Technology Reports　Cloud　MEC　Application Design Patterns

Application Design Patterns in MEC

Innovation Management Department　Yoshikazu Akinaga

5G network construction has progressed in recent years, and NTT DOCOMO launched 5G services in March 2020. 5G has three major characteristics - high-speed/capacity, low latency, and simultaneous connection to many devices. These high-speed/capacity and low latency characteristics enable services that have been difficult to achieve with conventional mobile technology, and thus hold great promise. For this reason, MEC is attracting attention as an approach to building systems that take advantage of high speed/capacity and low latency. Also, since June 2020, NTT DOCOMO has also been offering cloud server called "docomo Open Innovation Cloud" as well as a service to connect it directly into the mobile network called "Cloud Direct". To take advantage of these services, we propose application design patterns in MEC, which are architectural templates used for functions required to build applications when considering architecture. In this article, we describe some typical patterns and their effects.

1. Introduction

In recent years, construction of networks for 5th Generation mobile communication systems (5G) has been progressing. NTT DOCOMO launched its 5G services in March 2020. 5G has three major characteristics - high-speed and high-capacity with enhanced Mobile Broad Band (eMBB), Ultra-Reliable and Low Latency Communications (URLLC), and simultaneous connection of multiple devices with massive Machine Type Communications (mMTC), which are expected to have an impact on various applications, solutions and industries. With these high-speed, high-capacity and low-latency characteristics, services that were previously difficult to achieve with mobile devices are now expected to become a reality.

For this reason, the positioning of clouds on networks, called Multi-access Edge Computing (MEC), is attracting attention as an approach to building systems that can take advantage of high-speed, high-capacity, and low latency. NTT DOCOMO has also been offering cloud servers called "docomo Open Innovation Cloud" as well as a service to connect it directly into the mobile network called "Cloud Direct" since June 2020. These services hold promise for the provision of a variety of applications and solutions that take advantage of 5G characteristics.

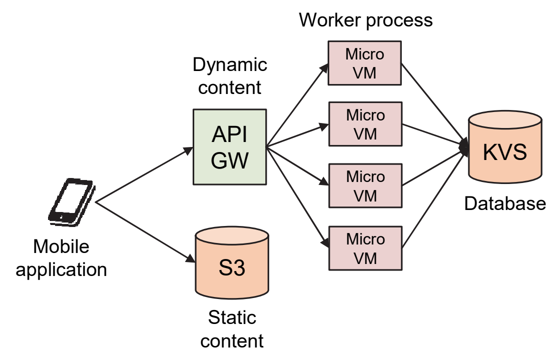
To take advantage of these services, we propose application design patterns in MEC. Referring to these patterns will enable developers to design applications with optimal placement of functions in the cloud and MEC. In this article, after describing application design patterns and MEC, we describe some typical patterns and their effects.

2. What is an Application Design Pattern?

An application design pattern is an architectural template used for functions required for building applications when considering architecture, and is a collection of generalized ideas based on typical use cases available for use as application architecture.

Similar to general design patterns, application design patterns are highly reusable because they are discussed as parts rather than as whole applications, which makes it easier for application architects to immediately consider and incorporate new functions.

As an example, **Figure 1** shows an application design pattern on a public cloud[*1] for building Web



MicroVM: A very small virtual machine that can be started up quickly.

Figure 1 Example application design pattern (serverless)

*1 Public cloud: A cloud computing service that can be used by anyone via the Internet.

services. Fig. 1 shows an application design pattern for a typical serverless application commonly used in public clouds. It is easy to see at a glance that dynamic and static contents are stored separately, that worker processes*2 are finely arranged for scaling out, and that a Key Value Store (KVS)*3 is used to enable access to avoid database bottlenecks. Designers can implement scalable mobile applications by following these design patterns to design applications. These organized patterns of design know-how are application design patterns.

3. What is MEC?

Although there is growing momentum to take advantage of low latency on 5G networks, low latency cannot be achieved just by making radio sections 5G. To achieve low latency as a system, the entire network must be considered, and its sending and receiving distances and the number of devices using it must be reduced. Therefore, NTT DOCOMO is attempting to solve this problem by having computing resources within the 5G network. This is MEC. MEC is achieved by placing the computing resources (virtual machines) as close as possible to radio sections (**Figure 2**). The closer locations are to radios (closer to base stations), the shorter the latency can be, although this would dramatically increase the number of base stations and inevitably increase the number of required virtual machines which would be economically impractical. In contrast, it's possible to attempt aggregation if locations are close to the Internet, but because this increases latency, it is important to allocate resources appropriately based on the balance between supply and demand.

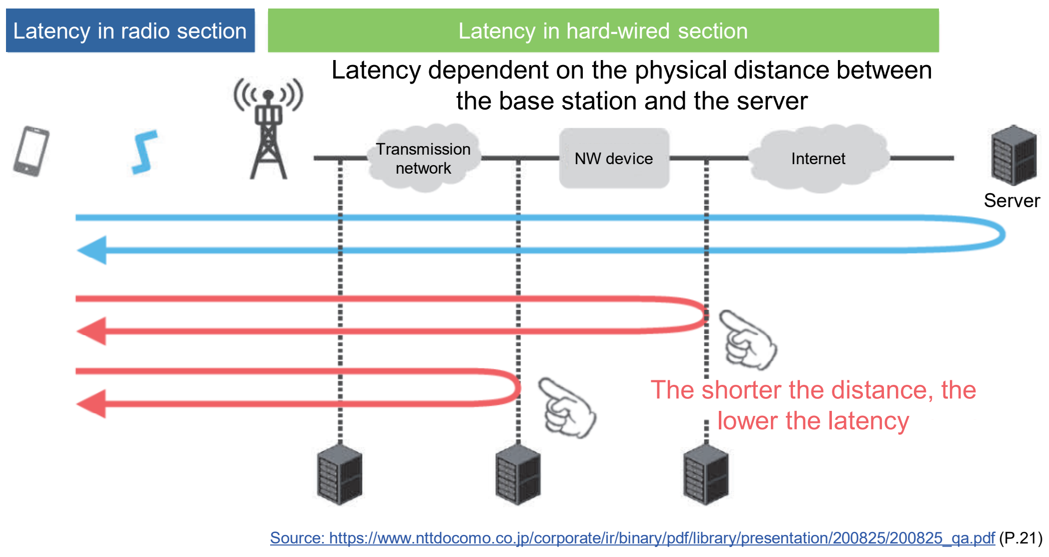

Latency in radio section
Latency in hard-wired section
Latency dependent on the physical distance between the base station and the server
Transmission network
NW device
Internet
Server
The shorter the distance, the lower the latency
Source: https://www.nttdocomo.co.jp/corporate/ir/binary/pdf/library/presentation/200825/200825_qa.pdf (P.21)

Figure 2 Location of MEC on the network

*2 Worker process: A single program that is launched with a certain role and terminates when the role is completed.

*3 KVS: A high-speed database that specializes in simple management of data with sets consisting of a key and a value. These are often used to avoid processing bottlenecks because they support distributed processing on multiple servers and can handle huge amounts of input and output simultaneously.

3.1 Four Benefits of MEC

From the user perspective, MEC promises the following four main characteristics.

(1) Traffic optimization, terminal-to-terminal communications

Peer to Peer (P2P) communications between terminals can be done with very low latency by accessing or passing traffic through a server that is geographically close to terminals and the server can send large amounts of data. This holds promise for games and other applications with strict response time requirements.

(2) Low latency, low jitter

Accessing servers that are geographically close makes it possible to connect with low latency and take advantage of the characteristics of 5G. Also, unlike the Internet, provision over a closed network reduces the causes of jitter and hence holds promise for streaming and other applications that are vulnerable to such fluctuations.

(3) Secure and private network

Secure connections can be provided because there is never any connection to an external network such as the Internet. Private connections with SIM authentication will be possible and enable networks that handle highly confidential information.

(4) Leveraging edge computing resources and complementary terminal functions

Edge computing[*4] enables processing of large loads such as advanced AI processing and the generation of high-definition 3D images, which cannot be processed by individual terminals, and is anticipated for use in games, 3D Computer Aided Design (CAD) and XR[*5] among others.

Incorporating these MEC characteristics into specific applications is discussed with application design patterns.

3.2 Preconditions for Application Design Patterns in MEC

Because of the nature of servers distributed across regions, MEC differs from systems operating in centralized data centers in terms of fault tolerance and robustness. Therefore, a method called "design for failure" is incorporated into the applications on MEC. This is a method of building applications assuming they could fail in public cloud computing. For example, data durability in MEC is not guaranteed, because it requires a significant amount of redundancy, which is difficult to obtain in a distributed system. Therefore, it is preferable to implement data durability in a public cloud or other storage service rather than MEC.

Other preconditions in MEC include the following.

- Robustness and availability of MEC are both low.
- MEC provides Infrastructure as a Service (IaaS)[*6].
- MEC does not provide Platform as a Service (PaaS)[*7] which has high durability data storage
- On the other hand, MEC could provide functions to improve portability, such as container[*8] management services.

---

*4 Edge computing: Technology that distributes edge servers closer to the users to improve response and reduce latency.

*5 XR: A general term for technologies such as VR, AR, and MR that provide new experiences through the fusion of virtual space and real space.

*6 IaaS: A online service in which servers and networks are provided virtually. The user sets up an OS and application software on the online server or network and uses it.

*7 PaaS: A online service that provides a platform including an OS and middleware to run applications on the service. The user creates and uses application software on the platform.

- MEC also provides Domain Name System (DNS)[*9] services.

4. Application Design Patterns in MEC

The following discusses some specific application design patterns and their use cases.

4.1 A Pattern for High Speed, Large Upload

A function that applications can take advantage of the high speed and high capacity of 5G is uploading, which has traditionally been a weakness of mobile networks. Significantly higher upload speeds make it possible to consider utilization, although due to issues such as increased latency, jitter, and reduction in throughput due to latency in Transmission Control Protocol (TCP)[*10] ACKnowledgement (ACK)[*11] responses due to such increased latency and jitter, it is not possible to obtain sufficient speed in communications to the Internet. However, MEC makes high-speed, large uploading possible.

1) Challenges

To speed up large uploads using the high-speed 5G network, and avoid the dependency of the upload process on the latency of TCP ACK packets, jitter and the influence of servers on the internet, and make the upload process stable. There is also the challenge of accepting a large number of large uploads at the same time.

2) Design Pattern

To take full advantage of 5G upload capabilities, uploads should be terminated at an MEC local server and temporarily stored. Then, stored contents are queued[*12], retrieved by a worker process from a separate temporary upload destination, and perpetuated in public cloud storage. It is also possible to run separate processes (e.g., image/video processing by AI) while data is in temporary storage. The DNS lookup service helps search for nearby servers to enable connection to the closest upload server. Also, since temporary storage is effective in load balancing, it can be used to handle mass uploads from the same location (e.g., simultaneous uploads from spectators in a stadium) when there is not enough bandwidth on the Internet. Since only the upload process can be implemented separately, it is easy to integrate into existing systems. In addition, a Load Balancer (LB)[*13] should be included in MEC to duplicate the upload server and ensure availability (**Figure 3**).

4.2 A Pattern for Ultra-low Latency Messaging and Status Management

The low latency of 5G enables multi-player games to be synchronized across multiple devices, simultaneous XR experiences and message exchange, etc. KVS Pub/Sub functions[*14] in MEC that achieve the synchronization provide very versatile messaging services.

1) Challenges

To achieve low-latency synchronization by putting ultra-low latency KVS and its Pub/Sub functions in MEC. To achieve state synchronization between terminals in competitive and open-world games and state synchronization of IoT devices with ultra-low latency. To also provide services with synchronization latency below a certain level to

[*8] **Container:** A type of computer virtualization technology in which a dedicated area (the container) is created on a host OS, and the necessary application software is run in the container.

[*9] **DNS:** A system for mapping host names and IP addresses on an IP network.

[*10] **TCP:** A higher-level Internet protocol that is used as a standard. It plays a complementary role to IP by confirming the connection destination and data arrival, controlling the flow of data, and detecting duplicate or missing data to realize highly reliable communications.

[*11] **ACK:** A reception confirmation signal to notify the transmitting node that the receiving node has received (decoded) the data correctly.

[*12] **Queuing:** The creation of a queue and temporarily storing the order of processing and the content to be processed in it.

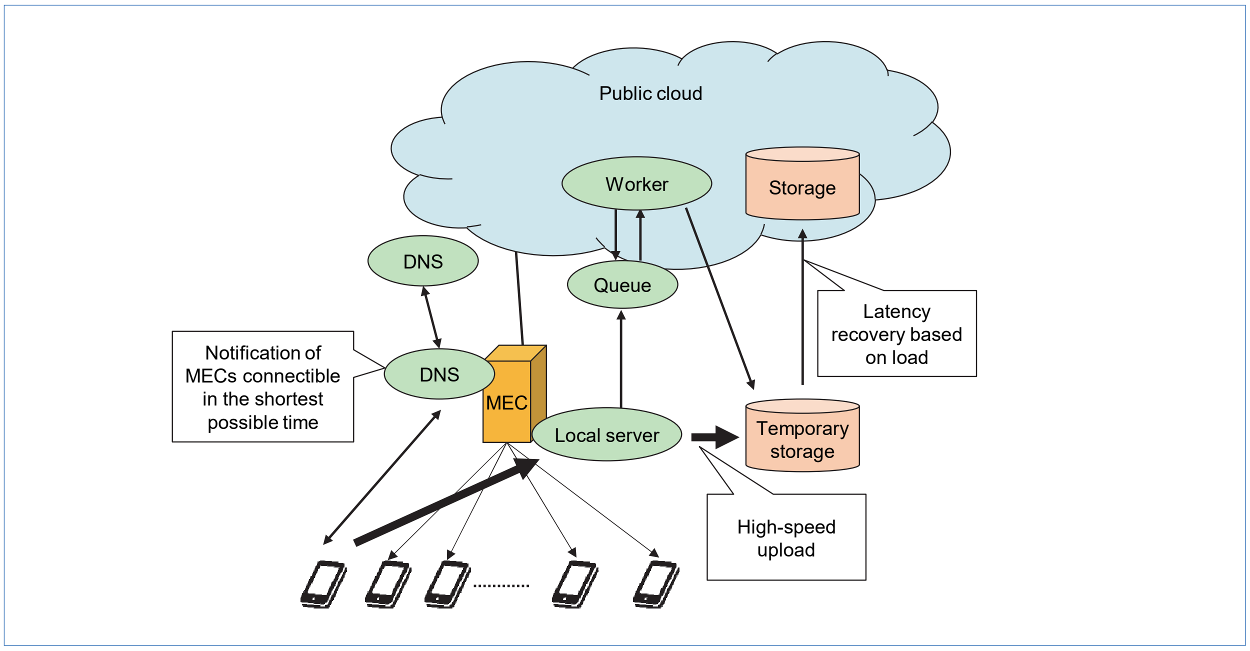

Public cloud
Worker
Storage
DNS
Queue
Latency recovery based on load
Notification of MECs connectible in the shortest possible time
DNS
MEC
Local server
Temporary storage
High-speed upload

Figure 3 High-speed large upload design pattern

eliminate impaired gameplay, such as warping[*15].

2) Design Pattern

An in-memory DB[*16] such as Redis should be installed in MEC to perform messaging, temporary status management and synchronization of application at ultra-high speed between terminals. These in-memory DBs can be distributed and reference information in the order of nanoseconds. Here as well, a neighboring in-memory DB lookup service in DNS can be used to connect to neighboring in-memory DBs. Using the Pub/Sub function, messages can be exchanged between neighboring terminals with ultra-low latency, and similarly, KVS can be used for application state management, etc. Using these in-memory DBs makes it possible to aggregate game scores for the regions in which players are playing at very high speed and aggregate game rankings (real-time leaderboards[*17]). These can also be used as service buses[*18] for coordination of multiple systems, state synchronization between players in network games, and synchronization of remote robots (**Figure 4**).

3) Points to Note

In-memory DBs have cases where authentication does not exist, so message encryption and network isolation, etc. are necessary when they are implemented.

4.3 A Pattern for Web Service Caching

One major method of real-time updating on the Web is to use Web sockets[*19]. When implementing such services, major issues include latency and load balancing of traffic. MEC is also effective in solving these issues.

1) Challenges

To update dynamic content with as low latency

*13 LB: A device that balances the load of communication traffic.

*14 Pub/Sub function: A function that allows asynchronous message exchange between Publish and Subscribe functions. Since the Subscribe side distributes to everyone what is sent by the Publish side, this function is suitable for distributing 1: *N* messages.

*15 Warp: A phenomenon in which a player suddenly disappears and reappears from a different location due to mis-synchronization of location information in a game in which two or more players are participating. This phenomenon is often detrimental to gameplay, therefore game providers try to avoid it if possible.

*16 In-memory DB: A database that enables fast response to information acquisition by expanding and holding data in memory for processing.

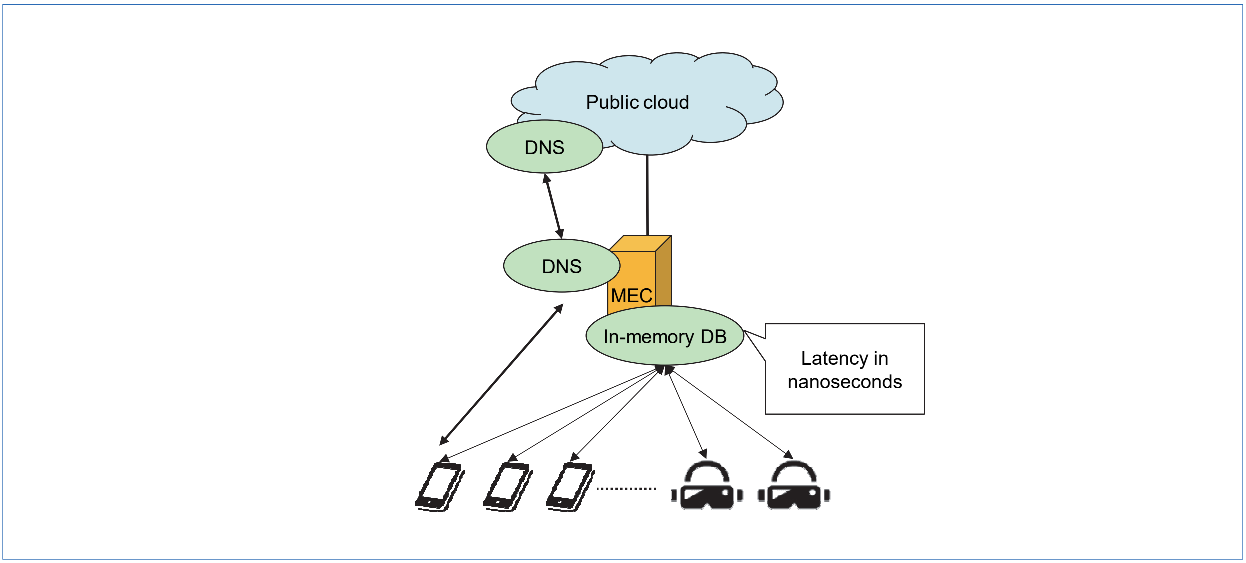


Figure 4 Ultra-low latency messaging and status management design pattern

as possible (e.g., Web sockets) and to achieve highly responsive services. There are also issues such as reducing the load of reading static content on the Web.

2) Design Pattern

A caching mechanism for Web services in MEC enables the presentation and application layers of the three-tiered Web structure[*20] to be placed in MEC to achieve higher speeds. The database layer can also be sped up by having read replicas[*21] and a cache mechanism in MEC. It is important to note that if the database layer is not deployed in the public cloud, data persistence becomes difficult, and consistency cannot be maintained over a wide area. Therefore, the cache is placed near the application layer to achieve higher speed.

On the database layer, the read replicas of MySQL[*22] and multi-master[*23] are also effective for load balancing. However, in anticipation of low latency, care must be taken to balance the latency of the database layer with the latency of the network, because the latency of the database layer is significantly greater than the latency of the network (**Figure 5**).

3) Points to Note

To automatically deploy the application layer close to the customer, schemes for containerization, etc. should be implemented in consideration of portability. Also, in-memory caching (KVS) is used for Web caching to enable ultra-fast response times. The data persistence issue needs to be considered for writing in this case.

This pattern is easy to implement, but care must be taken because management costs may increase as the number of MEC servers increases. In addition, using container management solutions and other methods to package implementation while deploying the minimum number of applications required is an important point in keeping the system efficient and cost effective.

---

*17 Leaderboard: A board that lists the top rankings and one' s order in a game, etc.

*18 Service bus: A mechanism for exchanging messages to mutually link statuses or pass on processing to the next system when multiple systems are linked.

*19 Web Sockets: A technical standard for exchanging two-way messages over the Web, defined as Request For Comments (RFC) 6455.

*20 Web three-layer structure: A method of dividing the components of a Web system into three layers (presentation layer, application layer, and data layer) and designing them as independent modules.

*21 Read replica: A read-only copy of a database. Such a copy of a main database is always kept to reduce the load of reading and searching.

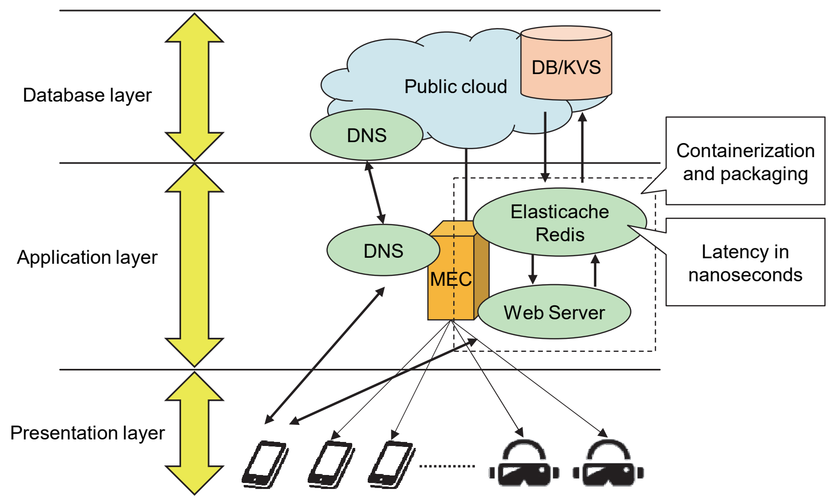


Figure 5　Web service cache design pattern

4.4　A Pattern for Lightweight Protocols of IoT Devices

The problem of security in IoT devices is solved by blocking access from the Internet and keeping low-load and low-security protocols within the carrier network. This is highly convenient because it enables isolation from the Internet and handling of processing from many devices with low latency.

1)　Challenges

Since some IoT devices are too small to implement Secure Sockets Layer (SSL)[*24] and other similar protocols, there is the challenge of adopting lightweight protocols while maintaining security. There are some examples of implementing Message Queuing Telemetry Transport (MQTT)[*25] with Transport Layer Security (TLS)[*26]. However, because communications encryption is too heavy for such devices, the challenge is to use lightweight protocols with less authentication and encryption to reduce device battery consumption, yet maintain security.

2)　Design Pattern

Lightweight protocols for IoT devices, such as MQTT, can be safely terminated within the mobile network and processed and encrypted when they leave MEC to enable secure management of IoT devices. This prevents attacks from the Internet and enables secure device management systems to be built (**Figure 6**).

4.5　Other Design Patterns

Other design patterns that have been proposed are shown in **Table 1**. Going forward, we would like to increase the number of design patterns as specific methods of effectively using MEC.

*22　MySQL: One of the most popular open-source Relational Database Management Systems (RDBMS).

*23　Multi-master: A method of improving the reliability and performance of a database. Refers to a system that can have multiple master servers. Even with multiple connections to such servers, the data is the same, and there are no restrictions on the functions that can be used.

*24　SSL: A protocol for encrypting communications and detecting data tampering between applications on a network, primarily between WWW browsers and WWW servers.

*25　MQTT: A lightweight message queue protocol of the Pub/Sub type, used to exchange messages between various devices and servers on the IoT.

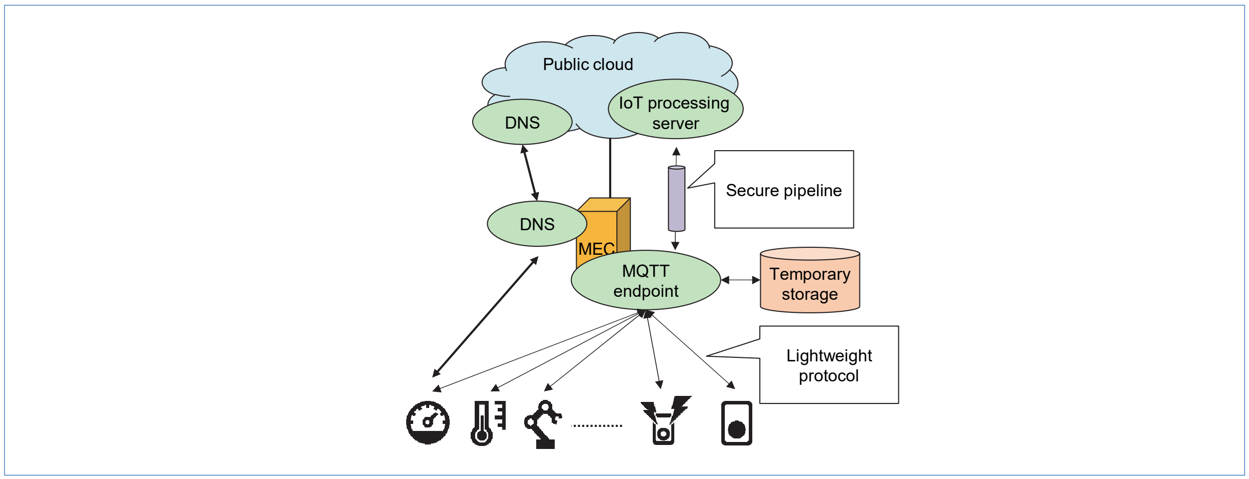


Figure 6　Implementation pattern of lightweight protocols for IoT devices

Table 1　Other proposed application design patterns in MEC

Pattern name	Challenge	Design pattern
CDN	To reduce the load on the origin server (delivery server) for video and streaming services. To prevent quality degradation due to network jitter to the origin server. When services are provided at ultra-high speed and ultra-large capacity, the higher the network level the more the load concentrates, and quality tends to deteriorate. The challenge is to provide as much video as possible in real time.	The closest server can be determined by having the DNS return the closest location, just as in an ordinary CDN system. It can retrieve the cache of the closest location. Each server retrieves the latest information from the origin server (e.g., Web server in the public cloud, streaming server) when there are changes. The same logic can be applied when using HLS, etc. for streaming, which is highly versatile.
P2P traffic optimization by reflection	To optimize P2P traffic and provide stable and high-quality services through loopback communications on the local edge network. For example, to optimize video calls by making them limited locally and looping them back.	The shortest possible routing within the network of the loop back traffic over the 5G/4G local network enables terminal-to-terminal communications with low latency. Placing a signaling server in MEC will make it possible to implement WebRTC and VoIP (SIP), etc. If the signaling server is deployed in a wide area, it does not have to be in MEC.
Local secure network	To realize a local network between terminals and build a secure network without using VPNs (e.g., file servers).	Placing VPN servers in MEC enables creation of any private network. In addition, because this can achieve high speed and large capacity, access to file servers, etc. can be built safely and easily.
Security using geographic constraints	To prevent access from outside of certain areas and provide services more securely.	Placing a file server in MEC only in a specific region enables the system to access servers only from systems in the specific region. The regional MEC server is identified with DNS and that server is accessed. Access to MEC is with information only accessible in the region or using an authentication method. Storage or DB is deployed on a public cloud, and its robustness and availability are ensured. To ensure availability, LBs should be deployed in MEC to create redundant systems.

CDN: Content Delivery Network	VoIP: Voice over Internet Protocol
HLS: HTTP Live Streaming	VPN: Virtual Private Network
SIP: Session Initiation Protocol	WebRTC: Web Real-Time Communication

*26 TLS: A protocol that regulates SSL as an Internet standard technology and ensures its extended security. It has extended cryptographic algorithms and error message regulations compared to SSL.

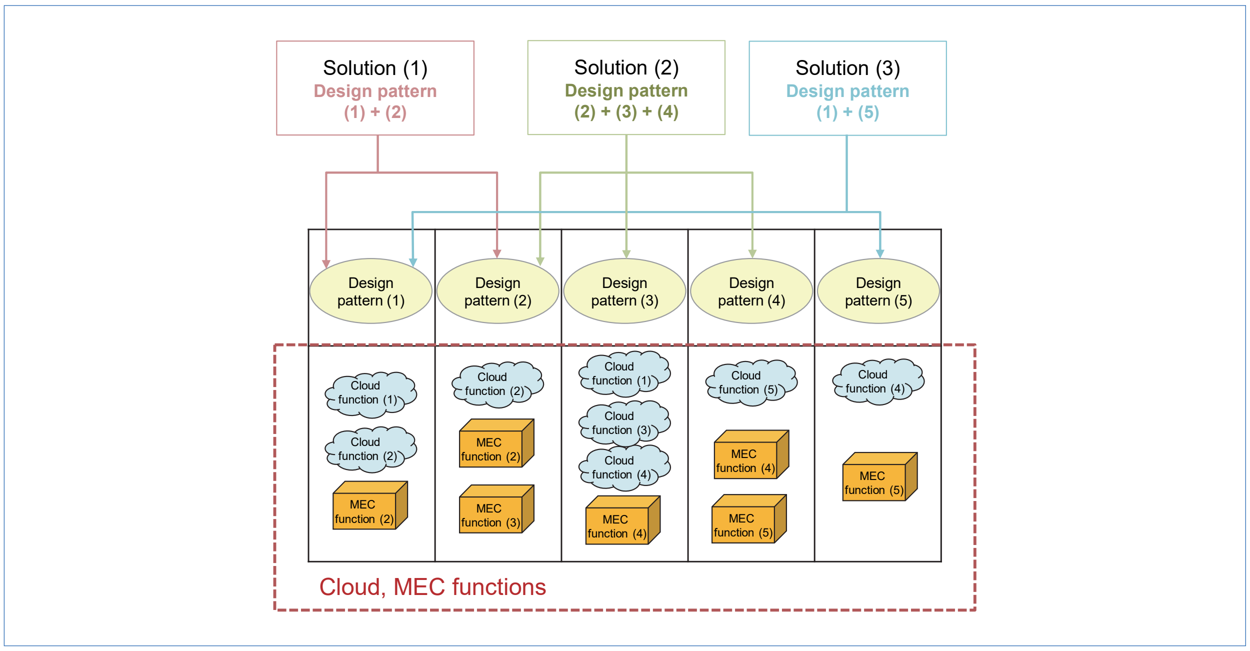


Figure 7　Relationship of application design patterns and solutions

5. Relationship between Application Design Patterns in MEC and Solutions

Application design patterns in MEC only indicate useful usage patterns (best practices) to achieve specific functions to use MEC effectively. These design patterns are combined to build an application or solution. The relationship between patterns and solutions is shown in **Figure 7**.

6. Conclusion

In this article, we explained the importance of application design patterns to make MEC more useful. For effective use of MEC, we believe that the accumulation of these best practices and the reuse of this knowledge will lead to the creation of new value-added services in the 5G era. Going forward, as policy for the provision of functions on MEC, we will strive to increase the variety of design patterns so that they can be applied to a wider range of applications. We also want to study and provide PaaS functions (functions that make it easy to use functions to achieve design patterns without having to implement them yourself) on MEC to make it easier to use common design patterns.