updates (or inserts) per second at peak hour. Note: There are at least 5 million records in total, even if no historical data is managed. Each record may contain 100-200 B of data assuming no history data is managed.
Control Plane-relevant		
Control signals between the Control Application and other elements	Structured	Various signals will be exchanged, such as configuration change requests, resource utilization status (telemetry), heartbeat, etc.
Map Information replication data from the Reference Map (Master) Store to the Reference Map (Replica) Store through the Control Application	Semi-structured	Geospatial information that represents the road structure and parking space location information

3.3. Available Parking Space Data Reuse Services

3.3.1. Scenario

The last example application concerning available parking space detection services is a data reuse scenario, because collecting data from connected vehicles would be very costly and using it only for a single service would be very inefficient. Therefore, the data collected, or the insight information extracted from the data should be reused as much as possible.

Such possible data reuse scenarios include:

- Record the parking space usage history and inform general users about the difficulties of parking per area by the time of day, or provide it to companies who develop the parking spaces to assist them to estimate profitability.
- Reuse the mosaicked raw data to provide a street viewing service. As mosaicked raw data is collected from many sensing vehicles frequently, compared to existing similar services, this service allows us to see the changing street conditions by time of day.
- Reuse the mosaicked raw data to run visual analysis to estimate traffic and/or pedestrian congestion, find prices such as congestion-based urban area entry fees, and so on. Since image data may capture various objects, there will be various ways of reusing it. When doing so, of course, privacy must be protected.

The following figure shows such data reuse scenarios on top of an available parking space detection service.

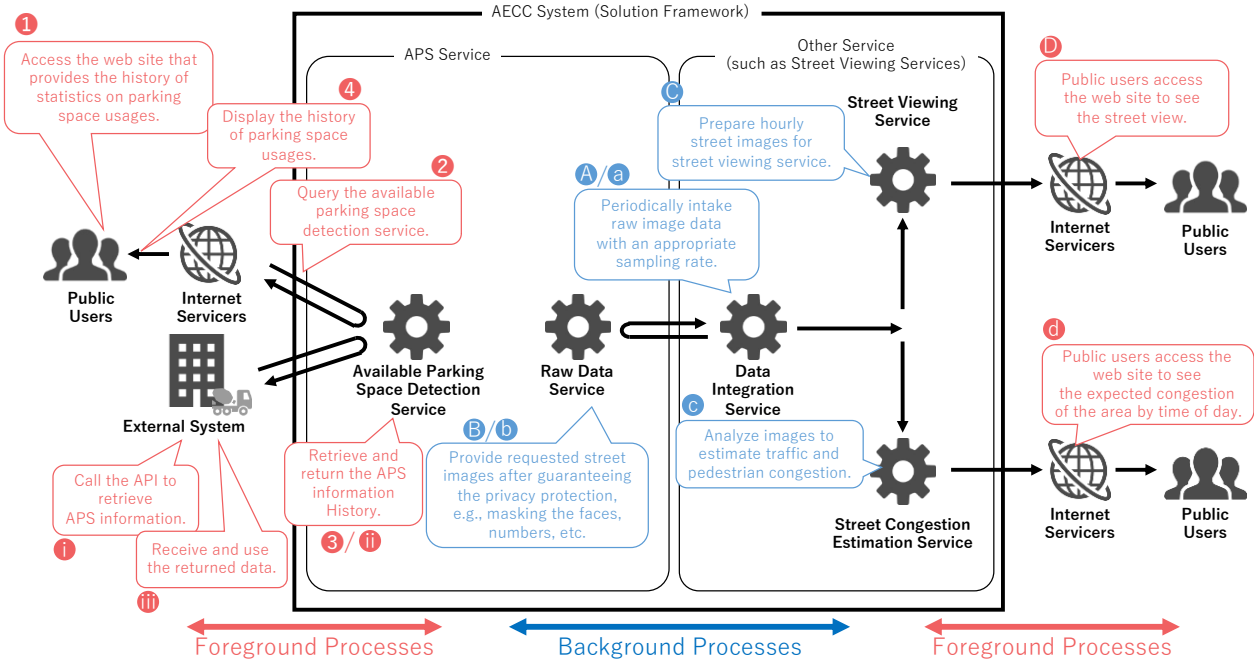

Figure 6. Service image of available parking space data reuse services

A more detailed explanation of the system behavior is as follows:

Pre-conditions:

- Pre-contracts/agreements are made between an available parking space detection service and its users.
- API schemas are published in machine-readable formats in advance and updated if changes are made, to easily build and maintain websites to deliver the value-added information to public users and/or to let a system of the contracted company consume the data easily.

System Behavior:

<u>Website to display parking space usage history</u>

1. Public users access a website that provides parking space usage statistics.
2. The website queries the available parking space detection service.
3. The available parking space detection service retrieves the parking space usage statistics and returns them to the website.
4. The website displays the retrieved parking space usage statistics.
   Note: the parking space usage statistics may be cached for a while on the website.

<u>APIs to retrieve parking space usage data</u>

1. A system of an external party, such as a construction company or local administrative organization, calls the APIs of an available parking space detection service to retrieve parking space usage statistics.
2. The available parking space detection service retrieves the parking space usage statistics and sends them to the external system through the API.
   Note: Besides usage rate, other necessary additional information such as parking time may be part of the response. These are design choices made by the service provider.

3. The external system will receive the data and use it.

Street viewing service

1. The street viewing service requests a raw data service exposed by the available parking space detection service provider to provide the mosaicked raw data periodically with an appropriate sampling rate.
   Note: If only blurry images are retrieved, then it retrieves other samples.
2. The raw data service retrieves the mosaicked raw data, checks whether personal information has been completely removed and responds to it.
3. The street viewing service prepares hourly street images.
4. Public users access the website to see the street images by the time of day.

Street congestion estimation service

1. The street congestion estimation service requests a raw data service exposed by the available parking space detection service provider to provide the mosaicked raw data periodically with an appropriate sampling rate.
   Note: If blurry images are retrieved, then it retrieves other samples.
2. The raw data service retrieves the mosaicked raw data, checks whether personal information has been completely removed and responds to it.
3. The street congestion estimation service analyzes the retrieved mosaicked raw data to estimate hourly traffic and pedestrian congestion levels.
4. Public users access the website to see the congestion levels of the street by the time of day.

3.3.2. Deployment Model

When such services are implemented, additional APIs, as well as service applications, will of course be added. However, even in such cases, relevant data and the common gateway to host APIs can be considered common elements. By standardizing the system, the system cost can be lowered.

Figure 7 shows such a deployment model of available parking space data reuse services.

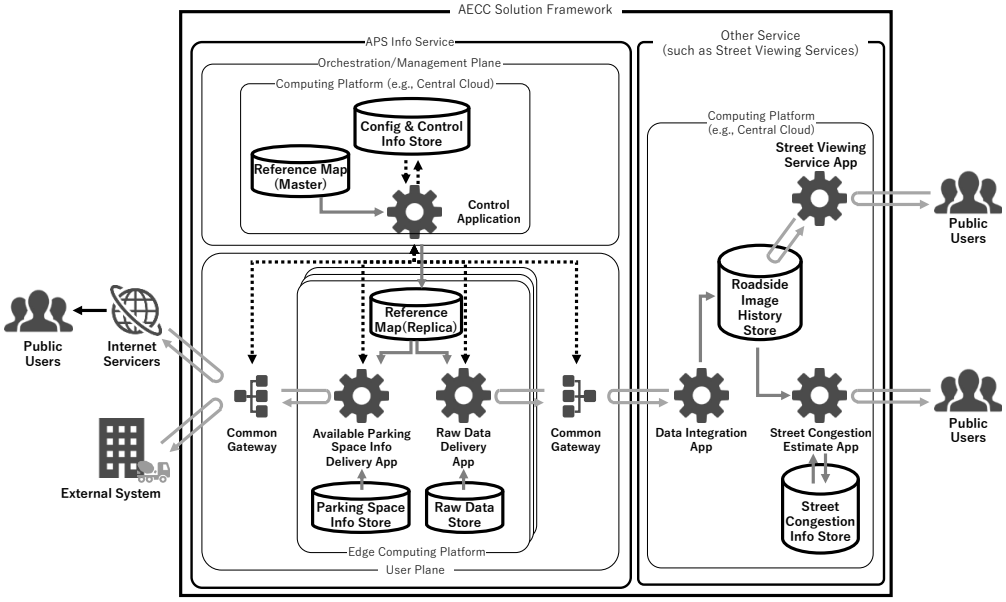

Figure 7. Deployment model example of available parking space data reuse services

The system elements considered are listed below:

- **Public Users:** Generic users who enjoy the value-added available parking space detection services from the available parking space detection service provider or its partners
- **External System:** A system that retrieves data from the available parking space detection service through exposed APIs
- **Common Gateway**
- **Available Parking Space Information Delivery App**
- **Raw Data Delivery App:** An application that provides the mosaicked raw data to external applications through the common gateway
- **Available Parking Space Information Store**
- **Raw Data Store**
- **Reference Map (Master)**
- **Reference Map (Replica)**
- **Control Application**
- **Configuration and Control Information Store**
- **Data Integration App:** An application that periodically intakes the mosaicked raw data from the raw data delivery app through the common gateway
- **Roadside Image History Store:** A store that holds the selected mosaicked images that are required in order to provide the street viewing service
- **Street Viewing Service App:** A service application that provides hourly street images to public users
- **Street Congestion Estimate App:** An application that estimates the traffic and pedestrian congestion levels for each road segment in an urban area based on image analysis
- **Street Congestion Information Store:** A store that holds the traffic and pedestrian congestion information

Note: Descriptive text is given only for new elements.

3.3.3. Overview of Data Flows

To estimate data flow volumes and velocities, we assume the following:

- There are 100,000 people per hour who check the parking space usage statistics at peak times.
- A cache hit rate at the website is assumed to be 95%. Inquiries are made to the backend system once every 20 times.
- There are 50,000 streets in each urban area, such as Los Angeles, New York and London.
  Note: As the image data is collected at a speed of 200,000 frames per second, the status of each road segment is updated every 4 seconds on average at peak hours.

Expected data flows with data ingestion/processing velocities for available parking space data reuse services for an urban area are summarized in Table 3.

Table 3. Data flows and data velocities of available parking space data reuse services

Data Flow	Data Type	Data Transfer Frequency and Speed at Peak Hours
Website to display parking space usage history		
Parking space usage statistics requests from public users to the website	Semi-Structured	28 requests per second Note: Each request may contain several KB of data representing the search areas and conditions.
Parking space usage statistics requests from the website to the Parking Space Info Store through the Common Gateway and the Availability Parking Space Info Delivery App	Semi-Structured	1.4 requests per second Note: Each request may contain several KB of data representing the search areas and conditions.
Parking space usage statistics replies from the Parking Space Info Store to the website through the Availability Parking Space Info Delivery App and the Common Gateway	Structured	1.4 responses per second Note: Each response may contain as much as 1,000 pieces of parking space information, which consists of hourly usage rates for a designated week.
APIs to retrieve the parking space usage data		
Parking space usage statistics requests from the external systems to the Availability Parking Space Info Delivery App through the Common Gateway	Semi-Structured	Ad hoc Note: Each request may contain several KB of data representing the search areas and conditions.
Parking space usage statistics replies from the Availability Parking Space Info Delivery to the External Systems through the Common Gateway	Semi-Structured	Ad hoc Note: Each response may contain as much as 1 million pieces of parking space information, which consists of hourly usage rates in designated months.
Street viewing service/street congestion estimation service		
Raw image data synchronization from the Raw Data Delivery App to the Roadside Image History Store through the Data Integration App	Unstructured	Every minute Note: In each batch job, 10 images with an average size of 100 KB are retrieved per road segment. This means 1 TB of data is retrieved in total every minute.
Raw image data synchronization from the Roadside Image History Store to the Street Viewing Service App	Unstructured	Hourly Note: In each batch job, 10 images with an average size of 100 KB are retrieved per road segment. This means 1 TB of data is retrieved in total hourly.
Raw image data synchronization from the Roadside Image History Store to the Street Congestion Estimation App	Unstructured	Every minute Note: In each batch job, 10 images with an average size of 100 KB are retrieved per road segment. This means 1 TB of data is retrieved in total every minute.
Congestion information from the Street Congestion Estimation App to the Street Congestion Info Store	Structured	Every minute Traffic and pedestrian congestion levels at each of 1 million road segments are represented numerically and stored.

4. AECC Data Management System

As described in the example scenarios of Chapter 3, the AECC system needs to collect, process and use a large amount of data rapidly and securely to provide a series of mobility services. As such a workload and volume of data are very large, the AECC assumes that there is geographically distributed computing architecture as well as locally distributed computing architecture to manage and process the data.

The reasons why we assume such a hierarchically distributed architecture are as follows:

1. **The traffic flow is too large to cover with just one data center**
   As described in the previous chapter, an available parking space detection service may generate 0.5 Tbps just for one metropolitan area. As there will be more mobility services and many metropolitan areas in many countries, the total traffic volume coming from the vehicles will be tens of Tbps, which may not be covered by one data center today.
2. **The data transfer cost is too high**
   If a large amount of data must be transferred over a long distance, it generates a huge workload for network devices, consumes a lot of energy there and increases communication costs. This burden and cost can be reduced if the data can be processed near its source in a regional data center.
3. **Regional data processing for mobility services**
   Mobility services are categorized into vehicle-specific services, driver/user-specific services and regional services. It is very clear that regional services best fit the regional data center, but other services are also well managed by regional data centers, because the speed of both vehicles and people is finite, and they stay in the same area for a certain period.
4. **Too much data to manage in just one instance**
   Even when a metropolitan area is managed by, for example, five regional data centers, the total data flow rate of 0.5 Tbps is enormous, so that each data center has to accommodate 0.1 Tbps data flow and accompanying data processing. Therefore, multiple instances will be required to manage and process the data.

However, such a hierarchically distributed architecture creates other challenges, such as how to let running vehicles connect and re-connect to the right edge computing platform, how to manage and utilize geo-distributed and locally distributed data, and how to manage the topology and configuration of the entire system. This section describes the AECC solution framework to address these challenges covering system elements in the geo-distributed architecture.

4.1. Structure of the AECC Data Management System

Figure 8 shows a generalized view of the AECC data management system, which is a generalized version of the deployment models written in Chapter 3.

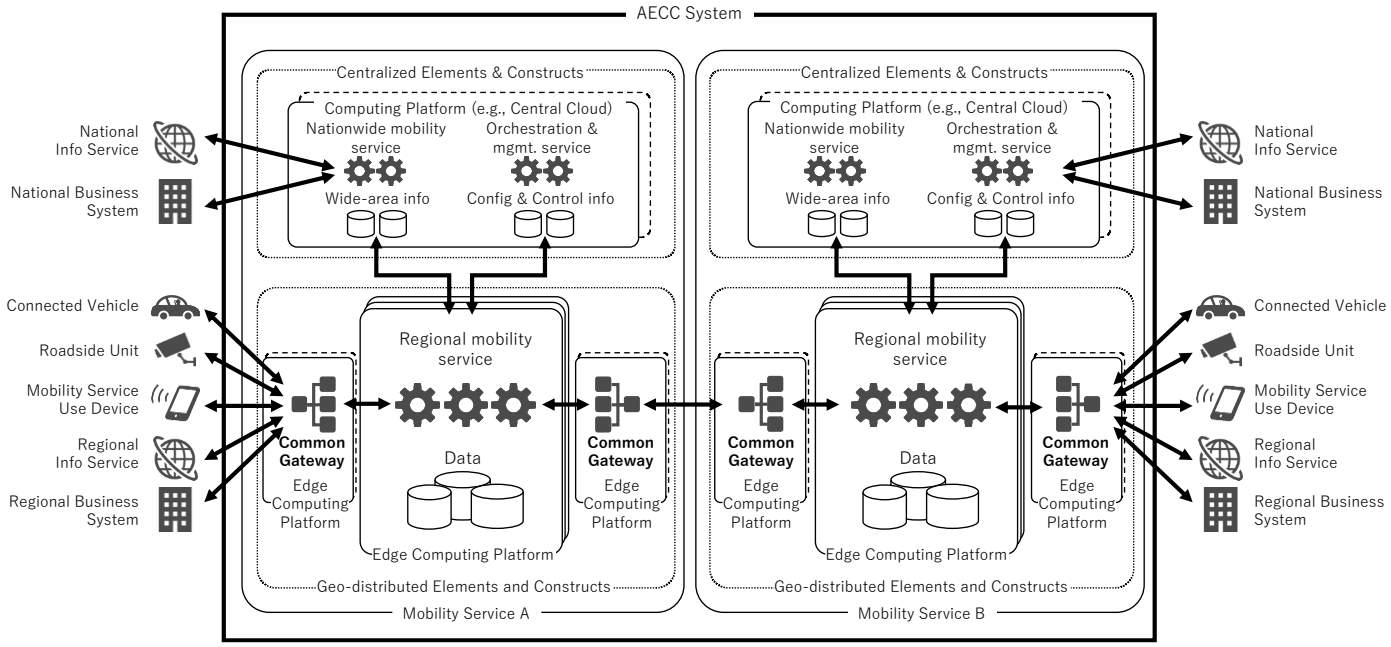

Figure 8. AECC data management system

One of the objectives of the AECC is to come up with guidelines and recommendations to help mobility service providers develop and deploy the required data management system, and also help edge cloud providers and network service providers prepare the infrastructure to fulfill such mobility service provider requirements.

In this chapter, we describe the elements and constructs of the AECC data management system first and then discuss what should be considered for each of the elements and/or constructs as well as the interactions between them.

4.2. AECC Data Management System Elements and Constructs

4.2.1. Clients

As described in Figure 8, we assume the following as a possible AECC data management system client:

Connected Vehicle

The connected vehicle is configured to connect with mobility services running on the AECC system. It receives information from mobility services to enhance passenger experiences. It also collects and sends data when requested to mobility services seeking to know the vehicle status, traffic conditions, etc.

It is assumed that connected vehicles are connected to the common gateway, which runs in a geo-distributed manner to intermediate traffic to the nearest mobility service instance at short latency. When geographically dispersed data is required, for route optimization, etc., the connected mobility service instance collects data from other instances, processes the data and responds to the vehicle with a data processing result through the common gateway. Since the speed of a vehicle is finite and it needs information only near where it drives, it is not expected that the vehicle will access centralized information in the central cloud.

Roadside Units

A roadside unit is a computing platform for mobility service functions that additionally supports sensors and actuators that interact with traffic on the nearby roadway. For example, it can capture traffic and pedestrian conditions and send them to another remote computing platform for further data processing. The mobility service functions may implement a stand-alone mobility service or may be part of a distributed implementation.

Mobility Service User Devices

A mobility service user device, such as a smartphone, is a user device that interacts with mobility services. The user can check on the condition of a vehicle and receive services such as ride-sharing vehicle arrangements.

Just like the connected vehicle, it needs only local information; it receives the service from the nearby mobility instance through the gateway, but not from the mobility service instance in the central cloud.

Information Service

The information service is an external service that re-uses the data collected by mobility services and provides value-added information, such as the congestion level of a city, to public users. The information services can be divided into the following two categories according to their connection destinations: 1) the regional information service, which connects to one or several regional mobility services, and 2) the national information service, which connects to the nationwide mobility service to utilize nationwide data, which might be stored in the regional computing platforms in a distributed manner.

Regional Business System

A business system is an external system that consumes the mobility service functionality or the data stored there through API-based interactions. Business systems can be divided into the following two categories according to their connection destinations: 1) the regional business system, which connects to one or several regional mobility services, and 2) the national business system, which connects to the central computing platform or the central cloud, to run and coordinate nationwide applications, such as geo-distributed machine learning algorithms.

Other AECC Systems

As shown in the example in Chapter 3, multiple AECC systems may work together to provide services. In such a case, one AECC system can be considered a client of another AECC system.

4.2.2. Server-side Data Management Elements and Constructs

An AECC System is assumed to have the following, as depicted in Figure 8.

AECC Data Management System

The AECC data management system provides a variety of data management services required for mobility services running on top of the distributed computing platforms. It consists of various elements and constructs, such as hardware, server racks, executables, modules, applications, data, etc. To let system elements and constructs be linked and work together, the AECC data management system has a certain mechanism and rules, which will be explained later in this paper.

Note: The system elements here mean the smallest unit, such as an executable, packet of data or instance, which is determined from the perspective of resource management in building a mobility service. The system constructs mean an assembly, such as modules or subsystems, that is composed of multiple system elements.

Computing Platform

A computing platform is an environment in which various mobility services are executed, and data collected from vehicles, roadside units and other user devices is managed and processed. Examples of such computing platforms include, but are not limited to, central clouds and edge clouds.

Depending on their position in the system topology, the computing platforms can be divided into the following two categories: 1) the regional computing platform, which covers a particular predefined region, and 2) the central computing platform, which covers the entire country or the world.

Note that for high-availability, disaster recovery purposes and scale-out purposes, the system elements and constructs may be replicated across multiple computing platforms, regardless of whether they are regional or central.

Mobility Services

Mobility services are a logical concept that represents a type of service relevant to mobility. Some examples are: 1) status management of a personally owned vehicle, 2) remote driving assistance, such as obstacle-on-road alerts, and 3) coordination of ride-sharing services, etc., running on top of the AECC system.

To provide such services to users, a complex system that consists of various system constructs, which in turn, consist of various system elements, such as applications (or executables), data and hardware, can be used.

Mobility services are divided into the following two categories according to their regional coverage: 1) ones that provide services that are required to meet various latency or locality-sensitive service level requirements, and 2) ones that provide location-independent services.

Common Gateway

A common gateway is a logical component that will be used when it is required to hide the implementation details of mobility services from in-vehicle systems and other clients, and/or to hide personal information included in the collected data from mobility services. For that purpose, the common gateway is placed between mobility service clients and mobility services, and it redirects/intermediates communications between them.

For connected vehicles, the existence of a large number of geo-distributed environments means that it needs to think about which one to connect to. The common gateway is designed to take over such data flow controls with low latency. As such data flow controls are common across multiple mobility services, the same common gateway can be used to hide the implementation details of mobility services of concern from mobility service clients. This also enables flexible changes to the mobility service configuration, as it is not required to deliver configuration change information to a massive number of vehicle systems.

Orchestration and Management Apps

Orchestration and management applications provide system capabilities that are designed to let distributed systems effectively and efficiently work together. These rely on configuration and control information as a knowledge base of the hierarchically distributed system.

Configuration and Control Information

Configuration and control information is data that represents a structure of the AECC system, such as placement of applications and data, interaction and interdependency between them, and hardware components used to host them, as well as rules and procedures to let multiple geo-distributed elements work together.

Without having configuration and control information that is always kept up to date, the orchestration and management applications of the AECC system cannot work properly.

It is assumed that such configuration and control information, together with orchestration and management applications, is managed in a central computing platform or a central cloud as described in Figure 8. Of course, theoretically, such information can be managed in a distributed manner, but then, a consensus algorithm or a chatty two-phase commit control needs to be implemented to manage data, which will be too costly.

4.3. Data and Data Flow in the AECC System

Data and data flow are very important when considering an AECC system. This is because the following points must be addressed:

- Various data types are required for mobility services, including structured data (such as table data, protocol buffer, etc.), semi-structured data (such as JSON, XML, etc.), and unstructured data (image, video, etc.).
- The amount of data flow is very large, as image, video and LiDAR sensor data from millions of vehicles is collected.
- Data flow is very complex, as the AECC system clients and the mobility service instances are geo-distributed.
- Data privacy and secrecy are the most critical aspects for building mobility services, misleading information disturbs traffic, and information leakages result in a large penalty to the mobility service provider.

Table 4 shows assumptions about the data flows that may occur in the future, assuming there are 1) 1 million connected vehicles, 20% of which capture and transmit 100 KB of image data every second on average, 2) 100,000 roadside units capturing and transmitting 100 KB of image data every second, 3) 1 million mobility service user devices, 4) 1,000 regional services and/or business systems, 5) 100 national services and/or business systems, and 6) a group of mobility services operated in the target region/nation.

Table 4. Overview of data flow in the AECC system

From	To	Data Type	Data Size	# of Vehicles	% of Vehicles Transferring Data Simultaneously	Data Transfer Frequency Per Connection	Bit Rate (in total)
Connected Vehicle	Regional Mobility Service	Image, video, LiDAR	> 100 KB	1 million	20%	> Every second	> 160 Gbps
Connected Vehicle	Regional Mobility Service	Cruising data (e.g., location, speed, etc.)	> 100 B	1 million	100%	Less frequently than every second	> 0.8 Gbps
Regional Mobility Service	Connected Vehicle	Control signals	> 1 KB	1 million	100%	> Every 50 seconds	> 160 Mbps
Connected Vehicle	Regional Mobility Service	Service request	> 1 KB	1 million	100%	> Every 2,000 seconds	> 4 Mbps

From	To	Data Type	Data Size	# of Vehicles	% of Vehicles Transferring Data Simultaneously	Data Transfer Frequency Per Connection	Bit Rate (in total)
Regional Mobility Service	Connected Vehicle	Service reply	> 1 KB	1 million	100%	> Every 400 seconds	> 20 Mbps
Roadside Unit	Regional Mobility Service	Image, video, LiDAR	> 100 KB	0.1 million	100%	> Every second	> 80 Gbps
Mobility Service User Device	Regional Mobility Service	Service request	> 1 KB	1 million	> 0.1%	> Every 5 seconds	> 1.6 Mbps
Regional Mobility Service	Mobility Service User Device	Service reply	> 1 KB	1 million	> 0.1%	> Every second	> 8 Mbps
Regional Info Service/Biz Sys	Regional Mobility Service	Service request	> 1 KB	1,000	100%	> Every 5 seconds	> 1.6 Mbps
Regional Mobility Service	Regional Info Service/Biz Sys	Service reply	> 1 KB	1,000	100%	> Every 5 seconds	> 1.6 Mbps
National Info Service Biz Sys	Wide-area Mobility Service	Service request	> 1 KB	100	100%	> Every 5 seconds	> 160 Kbps
Wide-area Mobility Service	National Info Service/Biz Sys	Service reply	> 1 KB	100	100%	> Every 5 seconds	> 160 Kbps
Regional Mobility Service	Another Regional Mobility Service	Service request	> 1 KB	1	100%	> 100 times per second	> 800 Kbps
Another Regional Mobility Service	Regional Mobility Service	Service Reply	> 1 KB	1	100%	> 100 times per second	> 800 Kbps

The above table is merely a set of assumptions. The frequency of data flow generation and its volume will vary greatly depending on the configuration of mobility services. In addition, various techniques will be applied to reduce the amount of data collected. However, the volume of data collection is still far greater than the volume of data distribution. This means that in order to build a profitable mobility service, it is required to collect, process and manage data at a low cost and to provide value from the same data repeatedly.

4.4. Data Management Patterns in the AECC System

When building various mobility services, we must consider various patterns of data management to determine the right architecture and technologies to properly support them. Major patterns of such abstracted data management are described below:

4.4.1. Data Shuffling (Contextualization)

As one million vehicles may be connected to one regional computing platform and hundreds of thousands of road segments may be managed together there, data shuffling or data contextualization is an important data processing step to build any mobility service. Because the mobility service is to provide vehicle-specific or location-specific insights and orchestrations, the data collected from millions of vehicles needs to be filtered efficiently by vehicle or location. The data may also be consolidated to increase statistical accuracy.

Figure 9 shows an image of such a data shuffling or contextualization process.

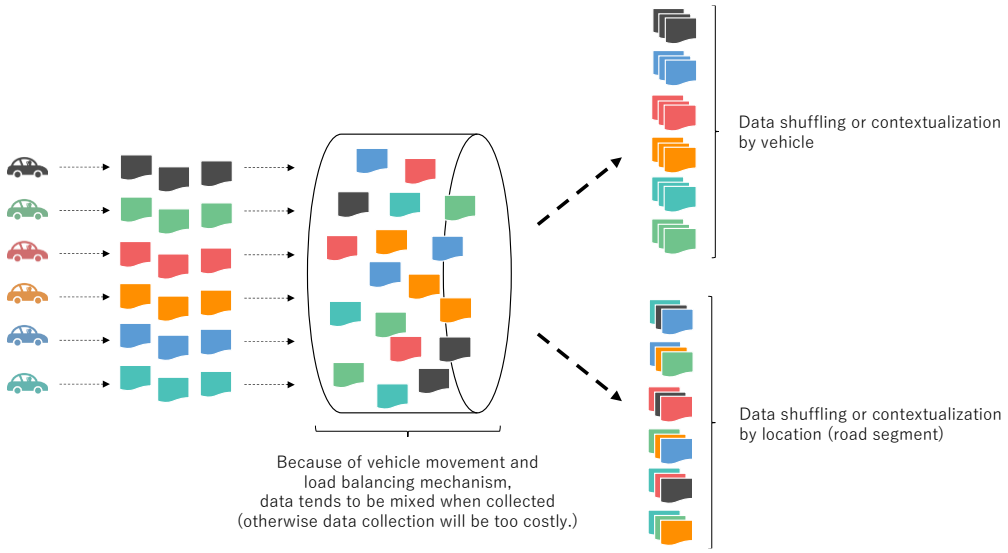

Figure 9. Data shuffling or data contextualization

4.4.2. Data Sampling (First N Values Extraction)

When extracting insights from data, such as local traffic congestion, it would be enough to analyze only a certain amount of data to get statistically sufficient accuracy. Therefore, data sampling or first N values extraction will be used in many applications before analyzing data.

Figure 10 shows an image of such a data sampling or first N value extraction process.

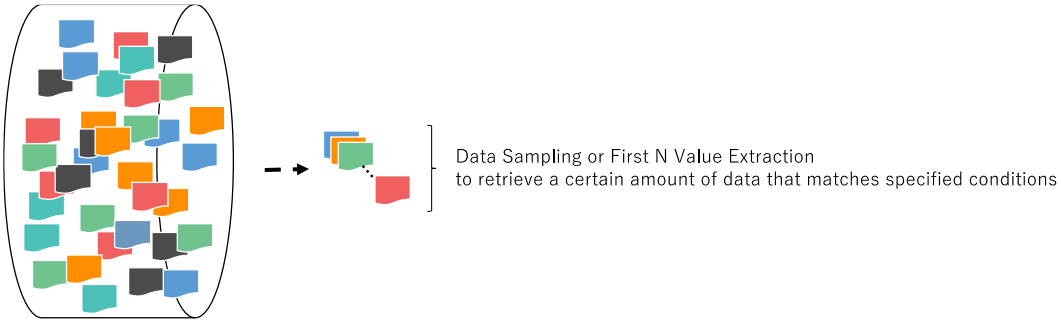

Figure 10. Data sampling or first N value extraction

4.4.3. Feature Extraction

To streamline data analysis and utilization, features may first be extracted from the collected raw data. Examples include road lines and surface marks identified in image analysis, and average velocity and acceleration distribution calculated from the cruising data. By extracting features, the amount of data passed to subsequent processes can be reduced.

Figure 11 shows an image of such a feature extraction process.

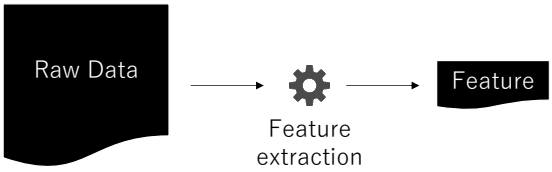

Figure 11. Feature extraction

4.4.4. Data Compression

When data is transferred over a long-distance network, or stored in storage for a long period, it is reasonable to compress the data, if the cost of compression and decompression is lower than the cost for data transfer or data storage. Various data compression methods are used depending on the data structure, such as zip compression, columnar conversion, delta encoding, etc.

Figure 12 shows an image of such a data compression process.

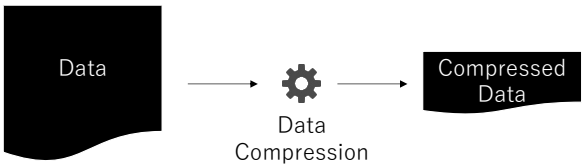

Figure 12. Data compression

4.4.5. Generating Insights from Data

Once enough samples are collected and features from them are extracted, the data is ready to be analyzed to generate insights. For example, the availability of parking spaces and the status of road construction will be confirmed. It should be noted that in order to acquire reliable results, multiple feature data within a certain time range may be considered together to evaluate statistical certainty, especially when the machine learning inference model has a not-small error rate or the analysis is to deal with situations/events that change rapidly over time.

Figure 13 shows an image of such an insight extraction process.

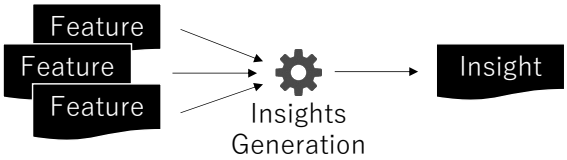

Figure 13. Insight extraction from feature data

4.4.6. Insight Data Management

Generated insight data must also be managed because this will be used repeatedly by various vehicles and external applications. The AECC HD Map White Paper [AECC HD Map]describes an example of such a data management mechanism for location-based insights. Of course, other insights can be tied to the vehicle and driver. All of them must be properly and securely managed.

Figure 14 shows an image of such an insight data management system.

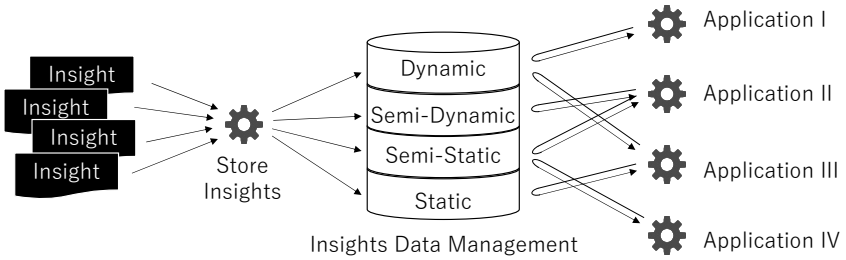

Figure 14. Insight data management

4.4.7. Push Notifications to Running Vehicles

Some of the generated insight data must be pushed to the running vehicle. For example, if the mobility service detected a vehicle entering a poorly visible intersection and disregarding a traffic light, push notification of this to other vehicles that might be hit by such a vehicle would help avoid an accident. In such cases, the information must be pushed to vehicles that are approaching the intersection of concern and are subscribed to the push notification service. To enable rapid delivery of such push notifications, the required information is collected from subscribed running vehicles in advance.

Figure 15 shows an image of such a push notification mechanism.

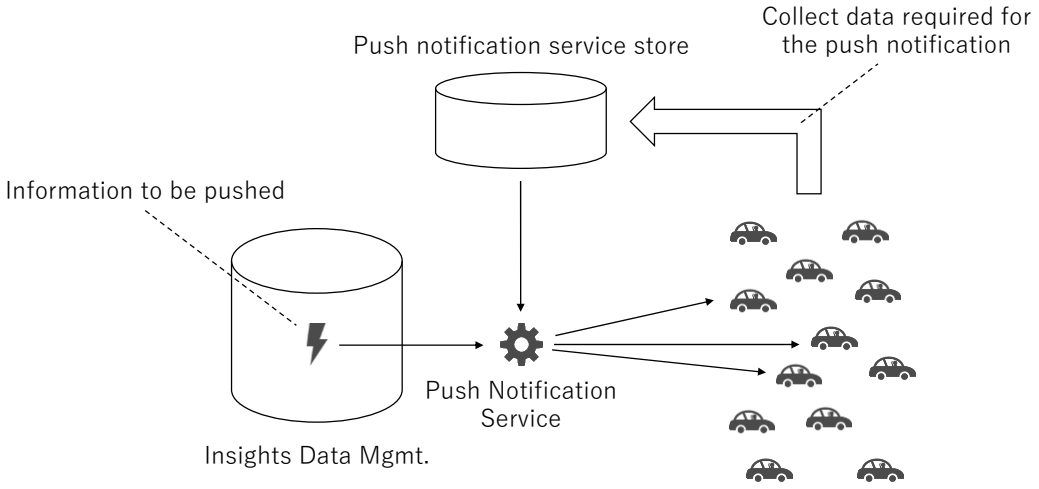

Figure 15. Push notifications to running vehicles

4.4.8. Data Catalog

Data may be shared and used by various mobility service applications. To streamline data access and retrieval, metadata and indexes are created and managed in the data catalog. Such a data catalog should take into account the distributed environment envisioned by the AECC. As there can be many distributed computing platforms that are geographically far apart, regional data catalogs can be created and configured to work together to support global data utilization.

Figure 16 shows an image of such data cataloging and indexing processes for subsequent data accesses.

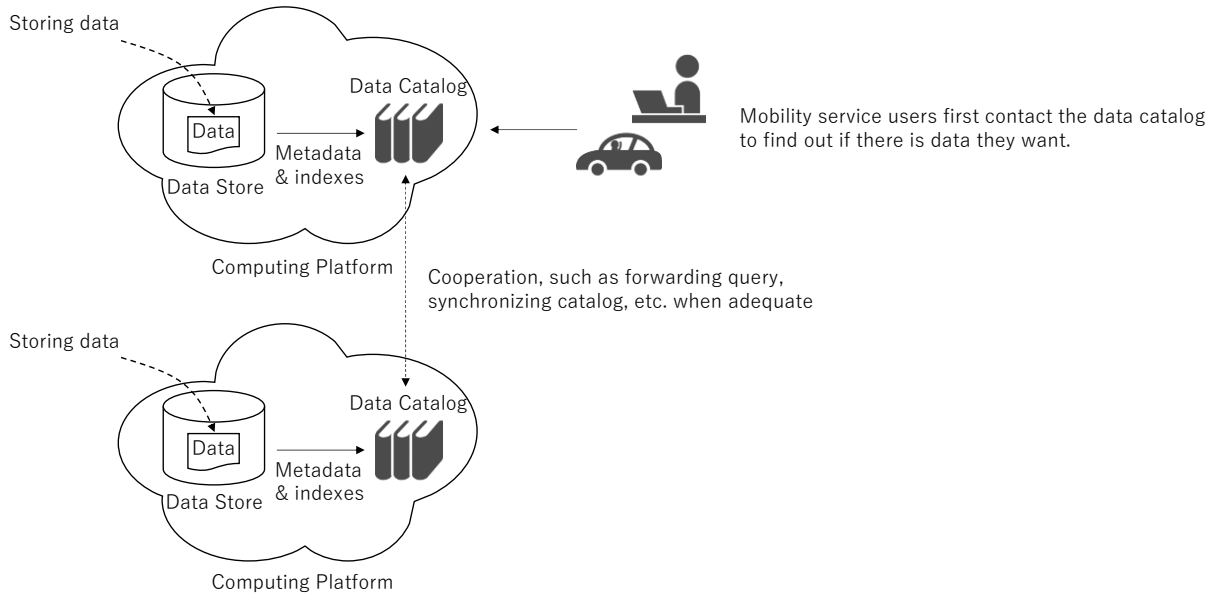

Figure 16. Data cataloging and indexing

4.4.9. Secure Data Sharing

Data can also be used by external mobility service providers and other organizations, so that value can be created from the same data repeatedly. However, the data collected from vehicles can be very sensitive. Therefore, it is almost impossible to exchange raw data directly. Instead, a safety component must be put in place. That component ensures that sensitive and/or private information is not included before providing the data to external parties. The external party accesses the component via APIs and obtains access to the secured data.

Figure 17 shows an image of such a secure data-sharing mechanism.

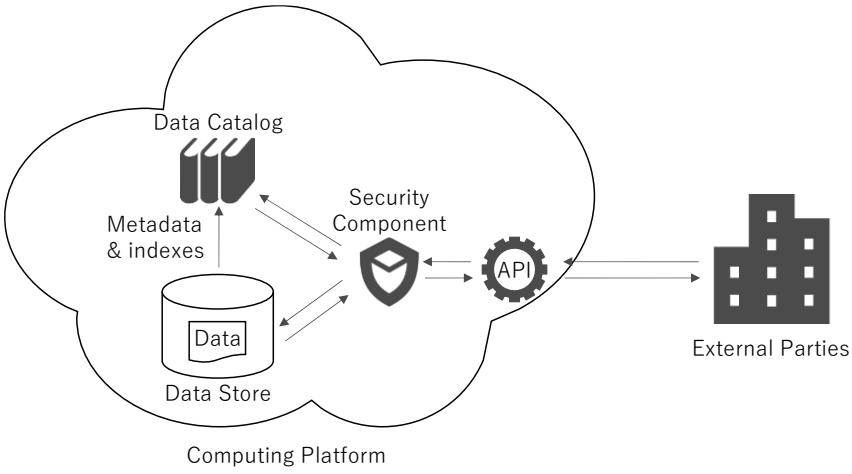

Figure 17. Secure data sharing

4.4.10. Distributed Data Processing

Data is distributed in the AECC system and may need to be rationalized to achieve a consolidated result. Such distributed data processing can only be performed efficiently by understanding the overall structure of the system and the locations, amounts and status of the data.

For example, the time required for each data processing instance may be proportional to the amount of data. To process data, it might be necessary to know the format, location, amount of data and the amount of available system resources at each computing platform.

Figure 18 shows an image of an AECC system with a data catalog and a configuration-and-control database:

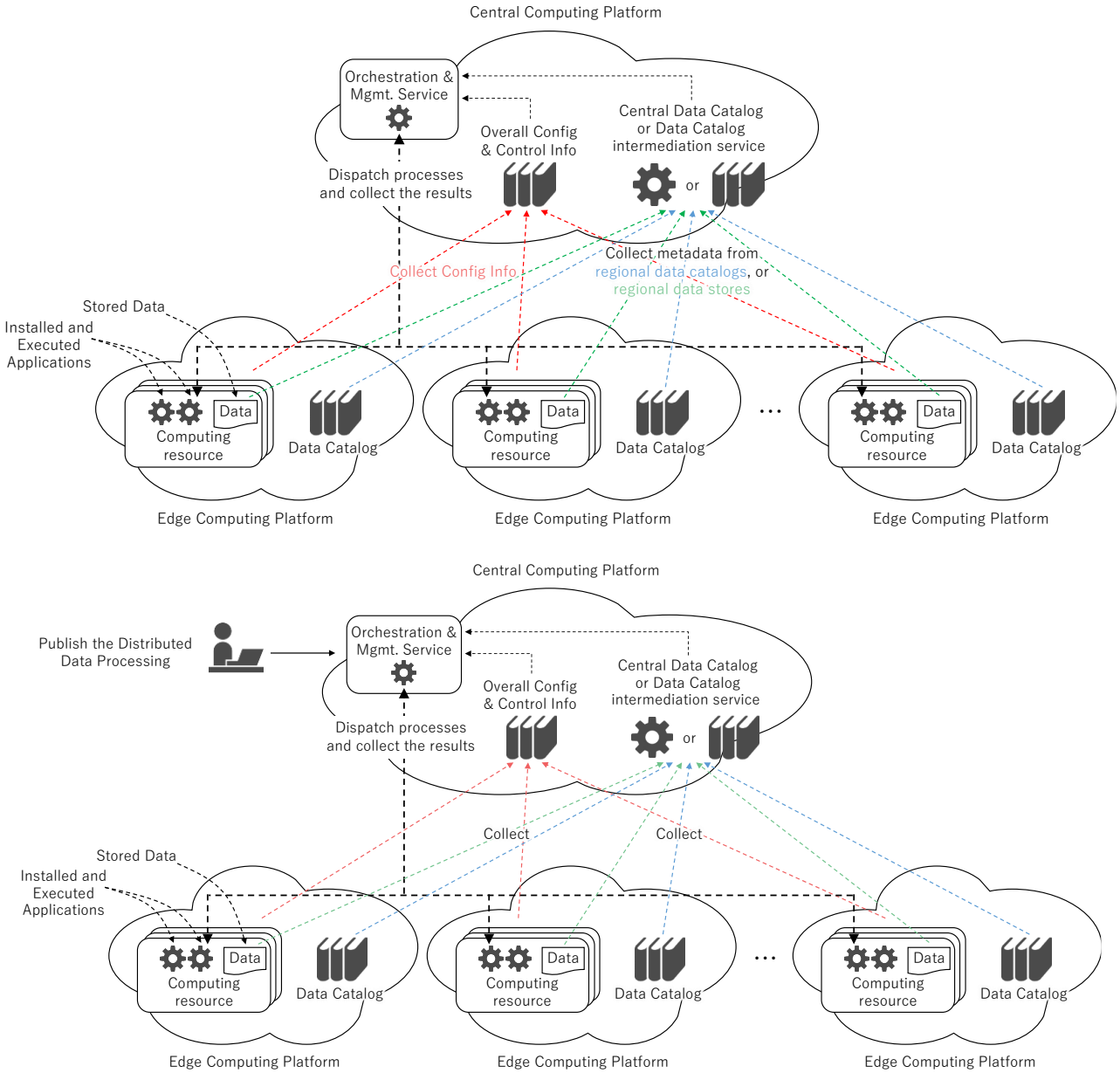

Figure 18. Distributed data processing

4.4.11. Federated Data Processing

Federated data procedures process the data in a distributed manner similar to the distributed process described above, but guarantees that the data does not cross the boundaries of data ownership.

This is especially beneficial because it allows the data to be used while reducing data movement. Moving large amounts of data can be costly and time-consuming.

Federated data processing is the solution in such situations. Highly sensitive raw data does not cross the system boundaries, and only acceptable intermediate output data is shared across multiple mobility service providers.

Figure 19 shows an image of such a federated data processing scheme supported by the AECC solution framework.

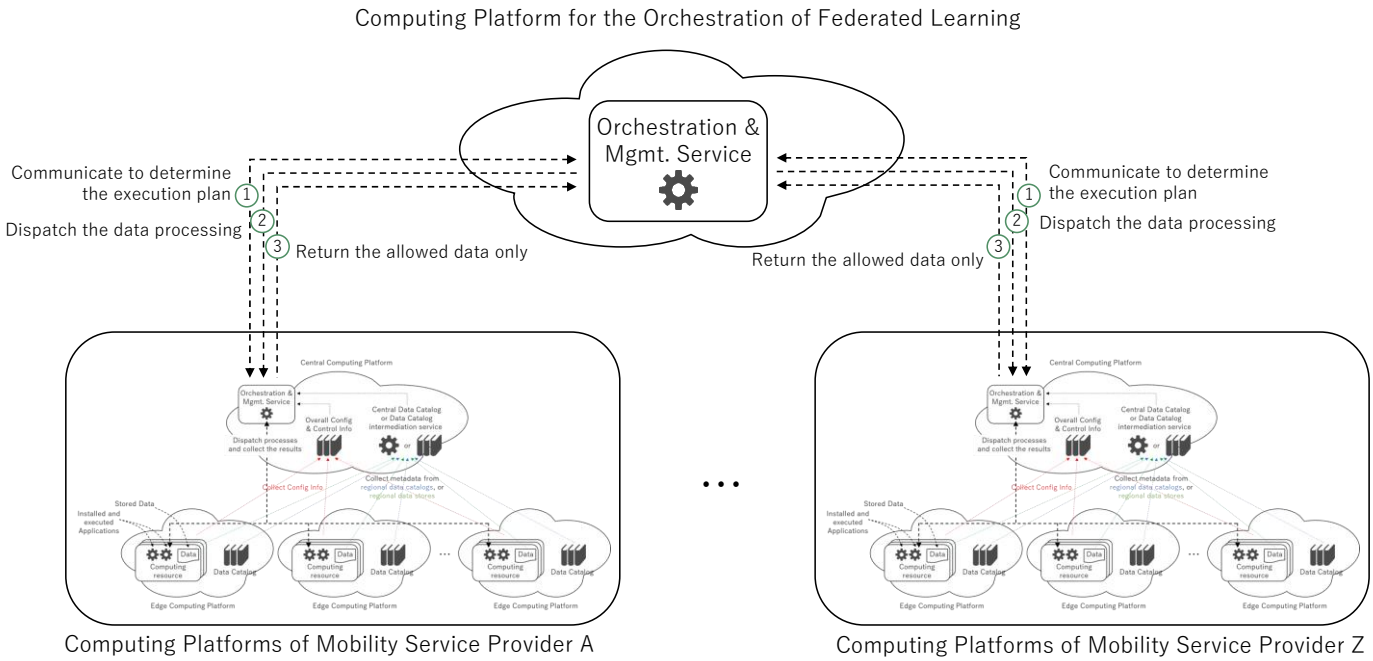

Figure 19. Federated data processing

5. AECC Data Management Requirements

5.1. General Requirements

General requirements define the common requirements applied to the entire AECC data management system. While there are multiple options for designing the system, adhering to general requirements can ensure the overall optimization of the system and its healthy growth.

1. Reducing the Network Data Transfer Burden

The burden of network data transfer must be minimized, by utilizing distributed computing architecture. To achieve this, consider the following points:

1). Collect data from a running vehicle by considering not only the workload distribution but also the distance between the vehicle and the edge computing facility. This means, for instance, if the nearest edge computing platform there is out of capacity and/or operation, then collect data in the next-closest computing facility. Such guidance will be provided to the vehicles by an off-vehicle mobility service.
2). Suppress data collection from running vehicles as much as possible. It is recommended to collect data as needed and stop data transmission from vehicles when a sufficient amount of data has been collected.
3). Optimize the timing of data collection according to requirements. If necessary, data can be collected immediately; otherwise, it is recommended to collect data at a time when the network is not congested.
4). Consider access network performance and the cost of data transfer. For example, if immediacy of data collection is not required, but a large amount of log data needs to be collected, and the availability of low-cost Wi-Fi service is expected, then collect data when the vehicle is connected to the Wi-Fi service.
5). If the edge computing platform is under-resourced and some processes need to be offloaded to another computing platform, such as the central cloud, choose the processes wisely so that data transfer is not increased too much.

2. Streamlining Data Flow and Data Access

In a hierarchically distributed environment, data is stored and processed on multiple computing platforms. Hence, it is recommended to build a good mechanism that streamlines data storage and retrieval. Asking the management component where to store data and/or where the necessary data is stored every time will cause delays in data processing. To avoid these, consider the following key points:

1). Establish data distribution/placement rules among data publishers, processors, consumers and system management (AECC services) to determine where data should be stored and where data is stored.
2). Keep data distribution/placement rules up to date. When data distribution/placement rules change, they must be quickly communicated to the publishers, processors, consumers and system management involved.
3). Design rules to evenly distribute data across instances while controlling relevant data to be stored nearby if it is expected to be processed at the same time.
4). Simplify rules so that their delivery can be done with short latency and a low workload.
5). Make data write access pinpointed to the appropriate instance based on the rules described above.
6). Update metadata and indexes of the data in the local data catalog when data is stored. Some of them may be replicated to the central data catalog in an asynchronous manner if global data access needs to be supported.

7). Make data read access pinpointed to the appropriate instance and the storage block of the instance based on the rules, the metadata and the indexes.

3. Data Processing Efficiency

Resources at the edge computing platform may be limited when compared to the hyper-scale cloud. Therefore, it is recommended to process data efficiently as much as possible while respecting resource constraints. For example, the following points may be considered:

1). Deploy the data processing application to a location near the target data, especially if the application needs to access a lot of data frequently.
2). Design the data processing pipeline to consist of various steps, such as queuing, shuffling, caching, transformations, etc. so that data transfer among them can be minimized.
3). Design the infrastructure elements, virtualization, workload orchestration, system monitoring, etc. so as to balance overhead and other benefits.

4. Effective Data Lifecycle Management

The AECC system must collect, process and store large amounts of data. Accordingly, to reduce data storage costs, it is necessary to store the data appropriately considering the requirements of each stage of the data lifecycle. For example, the following mechanisms are required:

1). Data should be moved to less-expensive storage when it is accessed less frequently. For example, data stored in expensive storage at an edge computing platform might be moved to storage in the central cloud at night when the network is not congested.
2). Convert data at the appropriate time to reduce storage size. For example, structured or semi-structured data may be compressed in a column-oriented manner, or images and videos may be compressed in a different Codec standard.
3). The management unit of the data will vary according to its lifecycle stage. For example, detailed data records will be created for fresh data, but these will be combined when data access becomes less frequent. By combining multiple records, data can be compressed more, and the burden of managing metadata and indexes can be reduced.

5. Dynamic Scalability

Data management and data processing must be flexible and scalable. Because these workloads vary with traffic, the system must be able to scale dynamically online. To achieve this, the following mechanism should be implemented.

1). Both online scale-up/down and scale-out/in must be supported. The reason is that scale-up/down can adjust system capacity without moving data, but its range is limited. On the other hand, scale-out and scale-in involve data movement but allow for greater capacity adjustment.
2). Scale-out/in operations should be done with data movement minimized. This means that it is required to use a well-designed data distribution algorithm and/or shared storage or memory grids that are decoupled from computing nodes.
3). The configuration and change information of the AECC system must be kept up to date and distributed without delay to the relevant AECC system elements.

6. Robust Security and Privacy

As data sent from vehicles may include sensitive information, the data and the data traffic should be securely protected by default. For example, vehicle ID and vehicle location information are considered private information. Image data collected from a vehicle may include persons unrelated to the vehicle. The following measures should be implemented for those purposes.

1). The combination of vehicle ID and vehicle location is private information. Therefore, it must be adequately protected. In addition to encrypting the payload of the transmitted message, the header information must be protected, as external attackers might peek into it. For example, if the messages are sent directly from the vehicle to a location-specific data processing instance, there is a risk that the vehicle location could be hacked, because the destination IP address of the message is linked to the location. Such a configuration should be avoided.
2). There is a risk that individual drivers can be identified by their stopping locations and driving route patterns. This data must be adequately protected. For example, an application that determines the degree of traffic congestion should only be able to access statistically processed driving data by road segment. Whatever the application, such data access controls must be implemented.
3). People may be captured in the image data collected from the vehicle. Therefore, image data must be handled securely. For example, images of pedestrians or drivers of other vehicles should be automatically excluded from image recognition, or image data that has not been confirmed as masked correctly should not be exported outside the system.

5.2. Functional Requirements

As described in Section 4.4, the AECC system should be designed to support several common data processing and management patterns required for mobility services. However, because of the variety of data (millions of different vehicles and different location points), dataflow rate (several Tbps), the amount of stored data (Exabytes-class data) and required data processing latency, there may be various challenges that might not be fulfilled by today's technology.

In this section, we will discuss such challenges and the requirements of the AECC data management system.

5.2.1. Data Collection Pipeline with Data Shuffling and Data Sampling

Challenges

As described in Sections 4.4.1 and 4.4.2, data is not just collected; it may need to be shuffled, or sampled so that it can be effectively and efficiently processed. Since a very large amount of various types of data is continuously collected from many vehicles and needs to be processed at very low latency, it is extremely difficult to build such a data pipeline mechanism.

Note: IoT Engineers may say that Kafka or Pub-Sub technologies can be used for this purpose. However, it is not so easy, because:

- Millions of partitions in Kafka technologies are difficult to manage and costly. In typical implementations of these services, data can be grouped and stored together, which is called a "partition," and partition can be used to accelerate data access. However, adding more partitions tends to incur more CPU and memory overhead, as each partition is tied to a different set of storage blocks and the number of partitions per CPU

core is limited, for example, up to 10; otherwise, the performance is degraded[IoT-Partition-1][IoT-Partition-2]. This means that data pipelining efficiency is not good if there are millions of partitions to be managed with fewer than tens of instances, which is certainly the case during low-traffic hours such as nighttime.

- Typical Pub-Sub technologies may not be inefficient, because each of the vehicle systems will subscribe and unsubscribe frequently to the topics they are interested in; for instance, on-road safety events, as they run through the area, if the topic is designed based on the location. However, such a subscription state may be rarely referenced.
- Configuration change to deal with a site-level failover will take some time, such as more than tens of seconds, to be effective across the entire system when there are too many vehicles to signal the need to reconnect. The same is applied to a server-level failover if a round-robin type of load-balancing mechanism is used. This poses a challenge in implementing real-time services with very high SLA.
- If collected data is grouped together in location-specific servers and the vehicle systems are configured to directly connect to one of those servers, such configuration tends to lead to invasion of privacy, because hackers can know where the vehicle is driving by looking at which instance the packet is sent to.

Requirements

To overcome challenges related to the largest instances of data shuffling and sampling, consider the following requirements:

- Let the vehicle attach some sort of tag to the data when it sends data to the mobility service, which can be used for shuffling. Such tags are, for example, something that can identify a location.
- Let the gateway extract and group multiple relevant data records by vehicle, location and/or time with short latency, so that time-series analysis can be executed smoothly.
- Manage system configuration information so that it is always up to date. At the same time, status during a configuration change is also managed. For example, if an instance needs to be excluded from the cluster, then it is flagged so that new data does not flow into it, and so on.

The above solution concept is currently being evaluated within the AECC. More detailed solution configurations are given in Chapter 6.

5.2.2. Data Pre-processing

Challenges

The next thing to accomplish is data pre-processing. This includes feature extractions and data compression as described in Sections 4.4.3 and 4.4.4. By processing data in such a way, data size can be greatly reduced. This also makes it easier to perform geo-distributed processing using the resources at the central cloud in a remote location. These processes are very computationally intensive and will require special resources such as a graphics processing unit (GPU). However, such resources are very expensive and limited, and therefore they must be utilized fully without having any idle time.

Requirements

To utilize resources such as GPUs fully, consider the following requirements:

- Allocate resources by purpose. This can reduce the reloading of model and reference data.
- Keep track of the workload status of resources and make data processing requests appropriately so that a resource does not become overloaded or idle.

- Optimize the network connection to special resources. For example, allocate enough bandwidth or apply a low-overhead protocol like RDMA.
- Scale the cluster of special resources considering its overall workload status and the length of the data queue waiting to be processed.

5.2.3. Insight Data Generation

Challenges

Once the features have been extracted, then insights will be generated based on them. This corresponds to a statistical verification process, because feature data instances extracted from individual pieces of data vary in reliability and quality. Each vehicle has sensors with different temporal or spatial resolutions that capture the same event from different angles at different times. Also, the situation may have changed since the time of sensing. Therefore, it is unavoidable that there will be differences in reliability and quality.

Accordingly, feature data must be reserved over time and statistically processed collectively to generate reliable insights. However, with a wide variety of feature data being generated continuously, it is challenging to perform such processing continuously with low latency. Achieving this will require a system that is optimized with respect to both hardware and software.

Requirements

To make this insight generation process work efficiently and fast, consider the following requirements:

- Manage feature data with attributes in in-memory space. Some attributes are used to extract related feature data tied to an interest in the mobility service, such as location, vehicle, etc. Other attributes, such as a timestamp indicating when the feature data was generated and a confidence level to manage the reliability of the feature data, are used during insight data generation.
- Statistically process the relevant feature data together. In doing so, each feature data instance can be weighted appropriately. For example, the confidence level can be decreased with the time since the feature was extracted, considering how long an event identified as insight will last in general.
- Purge the feature data from online processing periodically. As new feature data will be constantly inserted, a certain amount of data storage space may need to be reserved for this. To that end, feature data that is outdated and no longer contributes to insight generation may need to be archived and purged regularly.

5.2.4. Insight Data Management

Challenges

The insight data generated must be managed, because even in a single mobility service, many client vehicles will refer to the same data. Also, various mobility services may refer to the same data. Therefore, insight data must be managed appropriately.

However, it is not so easy. In particular, the following scenarios will present difficult data management challenges with respect to data volume, data processing frequency, link management between data records, etc.:

- Manage parking space availability, accident situations, etc. These must be mapped and managed on top of static road map information. In addition, such information will be updated very frequently. For instance, as

described in Section 3.1, we assume that the frequency of updates is 200,000 times per second, and there will be more read accesses, in the case of the available parking space detection service.

- Manage vehicle traffic congestion, pedestrian congestion, traffic control, etc. These could be managed by road segments. As described in Section 3.3, there would be one million road segments in an urban area, and if the information is updated every 10 seconds, then the update frequency is 100,000 updates per second. And obviously, there are many more read accesses. This is a huge workload and challenging, too.
- Manage emergency vehicles, reckless driving vehicles, tottering bicycles, etc. These objects to be watched must be tracked along with their movement on top of static road map information.
- Manage a static map, e.g., road structures, road paints, roadside buildings, etc. These need to be updated with consistency, because these are geo-spatially connected. And the same is true if these are distributed and managed at multiple edge computing platforms. In addition, it will be required to maintain a history of map data updates, because there will be multiple mobility service arrangements already created based on the old map.

Requirements

To address the insight data management challenges described above, consider the following requirements:

- Manage location-specific information efficiently with fewer resources and with lower latency. The AECC data management system may need to support hundreds of thousands of updates per second and millions of reads per second at the same time, to enable various mobility service use cases.
- Manage and keep track of moving objects efficiently with fewer resources and with lower latency. For some mobility services, moving objects, such as reckless driving vehicles, may need to be managed.
- Manage links between static structures and dynamic events. In a mobility service, the various events and moving objects of interest may need to be managed on a static map. Therefore, the mechanism to refer to the position on the map must be efficient.
- Manage the historical change to insight data. It is recommended to manage when and how the data is changed. Otherwise, if the arrangement of the mobility service needs to be changed, it is not possible to explain the reason for the change to its affected users.
- Update the insight data across edge computing platforms with consistency. Map data may be distributed and managed at each edge computing platform. In such cases, each edge computing platform manages map data for its assigned range. However, if the data update is completely closed at each edge computing platform, inconsistencies may occur in the map data near the boundaries. To avoid such a case, sometimes the insight data may need to be updated with consistency across the edge computing platforms.
- Scale the insight data management system according to the amounts of data volume and data processing. Traffic volume changes from moment to moment. It fluctuates greatly, especially when traffic jams occur due to accidents, weather or other reasons. These fluctuations have a significant impact on the amount of data and processing in the insight data management platform. Therefore, it is recommended to design the system so that it can scale flexibly. In addition, it is recommended to assure independent scalability of data storage and data processing, as each mobility service may have different requirements.

5.2.5. Push Notification

Challenges

As described in Section 4.4.7, there is a case where insight data needs to be pushed to nearby vehicles, or vehicles heading to a point of concern. This is an important mechanism for the safe driving of vehicles and convenient service.

It is possible to broadcast the information and let vehicles react to it based on their positions, but this increases network traffic and consumes the vehicle system's computing power, so it should be avoided. Instead, the information should be delivered only to the nearby vehicles as much as possible. However, this requires a mechanism for tracking the location of millions of vehicles at any given time.

Requirements

To enable the short-latency push notification to running vehicles, consider the following requirements:

- Prepare for the rapid push notification in advance. To quickly send push notifications to vehicles that need information, it is necessary to manage where vehicles are driving and what information they are subscribing to in advance.
- Reduce overhead for push notifications. Today's real-time push notification implementations, such as Apple Push Notification Service, often assume that TCP connections are maintained and keep-alive signals are sent every 15 minutes or so. To reduce the cost, it is recommended to develop a mechanism to reduce such a connection burden, such as a new protocol and/or a new access network technology that supports the push notification with fewer resources.

5.2.6. Data Catalog

Challenges

In the distributed architecture envisioned by the AECC, data is distributed across different instances at different sites. In such an architecture, it is required to manage where and what kind of data is stored, to make good use of it. The mechanism for that is the data catalog and the data index management system.

However, it is not easy in the environment that the AECC envisions. This is because there are many computing platforms and a wide variety of different kinds of data must be managed. Data may be moved or duplicated between sites. Therefore, such status must also be managed. In addition, some data may become less reliable over time. Attributes associated with such data quality must also be managed.

Requirements

To build the data catalog and indexing system required in the AECC system, consider the following requirements:

- Manage data catalog and index information globally if global data access needs to be supported. When data catalog and index information is generated locally at each site, this information will be shared between sites for global data utilization.
- Manage various kinds of data. Mobility services require various kinds of data, including three major categories of data, i.e., structured, semi-structured, and unstructured data, and within each major category there are sub-categories, such as JSON, XML, video, image, fixed-column table, wide-column table, etc. Therefore, it is recommended to manage data catalog and index information while considering the differences between these data types.
- Manage data schema and APIs to access the data. An important purpose of the data catalog and index is to streamline data access and usage. For this purpose, it is recommended to manage the data schema and API details required for data access, too.
- Detect the data platform information. To generate data catalog and index information, the underlying system that stores and manages the data may need to be detected and managed, too. Some of these would be a

specific IaaS or PaaS service in the public cloud, and some others would be the OSS-based infrastructure. Therefore, it is recommended to use technology that can detect and manage various kinds of infrastructures.

5.2.7. Secure Data Sharing

Challenges

Sharing data with external parties while ensuring that sensitive data and private information are protected is not so easy. This is because various data protection policies need to be considered, and various kinds of data need to be managed. In addition, different policies may have to be applied to different external parties. With so many different services being launched, it is very difficult to ensure that these are implemented and operated correctly.

Requirements

To enable secure data sharing among multiple parties, consider the following requirements:

- Define policies flexibly. As there are many different types of data and different types of users, various policies may need to be defined. For example, the vehicle registration number, or information that looks like a vehicle registration number, may need to be filtered. In addition, different users may have different data access and usage rights. Therefore, policies need to be constructed flexibly, in a data-centric and role-based manner.
- Apply policies automatically. Data sharing will be slowed down if the policies have to be checked manually every time.
- Standardize security mechanisms. There will be various mobility services, various applications and various data in the AECC system. Implementing a security mechanism for each of these is time-consuming, and the possibility of implementation errors becomes very high. Therefore, it is recommended to standardize security mechanisms whenever it is possible and delivers adequate results.

5.2.8. Distributed Data Processing

Challenges

In a distributed environment, it is necessary to manage where and what data is, as well as the topology of the system. Considering today's cloud-based environment, system topology includes which and what kind of computing platforms are available, what and how many resources are available at each site, how much bandwidth is available, how much latency is required between resources, and so on.

Based on such knowledge of the system topology, an optimal distribution plan for data processing will be determined. To do so, end-to-end requirements and operation policies of mobility services are considered first, such as end-to-end latency, data processing efficiency objectives, etc., and then possible patterns of data processing are simulated and compared.

If geo-distributed data processing based on multiple computing platforms from different vendors is assumed, the above task becomes very complex and challenging.

Requirements

To realize such optimized distributed data processing, consider the following requirements:

- Manage the system configuration information. This means managing what workloads (including data management) are running on which resources at which computing platforms, and what networks are used to connect resources.
- Manage the system control information. To determine the optimal distributed data processing plan, it is recommended to manage control information, such as which computing platforms are available, what level of resources at each computing platform can be used with what priority, and how much of the resources needs to be assigned to the workload considering the amount of data and the data processing type.
- Manage the distributed data. It is also recommended to manage the content and location of distributed data to determine the optimal distributed data processing, too. This can be achieved with the support of the data catalog and the data indexing system.
- Orchestrate the distributed data processing. Once the optimal distributed data processing plan has been determined, then the instances are launched, the workloads are deployed and the data processing is started according to the plan. Therefore, it is recommended to build a component responsible for such orchestration.
- Manage the distributed data processing. Once data processing has been successfully launched, then it is recommended to monitor its execution status continuously, so that failures and exceptions can be detected as soon as possible to take adequate actions for recovery.

5.2.9. Federated Data Processing

Challenges

To increase the ROI of mobility services, it is important to reuse the data collected and the insights generated from it as much as possible. The scenarios described in Sections 3.2 and 3.3 are such examples. However, the data collected from connected vehicles includes very sensitive information as well as the private information of the vehicle owner, etc., so it cannot be disclosed to an external party without pre-processing the data appropriately. Therefore, direct data exchange is not possible in practice, and data must be pre-processed and secured before it can be shared with external mobility service providers.

However, if only a very limited amount of data could be shared, the reusability of the data would be greatly compromised. For example, there may be a mobility service provider who wants to analyze image data and quantify the roadside bustle but does not have sufficient image data. Such a provider would ask other providers who have a great deal of image data to let it use the data. However, the providers who have data may not be able to provide the raw image data, because such images could contain private information.

This requires more advanced distributed data processing, called federated data processing. With such a mechanism, the provider can publish a data processing request to the other provider's system and analyze the raw data of other providers there, while ensuring that the raw data is actually not exported beyond the boundaries between providers and only results that do not contain sensitive or private information are returned across the boundaries.

Requirements

To support federated data processing in the AECC system, consider the following requirements:

- Accept data processing requests from external parties. To improve data reusability across multiple providers, it is recommended to provide a mechanism to accept data processing requests from other providers.
- Run the workloads of other providers securely. The data processing request may contain malware, etc. Therefore, run the accepted data processing within a secure area to defend against any attacks. The required data is securely passed into the area.

- Return the results of the analysis after ensuring safety. If the analysis process is completed, the results are returned to the provider who requested the data processing. When doing so, it is recommended to check whether or not the data to be returned contains sensitive or private information.

6. AECC Data Management System Cores

While there will be several options in designing the system to meet the requirements described in the previous chapter, there are essential cores that are indispensable in the AECC data management system. In this chapter, we will discuss such data management cores.

6.1. Data Distribution Framework

Mobility services data is enormous. Therefore, it is impossible to manage and process data in one instance in one computing platform, and a framework for distributing data is absolutely needed when designing the system.

6.1.1. Global Data Distribution

The AECC envisions distributed data management utilizing edge computing platforms to handle huge amounts of data. From several to dozens of edge sites will be used, depending on the amount of data to be collected/processed and the area/size of the country/region that provides the service.

For data distribution between edge computing platforms, there will be two methods to control it: 1) letting the vehicle system send data to the appropriate edge computing platform, or 2) letting the common gateway forward data to the appropriate edge computing platform according to the location, in consideration of the driving position and workload status of candidate edge computing platforms.

When using the former method, it is necessary to inform the vehicle system about the configuration and change information of the edge computing platforms in advance. Also, as GPS contains errors, data may be sent to a different edge computing platform. Therefore, it is also necessary to ensure that data is properly transferred between edge computing platforms.

When using the latter method, computing resources, so-called mobile/multi-access edge computing or MEC, will be used to handle network packets. Such a mechanism will be available in the near future.

Figure 20 below shows an image of such global data distribution based on geo-distributed edge computing platforms.

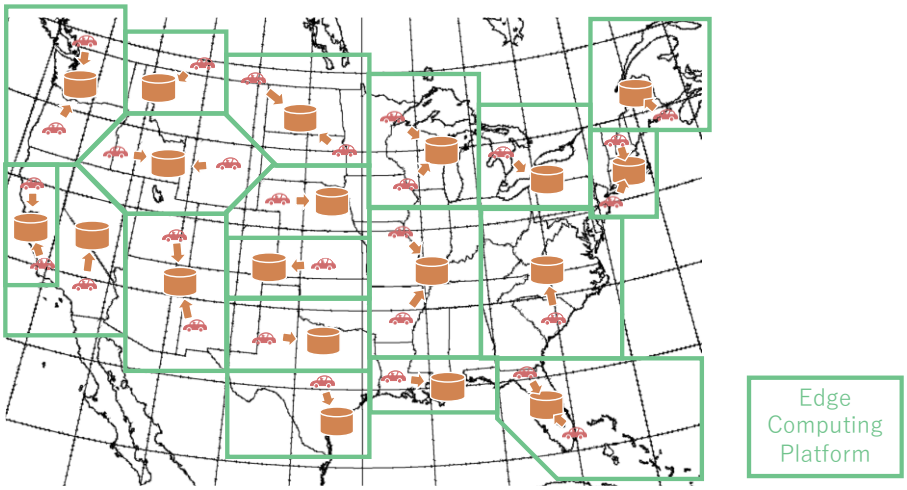

Figure 20. Global data distribution

6.1.2. Local Data Distribution

Even with the global data distribution described above, the amount of data to be processed at each site is still enormous. Therefore, distributed management of data is required even within each edge computing platform. To determine the appropriate data distribution in each edge computing platform, it is necessary to understand the kinds of data and their data processing requirements. In mobility services, the following data types need to be considered:

A. Raw/input data
  A-I. Cruising data, such as the vehicle's internal sensor data
  A-II. Image data, collected from vehicles or roadside units to know the conditions of a specific location
  A-III. Mobility service request data, to find an available parking space, etc.
B. Insight information (processed data)
  B-I. Vehicle status data, such as failure, the position of emergency vehicles, etc.
  B-II. Location event data, such as parking space availability, accidents, etc.
  B-III. Traffic condition data, such as traffic jams, snow-covered or icy roads, etc.

Each data type has different usage purposes and data management requirements, which data distribution must also take into account. In the following sub-sections, the results of such analyses are described.

A-I. Cruising Data

Cruising data is mainly used for ascertaining the condition of a moving vehicle and its driver. By collecting and analyzing cruising data, it will be possible to notify the driver before running out of fuel, predict a vehicle failure and detect a driver's lack of ability to control the vehicle or to stop safely.

Therefore, the cruising data needs to be analyzed per vehicle ID. And, to run the analysis efficiently, it will be necessary to group the data in advance for each analysis unit, that is, by vehicle ID. To group the data by vehicle ID, the data distribution will be determined by the hash value of vehicle ID and so on.

Of course, cruising data can also be used to find out location-specific information. For example, if sudden acceleration is detected, an accident may have occurred at that point. However, even in such cases, data will be collected by vehicle ID as it requires time series analysis.

Collection of cruising data might be collected by default, and its collection cycle depends upon the mobility service provider's policy. It should also be considered that the number of vehicles managed will fluctuate depending on the time of day. Therefore, the system should be designed for dynamic scaling. Data distribution must be designed to make it easier, especially when considering scale-out and scale-in operations. A technique commonly referred to as a consistent hash algorithm will need to be used.

Figure 21 below shows an image of such vehicle ID-based data distribution.

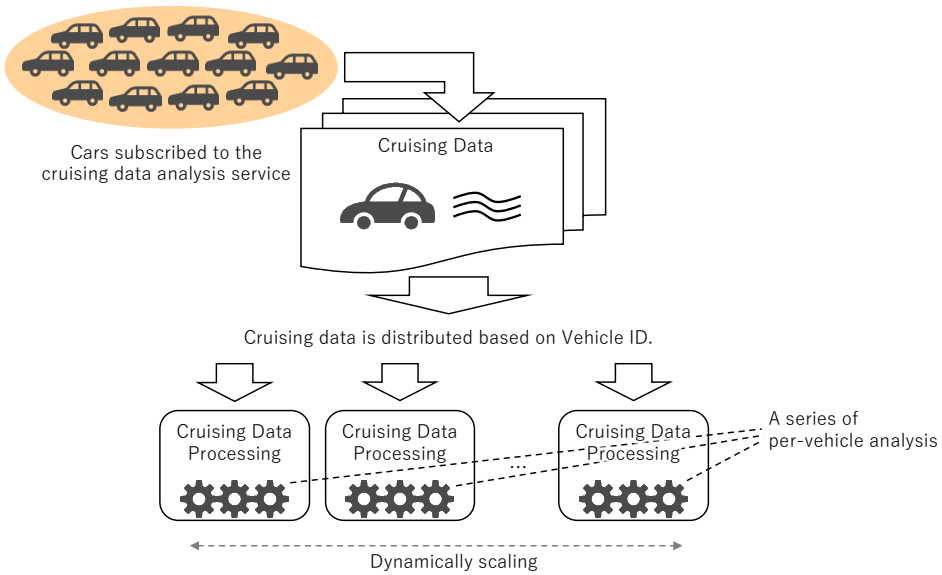

Figure 21. Vehicle ID-based data distribution

A-II. Image Data

Image data is mainly used to ascertain the conditions of a specific location. Data collection may be performed based on a request from the mobility service instance, or by intra-vehicle system judgment, such as the detection of sudden acceleration. By collecting and analyzing image data, location-specific conditions, such as parking space availability, the occurrence of accidents, etc. can be detected.

To know the location-specific conditions precisely, image data will be collected from multiple vehicles and jointly analyzed. Accordingly, it would be reasonable to manage the image data by location. Uber's Hexagonal Hierarchical Spatial Index (H3) [Uber-H3] is a good example of such location-specific data distribution.

Figure 22 below shows an image of such location-based data distribution.

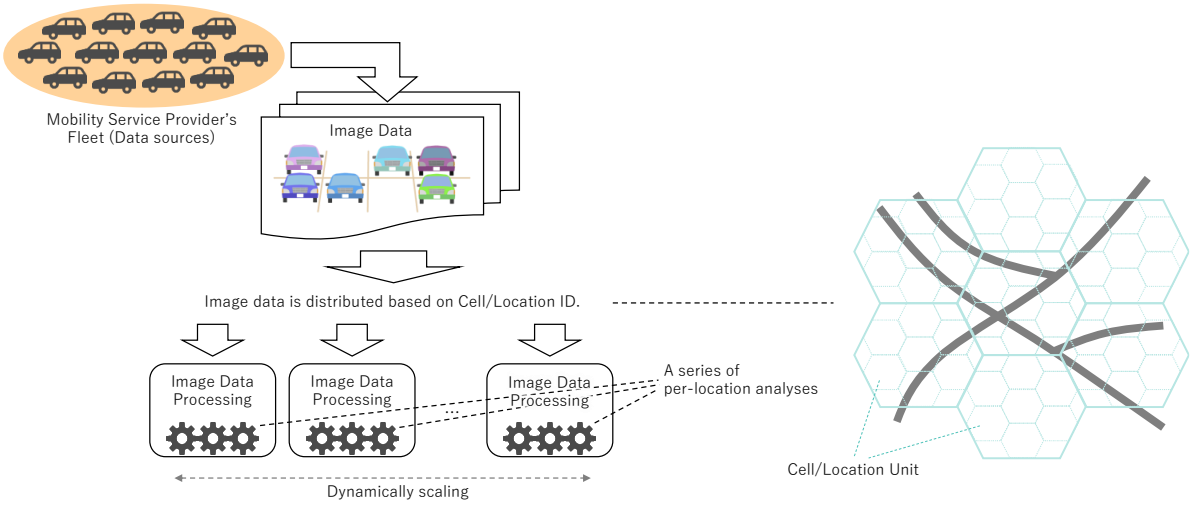

Figure 22. Cell/location ID-based data distribution

A-III. Mobility Service Request Data

Mobility service requests are made to find vacant parking spaces, find the best route to avoid traffic jams and arrange for ride-sharing vehicles. Such requests will be made very frequently, given the millions of vehicles and tens of millions of potential users. However, if counted by vehicle and user ID here, the occurrence is very sparse. Therefore, the conventional web-type load balancing is appropriate for receiving and processing mobility service requests on each edge computing platform.

Figure 23 below shows an image of such data distribution of mobility service requests.

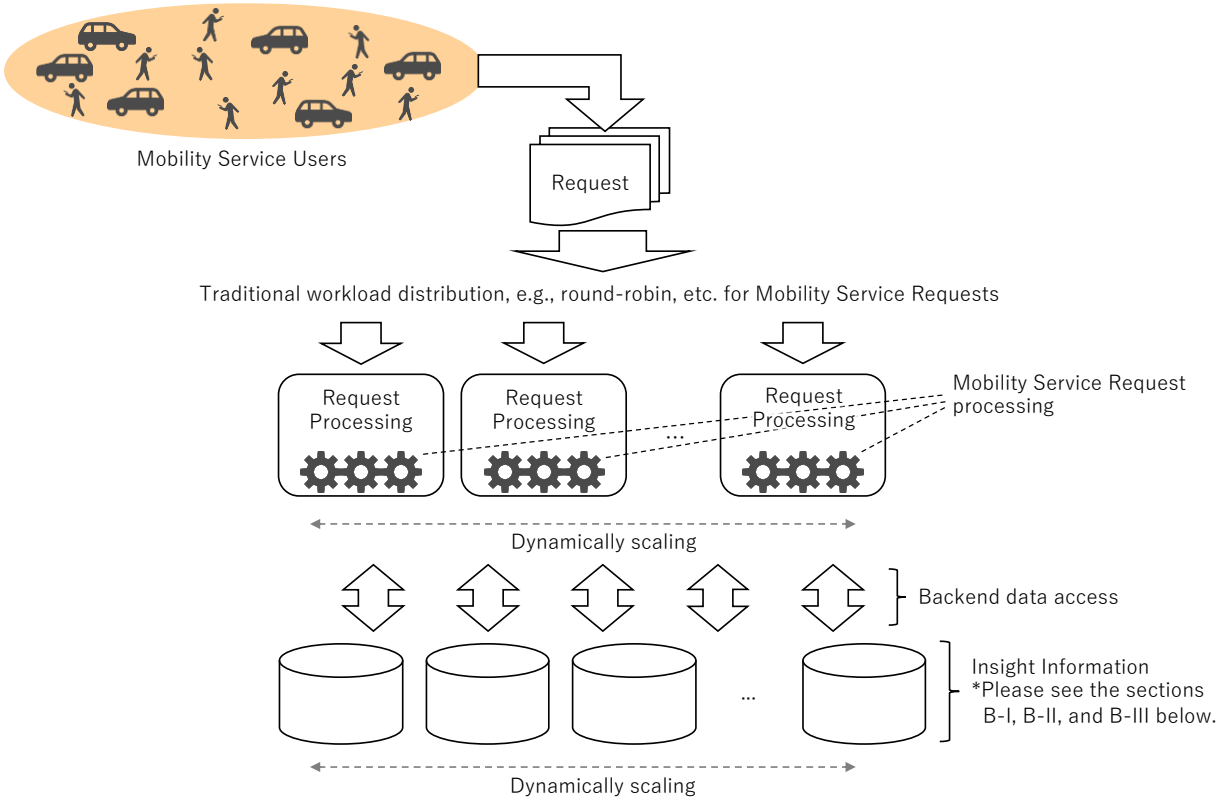

Figure 23. Cell/location ID-based data distribution

B-I. Vehicle Status Data

Vehicle status data represents the status of each vehicle and its driver, such as the condition of the vehicle, the driver's fatigue, the position of an emergency vehicle, etc.

This information is mainly used for mobility services that manage the information and orchestrate the service based on vehicle ID. For example, it is possible to discover the position of the vehicle/fleet of concern or to issue an alert when an abnormality is likely to occur in the vehicle.

The distribution of such vehicle status data will be based on the vehicle ID, similar to A-I, cruising data. However, regarding flexible scalability and reliability, the instance to manage the vehicle status data will most probably be different from the instance used for cruising data processing. The relationship between these two types of instances could be 1: 1, 1: N, N: 1 or N: M, depending on the mobility service requirements and the mobility service provider's design policy.

Figure 24 below shows an image of such a relationship between cruising data processing instances and vehicle status data management instances, assuming the N: M relationship is chosen.

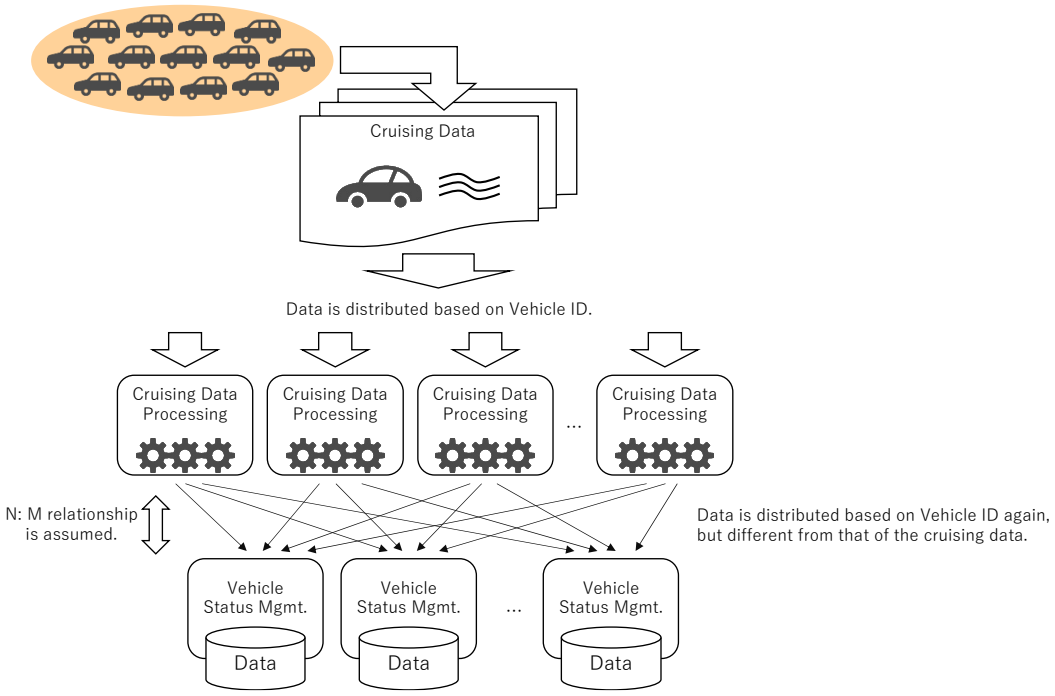

Figure 24. Relationship between cruising data processing and vehicle status data distribution

B-II. Location Event Data

Location event data represents location-specific events, such as the availability status of parking spaces, the occurrence of accidents, road closures due to road construction, etc. These events are managed by location and change over time. Because the system must be able to update location event data with good throughput at all times, distributed data management will be adopted here as well to achieve the required throughput, and it will be based on cell/location ID as chosen for the A-II image data processing.

However, it should be noted that location data is referenced by an unspecified number of mobility service users, while updated based on a limited number of inputs, such as images. So again, location data management instances and image data processing instances will form different layers, and their relationship will be 1: 1, 1: N, N: 1 or N: M depending on the requirements and design policy.

Figure 25 below shows an image of such a relationship between image data processing instances and location event data management instances, assuming the N: M relationship is chosen.

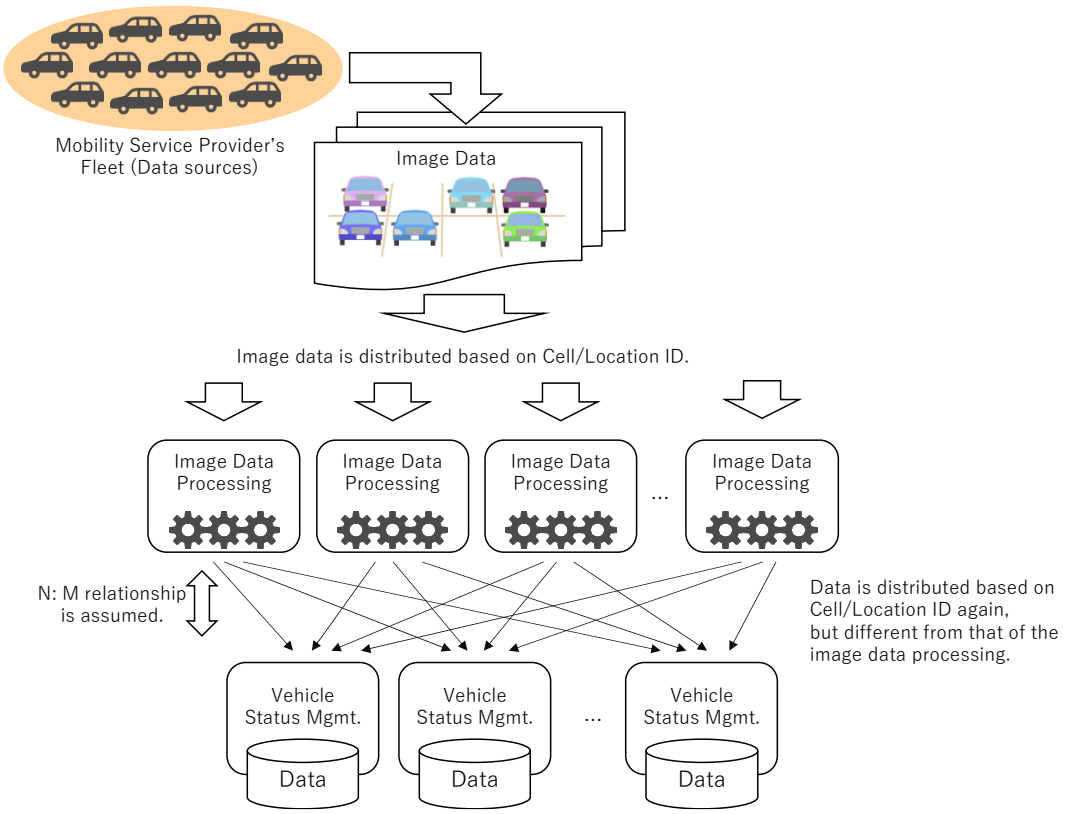

Figure 25. Relationship between image data processing and location event data distribution

B-III. Traffic Condition Data

Traffic condition data represents traffic conditions, such as traffic congestion, snow-covered and/or freezing roads, the presence of many pedestrians and bicycles, etc., which could be extracted from cruising data or image data analysis.

These situations often change over time and thus the characteristics of traffic condition data are similar to those of location event data. However, a particular traffic situation typically occurs in a certain range along the road, and the range of occurrence changes quickly, it is recommended to define a data model that describes such traffic situations effectively and efficiently.

Considering such data management characteristics, the data distribution scheme based on the cell/location ID adopted in A-II and B-II is not appropriate, and data distribution based on road segment ID will be more appropriate.

Figure 26 below shows an image of such distributed management of traffic condition data and its relationship to the cruising data and image data analysis.

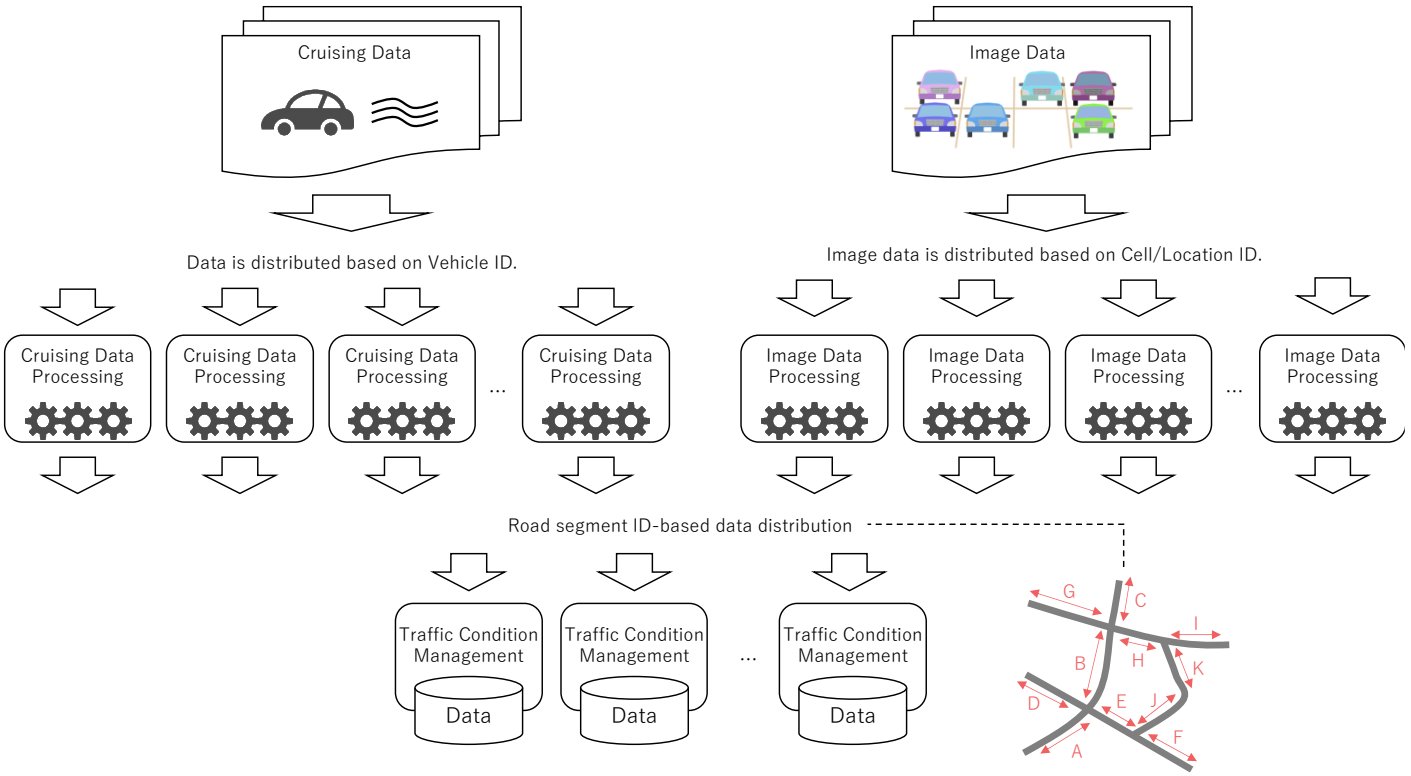

Figure 26. Distributed management of traffic condition data and its relationship to cruising data and image data analysis

6.2. Data Flow Controls in the Common Gateway

In Section 6.1, various data types and distributed data management models were introduced. To build a mobility service that efficiently processes a large amount of data, such a distributed data management mechanism must be used.

But what happens when vehicles and/or mobility service users must interact with these services? Of course, those clients could make contact with the instance where the target data exists if they know the data distribution in advance; however, that is not realistic, considering the following factors:

- Configuration Management
  It is very difficult to inform all clients how the data is distributed among instances and keep it up to date, considering millions or tens of millions of clients, and the configuration of a total of thousands of instances deployed on dozens of edge computing platforms.
- Increased latency due to dynamic connection switching
  Switching the instance depending on the situation leads to an increase in latency, because appropriate processes such as authentication/authorization and key exchange for communication encryption are conducted, to secure the connection.
- Limited resources of the vehicle system
  Mobility services should be provided not only to luxury vehicles equipped with high-performance computers for autonomous driving but also to ordinary vehicles equipped with small computers. It is not realistic that such a

vehicle system could understand the configuration of a complicated AECC system and switch the connection destination.

- Privacy protection
  As pointed out in Section 3.1, if each vehicle contacts the instance where the data is stored directly, the risk of being tracked by external attackers with respect to vehicle ID and/or the running position is increased. To mitigate it, direct communications to the instance where data of concern exists must be prevented.

For these reasons, the AECC assumes that a common gateway is located between the vehicle system and the mobility service instance to relay communications, and the gateway understands mapping between 1) a mobility service and a mobility service implementation, and 2) an anonymized vehicle identifier and a connection to the identified vehicle. The common gateway will act as a front-end layer of various mobility services or on the edge computing platform. An image of such a common gateway deployment was already shown in Figure 8.

6.3. AECC System Model

As previously discussed, the AECC data management system is a complex distributed solution where instances in different computing platforms are organically connected and manage the data together. To correctly grasp the whole system and orchestrate the data management and relevant workloads, it is necessary to precisely manage the configuration of the entire AECC system.

To ease the burden of managing such a distributed solution, the AECC has developed an AECC system model that represents the overall configuration of the AECC system. When designing the AECC system model, the following points have been considered:

- Computing platforms
  Central clouds, edge clouds and any environments in which data can be processed are represented as “computing platforms” in the AECC system model. These might be instantiated near an antenna or at a small facility such as a gas station. Types and capacities of CPU and storage are managed as their attributes.
- Networks
  Communication paths between sites, computing platforms and/or vehicle systems and sites are represented as “networks” in the AECC system model. Network categories and characteristics such as being wired or wireless, expected latency, bandwidth, etc. are managed as their attributes.
- Resource groups
  Groups of resources in each site are represented as “resource groups” in the AECC system model. For example, resources can be grouped by the mobility service provider that uses them. Another example is a concept called a placement group in the public cloud. This groups resources connected with a QOS-controlled network, which is also a resource group in the AECC system model. The types of resources available in a resource group are managed as attributes.
- Executables
  Software programs that are installed and executed on the computing platform are represented as “executables” in the AECC system model. Depending on their positioning in the AECC system, they will be categorized into one of the following service classes:
  a. Mobility Service Class: These represent software programs that interact with external clients such as vehicles to provide mobility services.
  b. Mobility Function Class: These represent software programs that are called internally within one or multiple mobility services but do not interact with external clients.

c. Infrastructure Service Class: These represent software programs that are used to manage the local resources in a site. These services may be pre-provided by the site provider or prepared by the mobility service provider.
d. AECC Service Class: These represent software programs that are used to manage and orchestrate geo-distributed data processing, as well as to establish interoperability between the mobility services belonging to different mobility service providers.

- Software images
  Software program templates that are referred to when installing the software program on the computing platform are represented as “software images” in the AECC system model.
- Dependencies between executables
  Executables may call or depend upon other executables in the AECC system. Such relationships are represented as “dependencies between executables” in the AECC system model. Communication methods, such as APIs involved, or process-to-process communications, synchronous/asynchronous communications, intervening load balancing mechanisms, parameter schemas used, etc., are managed as their attributes.
- Data
  Data stored on the computing platform and managed by one or several executables there is represented as “data” in the AECC system model. Data schemas and executables to manage the data are managed as its attributes.
- Endpoints
  Locations of APIs to interact with the executables of concern are represented as “endpoints” in the AECC system model.
- Vehicle system
  In the AECC system model, the vehicle system is supposed to consist of computing platforms, executables and data, just like the other center-side constructs. The difference is that the computing platform is connected to the sensors and actuators in the vehicle system.

Figure 27 below shows a view of the AECC system model.

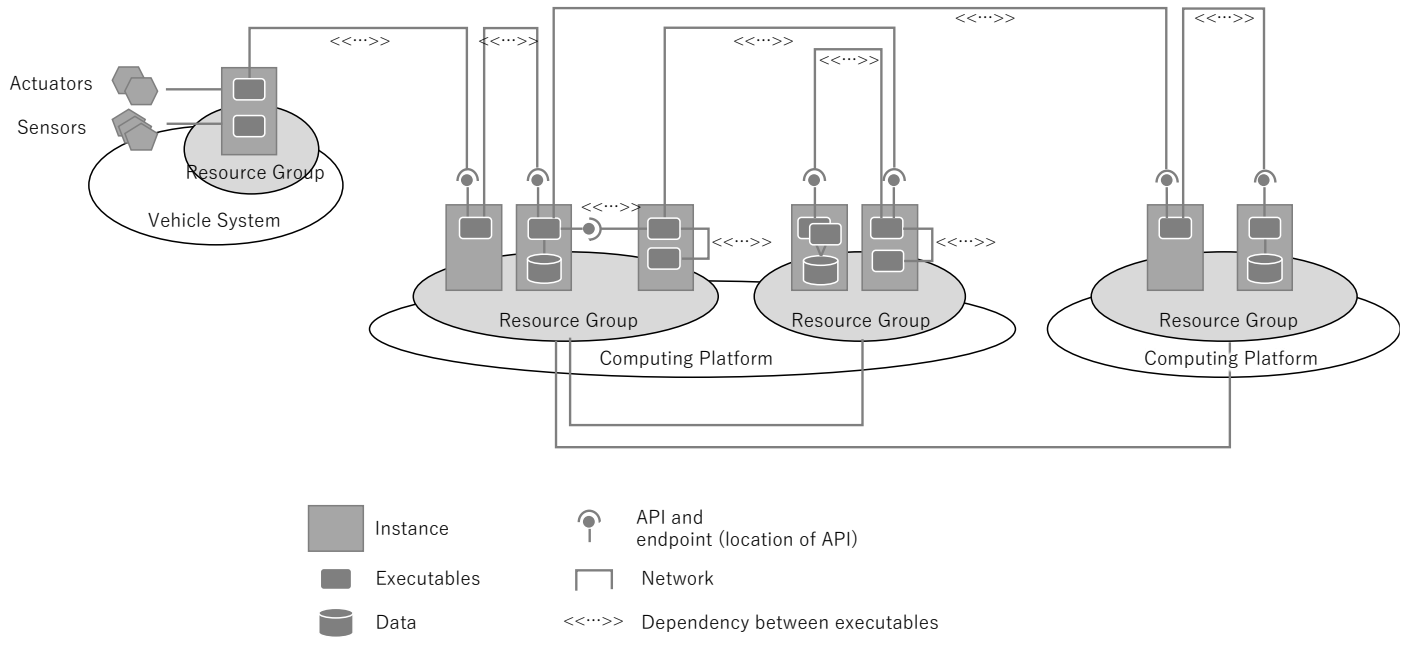

Figure 27. View of the AECC system model

7. Conclusion and Next Steps

As described in this white paper, the AECC assumes geo-distributed data management that utilizes edge computing platforms to accommodate huge amounts of data that cannot be managed by a single data center, i.e., the hyper-scale cloud. To achieve this goal, we have identified various measures to be implemented to manage and accelerate distributed data processing.

Such an architecture brings new complexity to the system that we do not need to consider when building the system in one data center. However, as analyzed in this white paper, this seems feasible.

Therefore, the next step is to determine detailed solution guidance to accelerate solution development, and to verify such solution concepts by conducting a series of proofs of concept. The results of those activities will be published by the AECC later. If you are interested in such efforts within the AECC, please contact AECC members and consider participating in our activities. The AECC always welcomes you.

8. References

New York Has 81,875 Metered Parking Spaces, And Millions of Free Ones, https://nyc.streetsblog.org/2011/03/22/new-york-has-81875-metered-parking-spaces-and-millions-of-free-ones/

Operational Behavior of a High Definition Map Application White Paper, https://aecc.org/wp-content/uploads/2020/07/Operational_Behavior_of_a_High_Definition_Map_Application.pdf

How to Choose the Number of Topics/Partitions in a Kafka Cluster?, https://www.confluent.io/blog/how-choose-number-topics-partitions-kafka-cluster/

The Power of Apache Kafka® Partitions: How to Get the Most out of Your Kafka Cluster, https://www.instaclustr.com/blog/the-power-of-kafka-partitions-how-to-get-the-most-out-of-your-kafka-cluster/

H3: Uber's Hexagonal Hierarchical Spatial Index, https://eng.uber.com/h3/

Appendix

Class Diagram of the AECC System Model

The following figure provides another view of the AECC system model based on a class diagram.

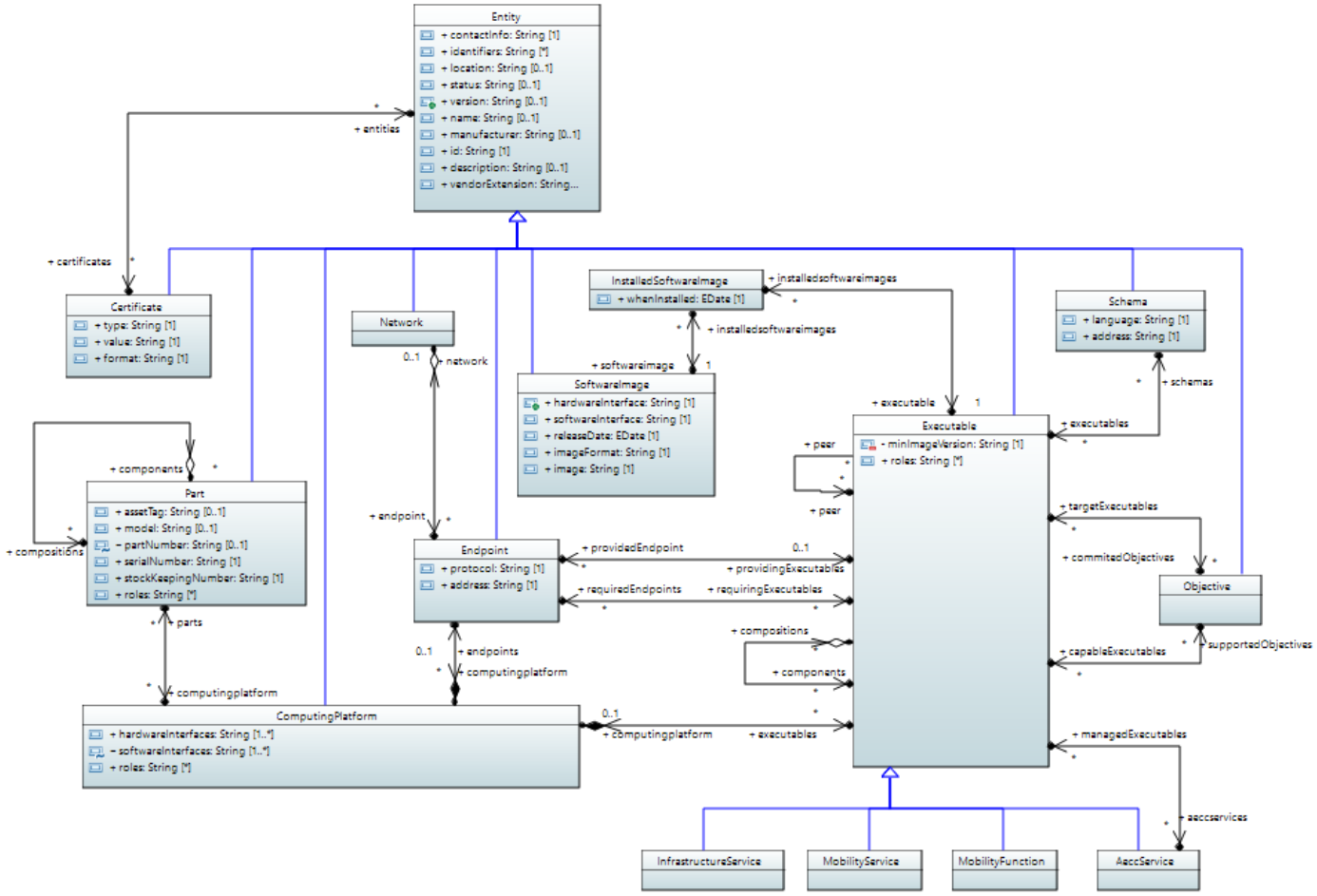

Figure A-1. Class diagram of the AECC system model