Activities and Results of Field Trials—Network Edge Computing Platform

Ryosuke Noji, Takayuki Kamei, Kaito Funabiki, Daisuke Murata, and Yuken Kanemaru

Abstract

Toyota Motor Corporation and the NTT Group conducted field trials to verify an information and communication technology platform for connected vehicles over a three-year period beginning in 2018 and shared their respective technologies and expertise regarding connected vehicles. Distributed processing using edge nodes increased the efficiency and processing speed of the platform. However, the distribution of processing sites created new challenges. To address these challenges, we developed an architecture in which multiple network functions are allocated to edge nodes and verified its effectiveness through these field trials.

Keywords: network, edge node, distributed processing

1. Introduction

At the early stage of the field trials conducted by Toyota Motor Corporation and the NTT Group, a large datacenter collected data from vehicles, and many servers were installed at the datacenter to process the data in a distributed manner. However, this architecture was not able to achieve the target scalability (handling 30 million vehicles) and processing time (sending a response within 7 seconds). To overcome this limitation, edge nodes were placed between vehicles and the datacenter, and some data processing was delegated to these edge nodes. Usually, the term "edge node" refers to a terminal device, for example, the computing resource in a vehicle in the case of a connected vehicle. With our new approach, however, edge nodes are also geographically distributed but located between vehicles and the datacenter.

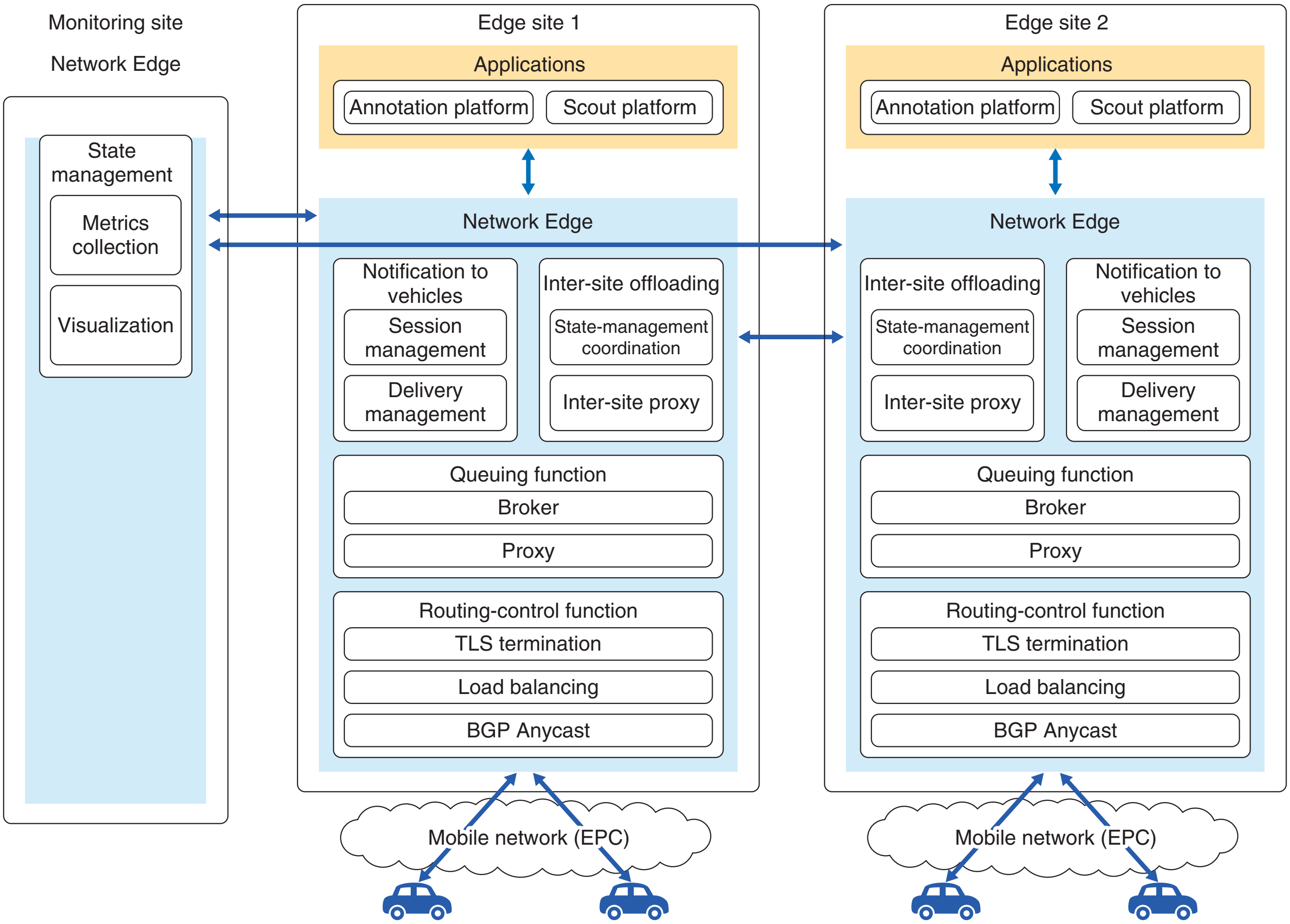
Challenges in using edge nodes include not only the need to establish the appropriate application architecture but also a network-related need to transport data appropriately so that load balancing can be achieved between the servers and distributed edge nodes. To address the latter issue, we developed an architecture in which processing systems executing multiple network functions are located between a vehicle and the application group. In the field trials, we verified whether this architecture is effective for solving this issue. To distinguish these processing systems from applications installed at terminals (edges), we refer to these processing systems as Network Edge (**Fig. 1**).

Network Edge acts as a gateway when vehicles upload data and a gateway when applications in the datacenter send notifications to vehicles. One of the goals with Network Edge is to hide the complexity of the network or infrastructure so that application developers can concentrate on developing application functions.

2. Target use cases

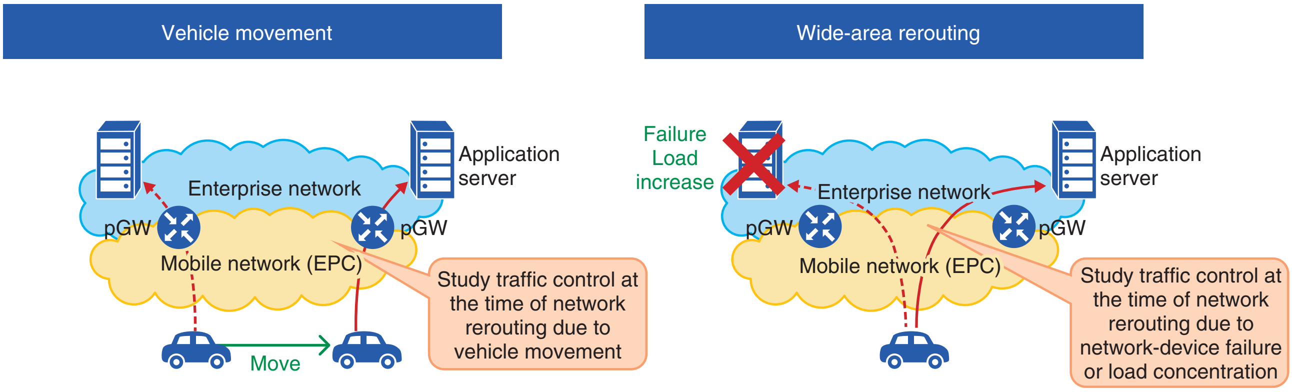
To avoid narrowing down target use cases too much, we selected two fundamental use cases to verify the performance of Network Edge: vehicle movement and wide-area rerouting (**Fig. 2**).

In the use case of vehicle movement, the application



BGP: Border Gateway Protocol
EPC: Evolved Packet Core
TLS: Transport Layer Security

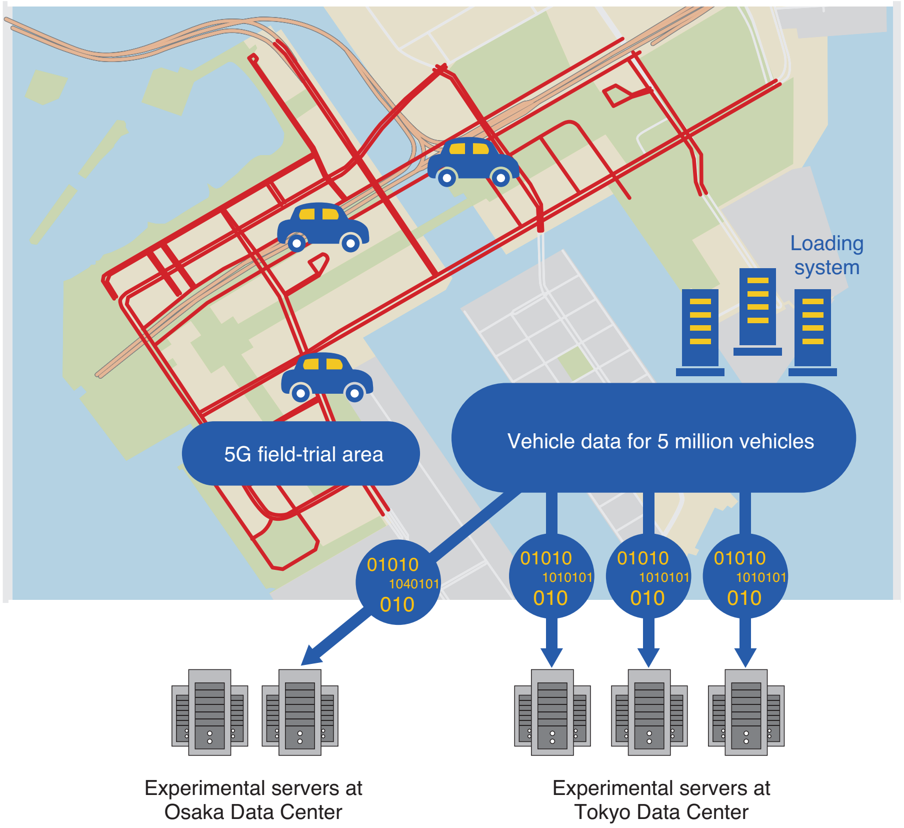
Fig. 1. Overview of Network Edge.


Vehicle movement
Enterprise network
Application server
pGW
pGW
Mobile network (EPC)
Move
Study traffic control at the time of network rerouting due to vehicle movement
Wide-area rerouting
Failure
Load increase
Enterprise network
Application server
pGW
pGW
Mobile network (EPC)
Study traffic control at the time of network rerouting due to network-device failure or load concentration

pGW: Packet Data Network Gateway

Fig. 2. Fundamental use cases.

The field trials using actual vehicles were conducted in the Odaiba area in Tokyo.



5G: the 5th-generation mobile communication system

Fig. 3. Overview of the environment for the field trials.

facility and server to which a vehicle is connected are switched as the vehicle moves. In principle, data are processed at the edge node closest to the vehicle concerned, and the results are sent from there to neighboring vehicles. This can be described as *local production for local consumption of data*. As the vehicle moves, it is necessary to switch not only the edge node to which the vehicle is connected but also the application executing the server. We verified whether this switching requirement can be supported by network technology.

In the use case of wide-area rerouting, the application facility to which a vehicle is connected is switched when a failure occurs at the facility or when it is necessary to execute load balancing. As edge nodes are distributed, a failure or overload of an edge node makes it necessary to transfer the processing load from the edge node concerned to neighboring edge nodes. We verified whether, in the proposed architecture, failures of edge nodes can be detected and transfer of processing loads between edge nodes can be supported.

3. Network Edge functions and verification items

Network Edge performs three main functions: control of routing to application servers at datacenters, load balancing based on metrics, and message queuing distributed over a wide area. If we put geographic distribution aside for the moment, data from a vehicle reaches an edge node via a wireless access network, such as a mobile network. The control functions mentioned above send data to applications on the edge node or, in some cases, to applications in the datacenter site.

In the field trials, we implemented an environment with multiple edge nodes using datacenters in Tokyo and Osaka to confirm the two use cases and verified the effectiveness of the three functions (**Fig. 3**). Since the use of Transport Layer Security (TLS) is assumed to encrypt communication with vehicles for increased security, terminating TLS at Network Edge may become a processing bottleneck. Therefore, we verified the performance of TLS termination. Details of these verifications are described below.

4. Routing control

We investigated two methods of controlling routing data from a vehicle to an application server: the domain name system (DNS) method and load balancer (LB) method. We confirmed the basic functions and performance of the two methods and investigated how they behave when a vehicle moves or wide-area rerouting takes place as a result of a facility failure.

(1) DNS method: Border Gateway Protocol (BGP) Anycast connects a vehicle to the nearest DNS server, and the DNS server selects the most appropriate datacenter and application server and notifies the vehicle of this information.

(2) LB method: BGP Anycast connects a vehicle to the nearest LB server, and the LB forwards requests from vehicles to the most appropriate server at the most appropriate datacenter.

With the LB method, vehicles always communicate with the application via the LB. This lowers performance but makes finer-granularity control possible. We compared the two methods in the early stage of the field trials and found that both were applicable to the target use cases. Therefore, from then on, the field trials were based on the LB method.

5. Load balancing using metrics

On the assumption that applications were located in multiple servers in multiple datacenters and processed in a distributed manner, we verified load balancing in which the datacenter and server to use were determined on the basis of metrics information, and in which the routing control described above was used.

We examined whether extreme performance degradation could be avoided by detecting server or datacenter failures, software process failures, or overloads from received information about response times from application servers, facility information, and metrics of the infrastructure resources, and by conducting load balancing on the basis of the detected situation. We also verified whether the load can be distributed over a wide area in specified proportions. For example, when the application load exceeds a threshold during an experiment, 70% of the load at Center 1 is processed at Center 1 and the remaining 30% is offloaded to Center 2.

6. Wide-area distributed message queuing

If the only response to a server or datacenter failure is to reroute data to a server in another datacenter, data transmission from vehicles would fail during the rerouting time. If this occurs, depending on the situation, it is necessary either to retransmit data or give up sending data. The purpose of the wide-area distributed message-queuing function is to prevent this from occurring.

As the term “message queue” indicates, the message-queuing function receives data only temporarily. A message queue resides at each edge node; therefore, message queues at different nodes work together to deliver data to the appropriate servers when offloading processing loads.

Since the edge node to which a vehicle is connected is switched as the vehicle moves, the datacenter that wants to send a notification to a vehicle needs to know the edge node to which the vehicle is currently connected. We aimed to satisfy this need using this queuing function. When the datacenter receives messages from applications, it retains them so that it can send them to the vehicle concerned even if the vehicle has moved to an area covered by a different edge node. In the field trials, we examined whether this occurs reliably.

If Apache Kafka or similar software is used to execute these processes, problems arising in that data are exchanged unnecessarily between datacenters; thus, it is impossible to manage notification delivery. To avoid this problem, we combined basic message-queuing software with a proxy program we developed. (We used NATS, which is open-source software, for the basic message-queuing function.) The properties required for data upload and data download (sending notifications) are different. Therefore, for the former, we used a mechanism with which the target data are first selected then sent. For the latter, we adopted a mechanism with which metadata are shared and there is a logical queue that spans different datacenters.

7. Verification of TLS performance

Vehicles communicate using TLS to ensure security. Applications communicate using HTTPS (Hypertext

Transfer Protocol Secure) or MQTTS (Message Queuing Telemetry Transport Secure) because they use HTTP or MQTT. Since we chose to use the LB method for routing control in this verification, the Layer 7 LB software needs to handle a large number of transactions. Normal tuning alone would result in a central processing unit bottleneck. Therefore, TLS processing was offloaded to another piece of hardware. We verified whether this would improve performance and resource-usage efficiency. Two types of hardware, TLS Accelerator and SmartNIC, both of which can be installed in a standard AMD64 server, were used for this verification. Although we have not yet been able to confirm the effectiveness of this method when the dominant traffic is data uploaded from vehicles, we will continue to verify this method since, for a connected vehicle platform designed for tens of millions of vehicles, it is important to minimize the number of required servers by using resources efficiently.

8. Future outlook

In these field trials, we implemented an architecture in which Network Edge, which perform multiple functions, is placed between vehicles and applications [1] and were able to confirm the effectiveness of this architecture.

On the basis of this architecture, we will further develop technologies and attempt to combine them with other technologies to develop a more advanced platform. For example, we are considering the application of artificial intelligence to metrics-based load balancing and coordination with priority control of application servers. We will continue our efforts to achieve a world in which intelligent network technology supports real-time communication of tens of millions of fast-moving vehicles and in which engineers can concentrate on developing applications without worrying about complex infrastructure conditions.

Reference

[1] Press release issued by NTT, “Tech Document on ICT Platform for Connected Vehicles Released by NTT and Toyota,” Nov. 29, 2021. https://group.ntt/en/topics/2021/11/29/techdoc_rel_ntt_toyota.html

Ryosuke Noji

Senior Manager, Smart Mobility, Smart World Business, Business Solution Division, NTT Communications Corporation.

He started his career as a solution engineer at NTT Communications in 2007. After several projects, he joined the research and development department. As a data platform architect, he designed and developed data platforms for the company's IPTV Service and the Data Science Lab, which is a data science Center of Excellence at NTT Communications, and analyzed and developed algorithms for several services. He has been engaged in solution development for the automotive industry since 2018 and has led a connected-vehicle project for developing network and computing services or solutions, including edge computing, for connected-vehicle big data.

Takayuki Kamei

Engineer, Dept. Technology, Innovation Center, NTT Communications Corporation.

He started his career as a network engineer at NTT Communications in 2013. He has continued to research and develop network functions virtualization, software-defined wide area network, application-layer overlay networks, and content delivery network. He has been engaged in the development of edge computing and traffic optimization since 2019 and led the architecture design of distributed platforms over a WAN including connected-vehicle use cases.

Kaito Funabiki

Engineer, Smart Mobility, Smart World Business, Business Solution Division, NTT Communications Corporation.

He started his career as a network engineer at NTT Communications in 2017. After working on the operation and development of an internet exchange service, he has been engaged in a connected-vehicle project for developing network and computing services or solutions, including edge computing, for connected-vehicle big data since 2020.

Daisuke Murata

Engineer, Innovation Center, NTT Communications Corporation.

He received an M.E. in information science from Nara Institute of Science and Technology in 2014. In 2018, he joined NTT Communications and has been engaged in the development/research of computer systems and computer networking technologies, especially in the area of TLS acceleration since 2020.

Yuken Kanemaru

Engineer, Dept. Technology, Innovation Center, NTT Communications Corporation.

He started his career as a network engineer at NTT Communications in 2019. He has continued to research and develop application-layer overlay networks and content delivery network. He has been engaged in the development of edge computing, traffic optimization, and distributed platforms over a WAN including connected-vehicle use cases.