International Telecommunication Union
Standardization Sector

Recommendation

ITU-T Q.5007 (12/2023)

SERIES Q: Switching and signalling, and associated measurements and tests

Signalling requirements and protocols for IMT-2020 – Signalling requirements and architecture of IMT-2020

Signalling architecture for microservices based intelligent edge computing

ITU-T Q-SERIES RECOMMENDATIONS

Switching and signalling, and associated measurements and tests

SIGNALLING IN THE INTERNATIONAL MANUAL SERVICE	Q.1-Q.3
INTERNATIONAL AUTOMATIC AND SEMI-AUTOMATIC WORKING	Q.4-Q.59
FUNCTIONS AND INFORMATION FLOWS FOR SERVICES IN THE ISDN	Q.60-Q.99
CLAUSES APPLICABLE TO ITU-T STANDARD SYSTEMS	Q.100-Q.119
SPECIFICATIONS OF SIGNALLING SYSTEMS NO. 4, 5, 6, R1 AND R2	Q.120-Q.499
DIGITAL EXCHANGES	Q.500-Q.599
INTERWORKING OF SIGNALLING SYSTEMS	Q.600-Q.699
SPECIFICATIONS OF SIGNALLING SYSTEM NO. 7	Q.700-Q.799
Q3 INTERFACE	Q.800-Q.849
DIGITAL SUBSCRIBER SIGNALLING SYSTEM NO. 1	Q.850-Q.999
PUBLIC LAND MOBILE NETWORK	Q.1000-Q.1099
INTERWORKING WITH SATELLITE MOBILE SYSTEMS	Q.1100-Q.1199
INTELLIGENT NETWORK	Q.1200-Q.1699
SIGNALLING REQUIREMENTS AND PROTOCOLS FOR IMT-2000	Q.1700-Q.1799
SPECIFICATIONS OF SIGNALLING RELATED TO BEARER INDEPENDENT CALL CONTROL (BICC)	Q.1900-Q.1999
BROADBAND ISDN	Q.2000-Q.2999
SIGNALLING REQUIREMENTS AND PROTOCOLS FOR THE NGN	Q.3000-Q.3709
SIGNALLING REQUIREMENTS AND PROTOCOLS FOR SDN	Q.3710-Q.3899
TESTING SPECIFICATIONS	Q.3900-Q.4099
PROTOCOLS AND SIGNALLING FOR PEER-TO-PEER COMMUNICATIONS	Q.4100-Q.4139
PROTOCOLS AND SIGNALLING FOR COMPUTING POWER NETWORKS	Q.4140-Q.4159
PROTOCOLS AND SIGNALLING FOR QUANTUM KEY DISTRIBUTION NETWORKS	Q.4160-Q.4179
SIGNALLING REQUIREMENTS AND PROTOCOLS FOR IMT-2020	Q.5000-Q.5049
Signalling requirements and architecture of IMT-2020	**Q.5000-Q.5019**
Protocols for IMT-2020	Q.5020-Q.5049
COMBATING COUNTERFEITING AND STOLEN ICT DEVICES	Q.5050-Q.5069

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T Q.5007

Signalling architecture for microservices based intelligent edge computing

Summary

Recommendation ITU-T Q.5007 specifies signalling architecture for microservices based intelligent edge computing (IEC) to provide intelligence to the edge network for efficient data processing within the network.

To develop the IEC architecture, there are a couple of software-oriented architectural ways for building a flexible protocol architecture achieved by deploying and operating the architecture. One approach is a unified software-oriented architecture that composes logically modular functions in a tightly coupled way similar to a monolithic architecture. Another approach is a microservice architecture, which loosely composes logically or physically separated processing functions as individual microservices.

Since the IEC has been developed on different hardware specifications and various functionalities desired by each business demands, it has been standardized based on the microservices and used as a reference standard for implementation. Adopting a microservices based IEC architecture facilitates continuous development and operation through the updating of individual microservices. This Recommendation specifies the signalling architecture, protocol interfaces, and protocol procedures for microservices based intelligent edge computing.

History *

Edition	Recommendation	Approval	Study Group	Unique ID
1.0	ITU-T Q.5007	2023-12-14	11	11.1002/1000/15729

Keywords

Intelligent edge computing, microservices, signalling architecture.

* To access the Recommendation, type the URL https://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents/software copyrights, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the appropriate ITU-T databases available via the ITU-T website at http://www.itu.int/ITU-T/ipr/.

Table of Contents

Recommendation ITU-T Q.5007

Signalling architecture for microservices based intelligent edge computing

1 Scope

This Recommendation specifies signalling architecture for microservices based intelligent edge computing to provide intelligence to the edge network for efficient data processing within the network. It describes the following details:

– Overview of microservices based intelligent edge computing;

– Signalling architecture;

– Protocol interfaces;

– Protocol procedures; and

– Security considerations.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T J.1301] Recommendation ITU-T J.1301 (2021), *Specification of cloud-based converged media service to support Internet protocol and broadcast cable television – Requirements*.

[ITU-T Q.5001] Recommendation ITU-T Q.5001 (2018), *Signalling requirements and architecture of intelligent edge computing*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 cloud computing [b-ITU-T Y.3500]: Paradigm for enabling network access to a scalable and elastic pool of shareable physical or virtual resources with self-service provisioning and administration on-demand.

3.1.2 intelligent edge computing [ITU-T Q.5001]: Intelligent edge computing is a network architecture concept that enables edge networking and data processing capabilities for edge analytics by applying artificial intelligence technologies.

3.1.3 microservice [ITU-T J.1301]: An architectural and organizational approach to software development where software is composed of small independent microservices that communicate over well-defined application programming interfaces. Microservice architecture makes applications easier to scale and faster to develop, enabling innovation and accelerating time-to-market for new features.

3.2 Terms defined in this Recommendation

None.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

5G	Fifth-Generation Mobile Network
AI	Artificial Intelligence
API	Application Programmable Interface
AR	Augment Reality
CRUD	Create, Read, Update, and Delete
DL	Deep Learning
DOS	Denial Of Service
DQ	Data Quality
EGE	Edge Gateway Entity
EME	Edge Identity Management Entity
ENE	Edge Networking Entity
ETL	Extract, Transform, and Load
ETSI	European Telecommunications Standards Institute
HTTP	Hypertext Transfer Protocol
ICE	Intelligent Computing Entity
IEC	Intelligent Edge Computing
IM	Ingress Management
IoT	Internet of Things
IT	Information Technology
JSON	JavaScript Object Notation
MDP	Markov Decision Process
MEC	Multi-access Edge Computing
ML	Machine Learning
MME	Monitoring and Management Entity
NFT	Non-Fungible Token
NR	New Radio
OJ	Online Judge
OME	Orchestration Management Entity
PLC	Programming Logic Controller
QoS	Quality of Service
RL	Reinforcement Learning
SM	Service Management
TE	Terminal Entity
URI	Uniform Resource Identifier
URLLC	Ultra-Reliable and Low Latency Communications

V2X Vehicle-to-Everything

VR Virtual Reality

XML Extensible Markup Language

5 Conventions

In this Recommendation:

– The keywords "is required to" indicate a requirement which must be strictly followed and from which no deviation is permitted, if conformance to this Recommendation is to be claimed.

– The keywords "is recommended" indicate a requirement which is recommended but which is not absolutely required. Thus, this requirement need not be present to claim conformance.

– The keywords "can optionally" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option, and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with this Recommendation.

6 Overview

Intelligent edge computing (IEC) is a network architecture concept that enables edge networking and data processing capabilities for edge analytics by applying artificial intelligence (AI) technology. Regarding the edge networking capability, IEC allows edge networking to support mobility, real-time and reliable communication, scalable networks, constrained network environment, and ease of operation and deployment. Additionally, for intelligent data processing, IEC is designed to operate tasks like data collection, dynamic storage, real-time trust data process, online learning, update models and so on.

Figure 6-1 briefly redraws the overall signalling framework of intelligent edge computing (IEC) defined in [ITU-T Q.5001], which consists of many functional blocks deployed in each IEC entity, such as edge networking entity (ENE), an intelligent computing entity (ICE), edge gateway entity (EGE) and edge identity management entity (EME). The IEC could be deployed in a location between terminal entity (TE), and big data analytics at cloud computing or other IECs. And all these functions could be classified into three functional blocks such as related to data processing, identity management, and edge networking for supporting intelligent edge networking and data processing capabilities, so that each entity could be deployed by composing functional blocks in various ways.

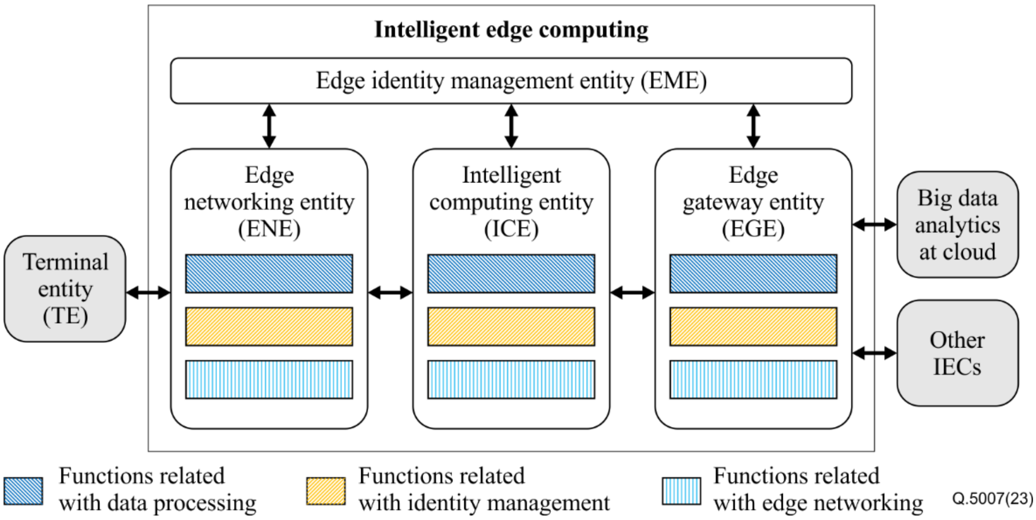

Figure 6-1 – Overview of intelligent edge computing signalling architecture [ITU-T Q.5001]

To develop the IEC architecture, there are a couple of software-oriented architectural ways for building flexible protocol architecture achieved by deploying and operating the architecture. For instance, one approach is a unified software-oriented architecture that composes logically modular functions in a tightly coupled way similar to a monolithic architecture. Another approach is a microservice architecture, which loosely composes logically or physically separated processing functions as individual microservices.

In the monolithic architecture, which is generally considered a traditional approach, there is a significant drawback when it comes to the deployment step. Once the architecture has been deployed completely, if it is needed to change or upgrade some part of the functions, the architecture should be required to be re-deployed in the entire architecture, even if only one function needs updating. Therefore, the IEC architecture should be desirable to implement and deploy using microservice-based architecture.

Microservices are a software development technique, which is a variant of the service-oriented architectural style that is composed of small independent microservices as a collection of loosely coupled services. Microservices architecture makes applications easier to scale and faster to develop, enabling innovation and accelerating time-to-market for new features through continuous refactoring [ITU-T J.1301].

Since IEC has developed to accommodate a variety of hardware specifications and diverse functionalities desired by different businesses, it has been standardized based on microservices architecture, and used as a reference standard for implementation. As a result of adopting a microservice-based approach, IEC architecture allows for continuous development and operation through the incremental updating of individual microservices.

6.1 Microservices based IEC framework

The framework specifies the microservice-based IEC architecture, which is based on the IEC high level architecture defined in [ITU-T Q.5001]. The framework extends the IEC architecture to build an applicable IEC that can be deployed in real-world applications such as smart factory, smart city, smart home, etc. To achieve this, new architectural entities have been defined including the microservice pool, microservice monitoring and management, and orchestration management. These entities are organized into three vertical layers, such as composition layer, orchestration layer, and microservice management layer, as depicted in Figure 6-2.

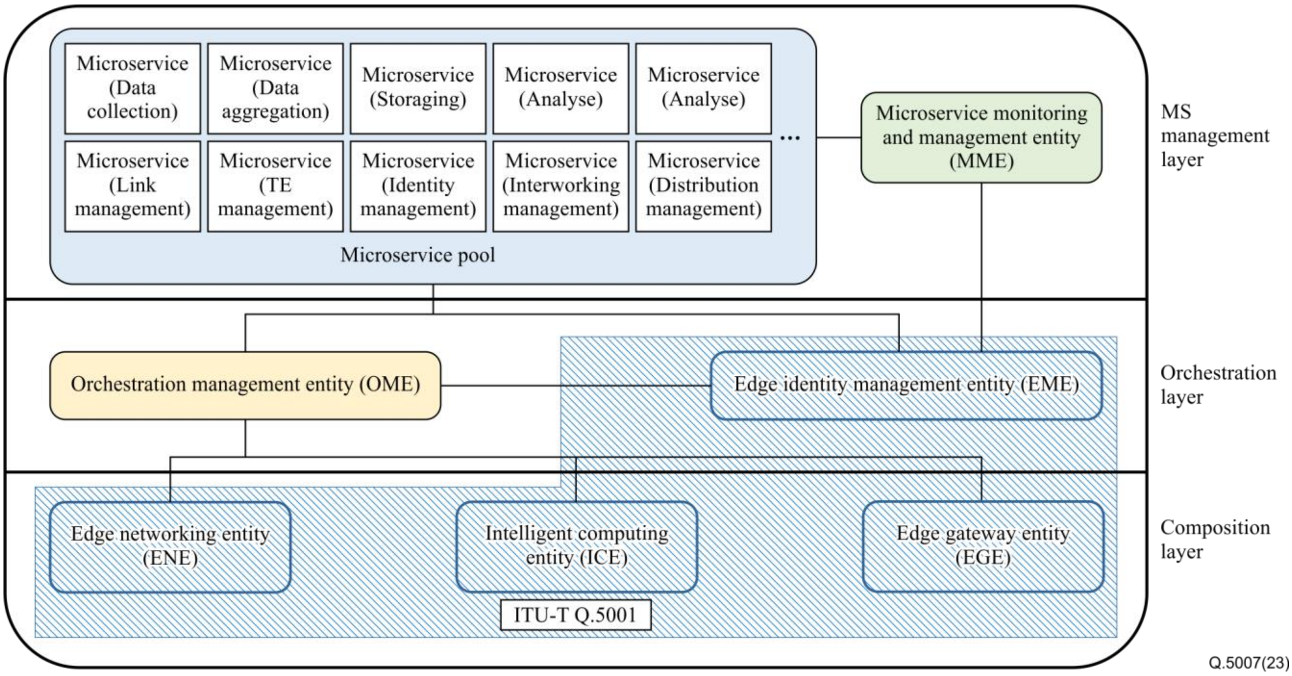

Figure 6-2 – Microservices based intelligent edge computing signalling framework

In Figure 6-2, the high-level IEC entities [ITU-T Q.5001] are depicted in the blue box, additional entities have been newly defined specifically for the microservice-based IEC architecture. Firstly, the framework defines a microservice pool, which serves as a repository to store and manage a multitude of individual operatable microservices. These individual microservices are required to be monitored for their own status, such as create, read, update, and delete (CRUD). These monitoring and management are handled by the microservice monitoring and management entity (MME).

Subsequently, these microservices can be deployed to IEC entities and compose them to operate their respective functions. The orchestration management entity (OME) handles these corresponding works. Both the microservice monitoring and management entity (MME) and the orchestration management entity (OME) can operate in a manner similar to the kubernetes gateway application programmable interface (API) and controller and management in kubernetes respectively [b-Kubernetes].

However, unlike cloud orchestration and monitoring management systems, an edge orchestration function manages microservices in different environments characterized by limited resources and low latency services in edge computing. So, only the monitoring and management entity (MME) is out of the scope in this Recommendation. This Recommendation focuses on implementing the EME, the ENE, the ICE, and the EGE with the orchestration management entity (OME) and microservices in the microservice pool.

The framework organizes these architectural entities into three layers; the composition layer, the orchestration layer, and the microservices management layer. These layers encompass various aspects such as protocol interfaces, protocol procedures, and message formats. The high-level IEC entitles are primarily located in the composition layer, with the exception of the EME, which is located in the orchestration layer. The remaining entities are distributed across different layers as appropriate.

The three layers are defined as follows:

– Composition layer: The IEC functional blocks within each IEC entity are deployed and operated by composing microservices from the microservice pool.
  - The ENE, ICE, and EGE entities are implemented by composing microservices.
  - Each microservices communicates with others to operate IEC functions.

- Microservices are managed securely using the EME.

– Orchestration layer: Each microservices is identified and orchestrated securely.
  - The orchestration management composes and manages microservices within each IEC entity.
  - The EME identifies all of the microservices before composing them.
  - This layer defines various IEC services.

– Microservice management layer: Each microservices is monitored and updated continuously.
  - This layer manages the microservice pool.
  - It monitors each microservice status.
  - It registers and updates microservices.

The microservice pool is defined as follows:

– Microservice pool: Various microservices are gathered and managed.
  - Topology management: A microservice that can provide network topology information for data exchange among microservices.
  - Distribution management: A microservice that can facilitate data distribution to other microservices based on the analysis and the transmission status.
  - Other microservices: Defining microservices is out of the scope of this Recommendation.

6.2 Orchestration management framework

Microservices are required to orchestrate with each other microservices in order to make various services, including intelligence analysis services on the edge side, and also each microservices is securely identified and managed within a kind of edge cluster or an edge computing device. In the orchestration layer, a new orchestration management entity (OME) has been defined. It allows a microservice to interwork with other microservices in the edge computing. Thus, the OME serves as a major entity in a microservice-based IEC architecture, connecting with other entities as depicted in Figure 6-3. It is recommended that all management flows are directed toward the OME.

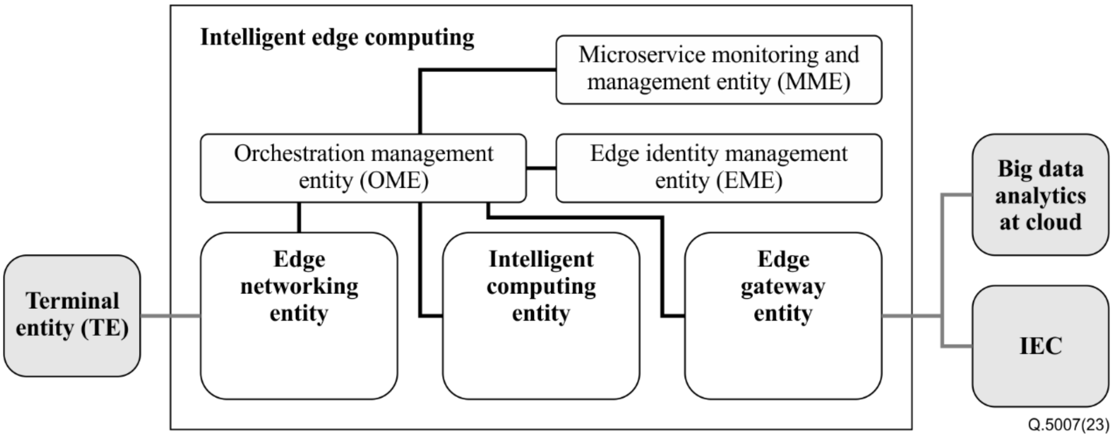

Figure 6-3 – Orchestration management framework for microservice based IEC

In the realm of cloud computing, orchestration management functions have already been deployed in a de facto way, such as kubernetes [b-Kubernetes]. However, the edge computing side presents lots of challenges because of specialized environments, such as limited resources, and stringent requirements for low latency, power, price, etc. Moreover, it is required to securely manage not only the microservices themselves but also raw data from external environments, including Internet of things (IoT) data, video data, etc.

Therefore, the orchestration management framework is designed to address these challenges as follows:

– Limited resource environments.
  - The framework is recommended to design requirements for three-low requirements, such as low-latency, low-power, and low-price.
  - The framework is recommended to consider a simple communication model, such as a central microservice is required to connect others.
  - The framework is recommended to decide the maximum number of microservices to be managed.

– Secure service and protecting raw data and microservices themselves.
  - The framework is required to design requirements for authentication, authorization, as well as account management functions.
  - The framework is recommended to protect both ingress raw data and egress processing data.
  - The framework is recommended to authorize external devices, which may include terminal entity (TE), cloud, or other IECs and inside microservices.

7 Signalling architecture

The microservice-based IEC architecture consists of many functional blocks deployed within each IEC entity, including the ENE, ICE, EGE and EME. Since all these functions can be classified under two primary IEC capabilities, these functions and interfaces are described with separated architectural frameworks.

The primary IEC capabilities are the intelligent edge networking and the intelligence data processing. This Recommendation focuses on the intelligence data processing capability.

7.1 Signalling architecture with orchestration management

Figure 7-1 represents the signalling architecture for an orchestration management point of view, incorporating new orchestration functions to enable interworking with other microservices within the IEC. These orchestration functions, such as the ingress management (IM) functions, service management functions, and egress management functions, allow the microservice to connect to each other securely with the EME. In collaborating with the EME, it is recommended that all microservices are identified and managed as well as external entities such as the TE, big data analytics in the cloud, and other IECs. Further, an explanation of interworking external entities can be described in [ITU-T Q.5001].

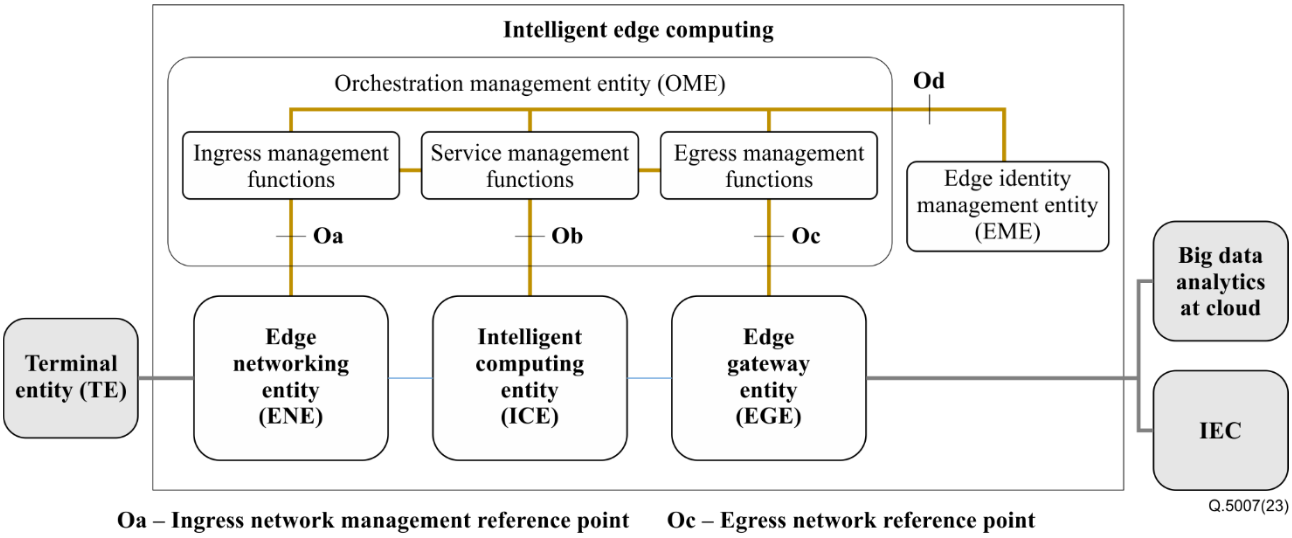

Figure 7-1 – Signalling architecture with orchestration management

The orchestration management entity (OME) interacts with other microservice-based IEC entities by the following functions:

– Ingress management functions:
  - Register IEC entities as well as raw data from TE. In the bootstrapping scenario, all of the data and entities register themselves with the EME through these functions. Thus, entities and data could be assigned unique identifiers. During the verification step, the validity of these entities' data is confirmed by the EME;
  - Filter out invalid data and microservices to protect and manage IEC services, so unpredictable operations can be prevented.

– Egress management functions:
  - Register external IEC entities, such as big data analytics at cloud and other IECs, so that processing data from ICE can be transmitted to validated external entities through the EGE;
  - Maintain validation of microservices for data transfer and helping to prevent unauthorized access to important data. This enhances the security of the service by sealing the outgoing interface.

– Service management functions:
  - Handle microservices themselves, facilitating the creation, reading, updating, and deletion (CRUD) of data from other microservices. It is recommended to store confident data, such as passwords, tokens, certifications, etc. in a secure storage;
  - Compose microservices to build, deploy, and manage IEC services. All control data are initially coming through these functions. Simultaneously, all microservices could be monitored by gathering log data.

7.2 Signalling architecture with intelligent service

Figure 7-2 represents the microservices-based architecture with data processing capability. The architecture consists of core intelligent edge computing entity (such as the ENE, ICE and EGE).

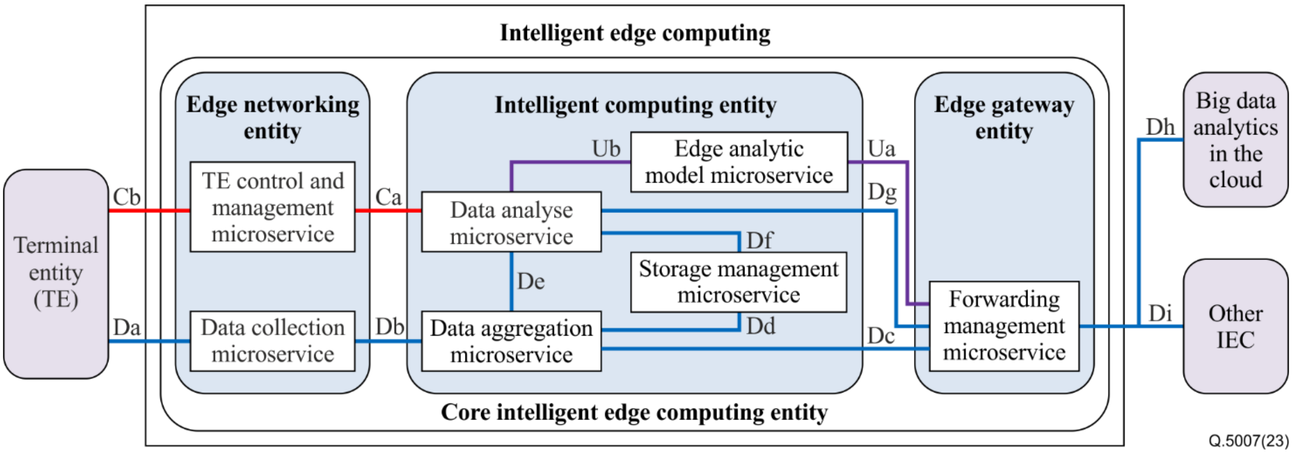

Figure 7-2 – Signalling architecture with intelligent service

The mapping between entities and microservices is described in Table 7-1.

Table 7-1 – Mapping between entities and microservices

Entities		Microservices
Edge networking entity	TE control and management functions	TE connection management microservice
	Data collection functions	Data collection microservice
Intelligent computing entity	Data analyse functions	Data analytics microservice
	Data aggregation functions	Data aggregation microservice
	Edge analytic model functions	Machine learning / deep learning (ML/DL) model management microservice
	Storage management functions	Storage management microservice
Edge gateway entity	Forwarding management functions	Forwarding management microservice

7.3 RESTful intelligent edge service

RESTful intelligent edge service APIs are not dependent on specific technology implementations. These APIs fully incorporate all aspects of hypertext transfer protocol (HTTP) v1.1 [b-IETF RFC 7231], including its request methods, response codes, and HTTP headers. Each specification for RESTful mobile edge service API should be included as the following:

– Purpose of the API.

– Uniform resource identifier (URIs) of resources including version number.

– HTTP methods [b-IETF RFC 7231] supported.

– Representations supported: JavaScript object notation (JSON) and, if applicable, extensible markup language (XML).

– Response schema(s).

– Request schema(s) when PUT, POST and GET are supported.

– Links supported (Optional in Level 2 APIs).

– Response status codes supported.

8 Protocol interfaces

8.1 Normative interfaces

8.1.1 Interface Cx

Cx denotes signalling interfaces to control TEs according to the result of edge analytics. Most of the control messages are required to be channelled through the following interfaces:

– Ca denotes an interface between the data analytics microservice and the TE control and management microservice, facilitating the control of the TE's actions.

– Cb denotes an interface between the TE control and management microservice and a TE, allowing for direct control over the TE's actions.

8.1.2 Interface Dx

Dx denotes signalling interfaces to forward data to build a data pipeline from TEs to the outside systems. In the IEC, raw data should be transformed into information enriched with meaningful metadata such as average, timestamp, location, etc., through edge analytics. Most of the data are required to be channelled through the following interfaces:

– Da denotes an interface between a TE and the data collection microservice to collect raw data.

– Db denotes an interface between the data collection microservice and the data aggregation microservice to aggregate data.

– Dc denotes an interface between the data aggregation microservice and the forwarding management microservice to directly forward aggregate data toward outside systems.

– Dd denotes an interface between the data aggregation microservice and the storage management microservice to store aggregated data which were provided to the data analytics microservice.

– De denotes an interface between the data aggregation microservice and the data analytics microservice to forward aggregate data directly to analyse them for online analysis.

– Df denotes an interface between the data analytics microservice and storage management microservice to store analysed data or to be provided as aggregation data from the database.

– Dg denotes an interface between the data analysis microservice and the forwarding management microservice to forward the result of the analysis.

– Dh denotes an interface between the forwarding management microservice and an external big data analytics cloud.

– Di denotes an interface between the forwarding management microservice and other IECs.

8.1.3 Interface Ux

Ux denotes signalling interfaces to request or to response to the ML model from big data analytics at cloud computing. Most of the control messages are required to send requests to the ML model periodically through the following interfaces:

– Ua denotes an interface between the edge analytic model microservice and the forwarding management microservice. This interface is utilized to send requests to the serving ML model to big data analytics at cloud computing.

– Ub denotes an interface between the data analysis microservice and the edge analytic model microservice to serve the ML model.

8.2 Orchestration interfaces

8.2.1 Interface Ox

Ox denotes signalling interfaces utilized to orchestrate all of the microservices in order to make various services in each entity. Most of the control messages for orchestration management are required to carry out exchanges through the following interfaces:

– Oa denotes an interface between the ENE and the ingress management functions to register and monitor the ENE entity as well as raw data from the TE.

– Ob denotes an interface between the ICE and the service management functions to handle and build microservices.

– Oc denotes an interface between the EGE and the egress management functions to register and manage outside IEC entities as well as processing data from own IEC microservices.

– Od denotes an interface between the PULL and PUSH message management microservice and the interworking management microservices to establish connectivity.

9 Protocol procedures

9.1 Signalling protocol procedures for orchestration functions

9.1.1 Procedure for orchestration microservices with raw-data filtering service

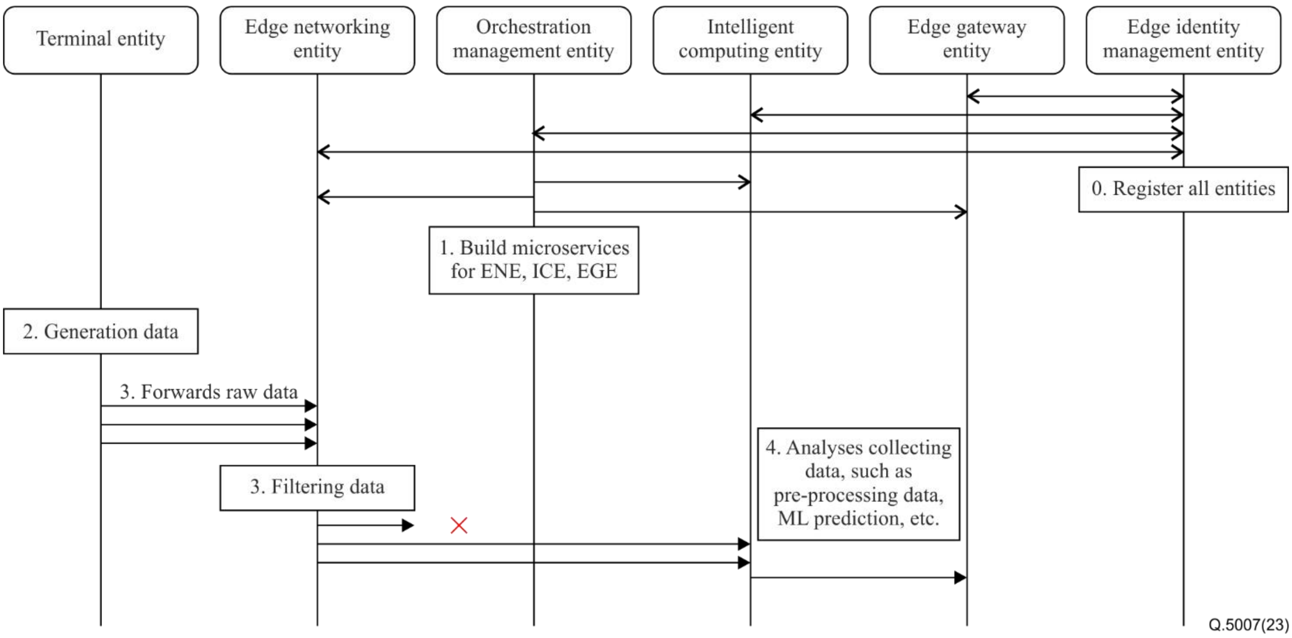

Figure 9-1 – Procedure for orchestrion microservices

0. IEC entities such as the ENE, ICE, EGE, and OME can be connected to the EME as a control channel through a bootstrapping procedure.

   NOTE 1 – Bootstrapping procedure could be out of scope.

   NOTE 2 – EME could be also connected to entities within the microservice management layer.

1. OME should build microservices as referenced from a service profile in the EME and deploy them to each of the IEC entities.

NOTE 3 – A filtering service is deployed within the ENE.

NOTE 4 – Service profile related procedures could be out of scope.

2. TE sends generated data to the ENE.

3. ENE filters out special data through the filtering microservice. And other data could be forwarded to the ICE.

4. ICE analyses aggregated data.

NOTE 5 – Aggregation data can be either stored in a storage or analysed by using a machine learning and AI model.

NOTE 6 – ICE functions could be out of scope.

9.2 Signalling protocol procedures for intelligent data processing

9.2.1 Procedures for data lifetime from collecting to forwarding data target entity

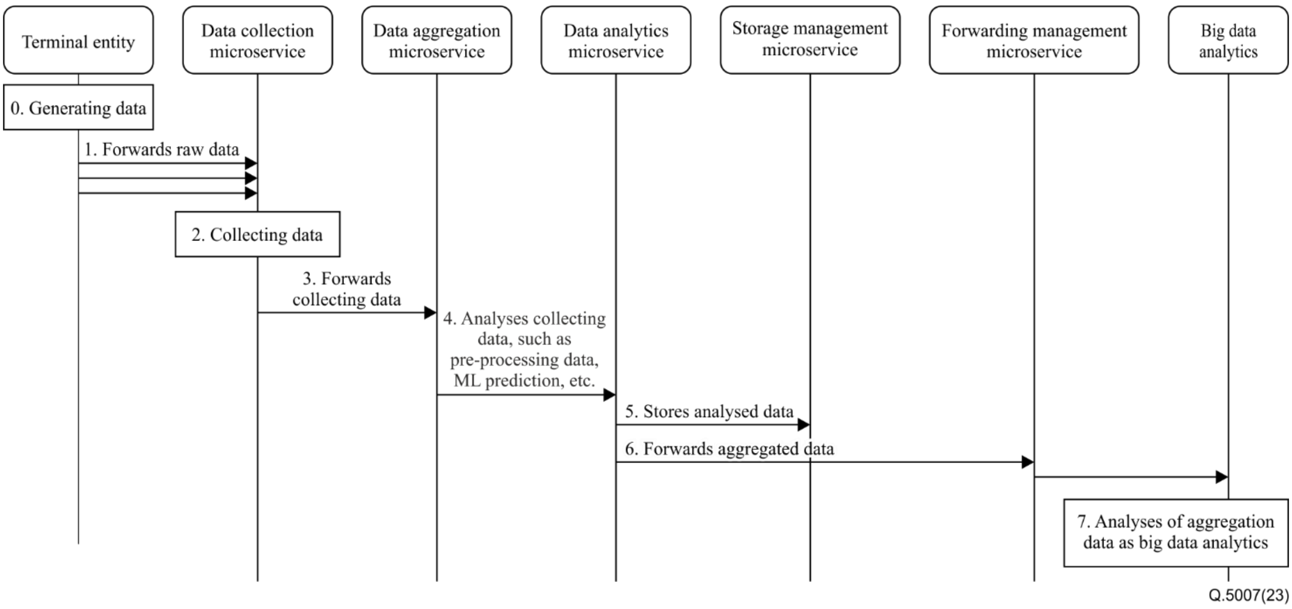

Figure 9-2 – Procedures for data lifetime

0. TE can generate raw data (or single unit data) triggered by some event.

NOTE 1 – Some triggering can be initiated by various sources including the TE itself, big data analytics, data collection microservice and so on.

1. TE forwards raw data to the data collection microservice.

NOTE 2 – Partitioned raw data can be gathered by the data collection microservices through connectivity between TE and the data collection microservice.

2. Data collection microservice accumulates partitioned data.

3. Data collection microservice forwards the collected data to the data aggregation microservice.

4. Data analytics microservice analyses aggregation data.

NOTE 3 – Aggregated data can be either stored in a storage or analysed directly as stream data processing.

NOTE 4 – Aggregated data can be first processed in a pre-processing phase, involving data quality (DQ) and extract, transform, and load (ETL). Secondly, normalized data can be processed by using the AI model. For instance, in order to predict future events, the ML prediction model can be applied to the edge analytics.

5. Edge analysed data are stored through the storage management microservice.

6. As a result of the edge analytics, the data analytics microservice forwards the analysed data to the big data analytics server on cloud computing.

   NOTE 5 – Similar to Note 4, according to the result of ML predictions, data forwarding can be controlled to reduce the traffic load between the data analytics microservice and the cloud server by using a type of video quality adaptation function, such as the video transcoding function.

7. Big data analytics of edge analysed data can be classified as new features, facilitating updates to the AI model.

9.2.2 Procedures for control TE

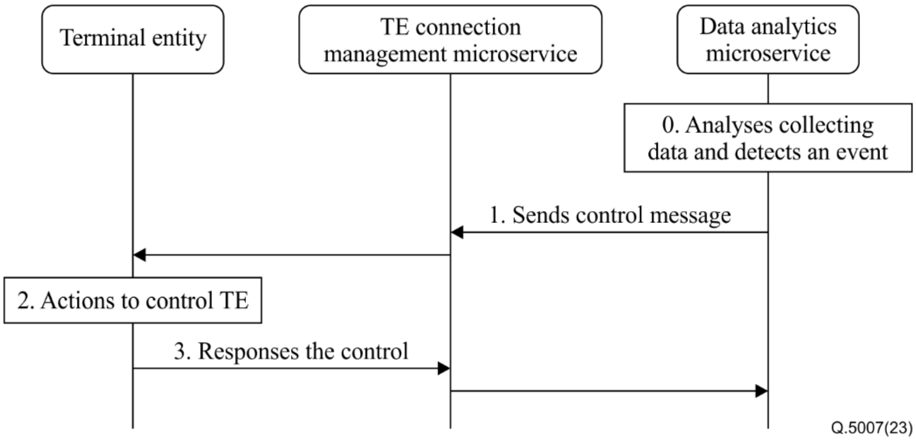

Figure 9-3 – Procedures for control TE

0. According to edge analysis for collecting data, an abnormal situation or a specific event has been predicted.

1. TE connection management microservice sends a request message to initiate a certain action.

   NOTE 1 – Types of actions can be defined by a service profile.

   NOTE 2 – In the case of video surveillance systems, TE connection management microservice can control a surveillance camera directly.

2. TE takes action immediately.

   NOTE 3 – Similar to Note 2, the camera can encode high-quality video in prediction time directly.

3. TE sends a response to this request to the TE connection management microservice.

9.2.3 Procedures for edge analytic model update

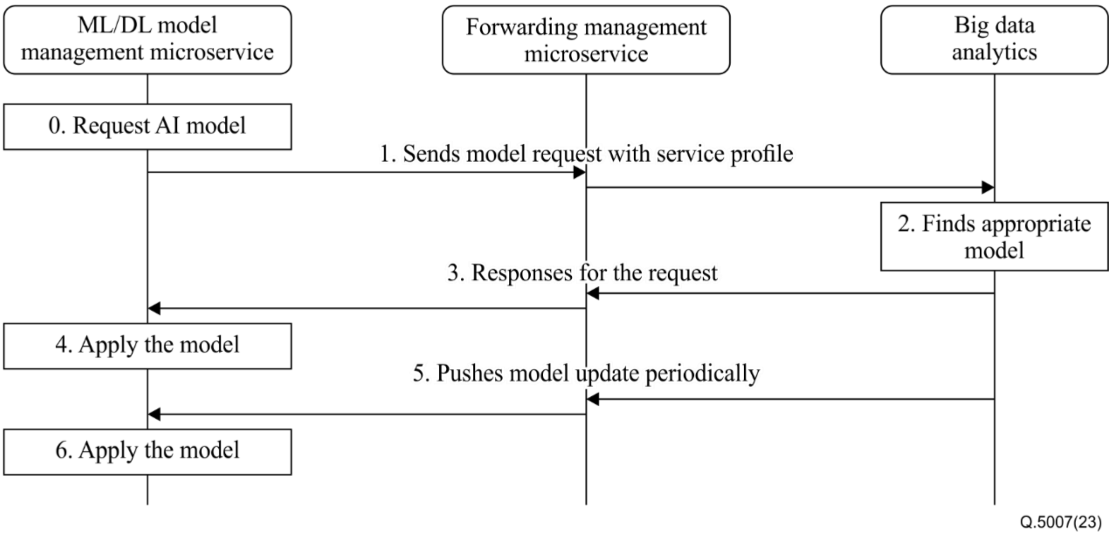

Figure 9-4 – Procedures for edge analytic model update

0. Before achieving edge analytics, an ML/DL model management microservice is required to request an AI model from big data analytics on a cloud server or a model repository in advance.

1. ML/DL model management microservice sends a request message for an AI model with a service profile.

   NOTE 1 – A model serving procedure can be substituted for the request.

2. In line with the service profile, the appropriate AI model is located at the big data analytics server or within the model repository.

3. The server responses to the request, attaching the AI model along with the necessary parameters.

   NOTE 2 – This procedure is similar to the model serving operation. More details of the serving operation could be out of scope.

4. The retrieved model is applied to the ML/DL model management microservice to proceed with the edge analytics.

5. After a reasonable period of time, the big data analytics server can push to update the model.

   NOTE 3 – In other cases, the ML/DL model management microservice can explicitly request a new AI model.

6. ML/DL model management microservice applies the model.

10 Security considerations

The microservice IEC architecture provides signalling architecture and requirements for edge computing environments. Thus, it is recommended to support security considerations in general based on the security framework from network functions virtualization [b-ITU-T X.1046] and [b-ETSI GS NFV-SEC 022] and security requirements and architecture for network slice management and orchestration [b-ITU-T X.1047].

Appendix I

Related works of microservices based edge computing

(This appendix does not form an integral part of this Recommendation.)

Microservices are a software development technique that structures an application as a collection of loosely coupled services. Microservices have the advantages as the following.

– Independent development – Each microservices can be easily developed based on their individual functionality.
– Independent deployment – Microservices can be individually deployed in any application, based on the respective services.
– Fault isolation – Even if one service of the application does not work, ensuring continuous operation.
– Mixed technology stack – Different languages and technologies can be used to build different services of the same application.
– Granular scaling – Individual components can scale as per need, there is no need to scale all the components together.

I.1 EdgeX

EdgeX is an open source and vendor neutral project in the Linux foundation [b-EdgeXFoundry]. EdgeX has architecture based on microservices which are loosely coupled software frameworks for IoT edge computing. This is hardware and OS agnostic and remains agnostic regarding the microservices implementation. Many of the microservices were in Java and are now in Go. C/C++ is envisioned for south side connectors and to address real time needs. This project enables and encourages growth in IoT solutions.

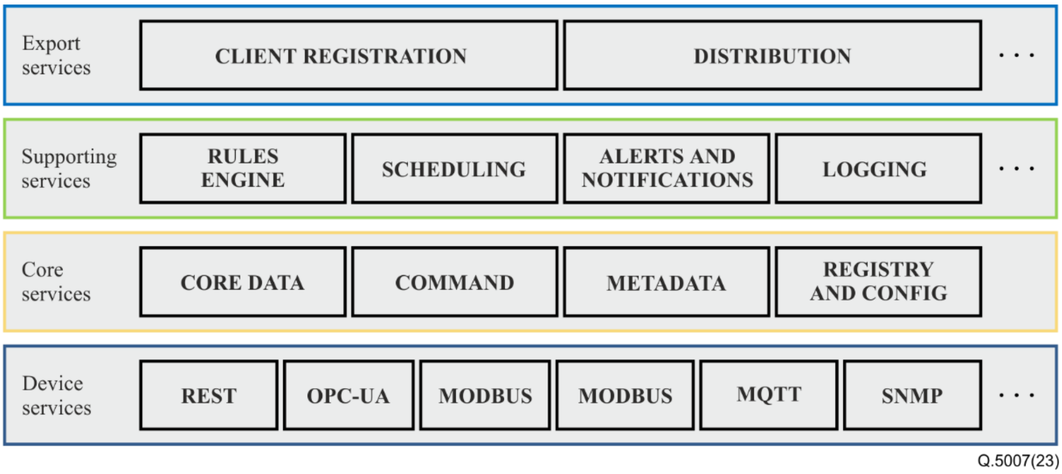

Figure I.1 – Edge X platform architecture [b-EdgeXFoundry]

The following currently exist:

– A collection of over a dozen microservices
  – Microservices are deployed via docker and docker compose
  – Written in multiple languages (Java, Go, C, etc.)
  – Several commonly used library projects (common domain objects, client libraries, etc.).

– Data flows among the microservices
  – REST communications between the microservices
  – Some services exchange data via message bus (core data to export services and rules engine)
  – Sensor data is collected by a device service from a thing
  – Data is passed to the core services for local persistence
  – Data is then passed to export services for transformation, formatting, filtering and can then be sent "north" to the enterprise/cloud systems
  – Data is then available for edge analysis and can trigger device actuation through the command service.

I.2 ETSI MEC

ETSI MEC (European Telecommunications Standards Institute multi-access edge computing) focuses on enabling edge computing at the access network (mobile or otherwise), thus bringing edge computing as close as possible to the user without it being in the user device [b-ETSI MEC].

To provide these new services and to make the most out of MEC, it is also important for the application developers and content providers to understand the main characteristics of the MEC environment and the additional services which distinguish MEC from other "edge computes", namely: extreme user proximity, ultra-low latency, high bandwidth, real time access to radio network and context information and location awareness.

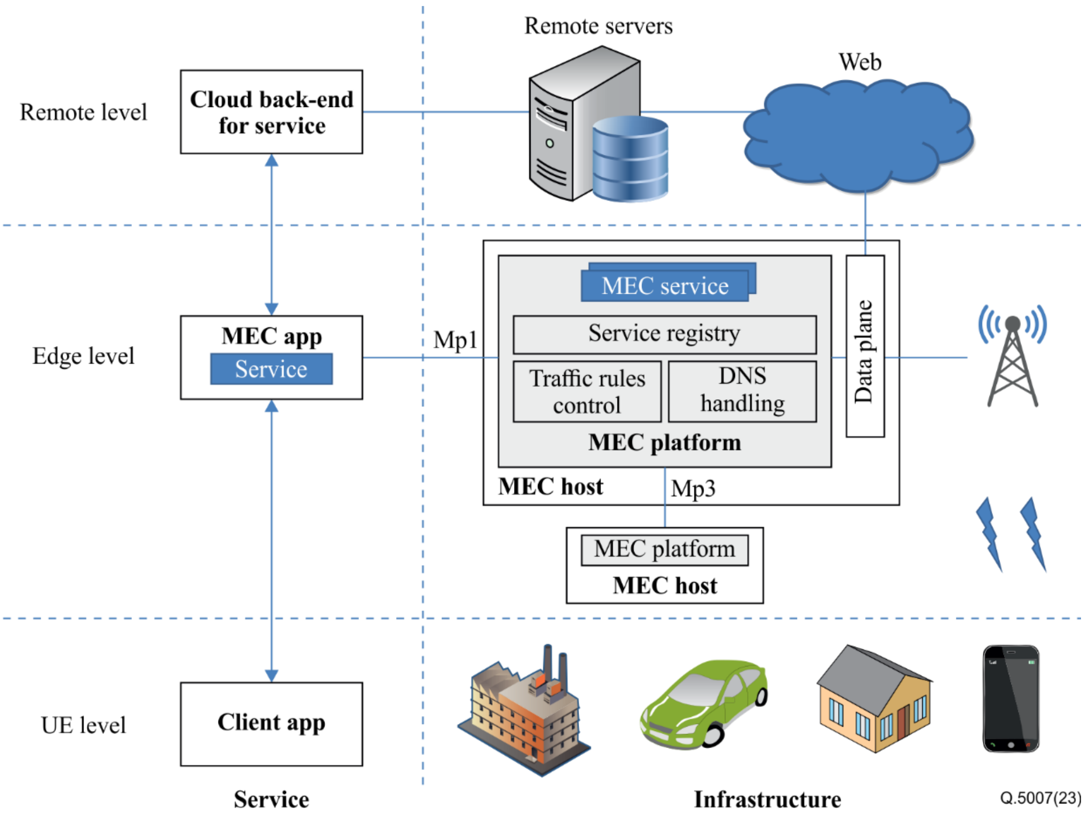

Figure I.2 – MEC platform architecture [b-ETSI MEC]

MEC offers application developers and content providers cloud-computing capabilities and an IT service environment at the edge of the network. Consequently, MEC introduces a standard for supporting an emerging cloud paradigm for software development communities. In fact, up to now a "traditional" client-server model of application development has been the dominating approach to developing applications for at least two decades. The emergence of edge computing, e.g., MEC, evolves this environment by introducing an intermediate element at the network edge.

An MEC is distinct from a traditional cloud. It may offer significant advantages to application components / services running there, while also presenting some challenges, e.g., higher cost, relatively small compute footprint, good local but not global reachability, etc. As such, it is crucial for an application developer to design with specific intent towards running some application components at the network edge when developing for an MEC.

This results in a new development model with three "locations": client, near server, and far server. The client location can be a traditional smartphone or other wireless connected compute elements in a car, smart home, or an industrial location that can run dedicated client applications. The model is quite new to most software developers, and while modern development paradigms (e.g., microservices) make it easier to adapt to it, a clear and concise summary of this new development model and guidance on how to properly approach it will help accelerate the application development for the network edge and thus accelerate MEC adoption.

In Figure I.2, an MEC host, usually deployed at the network edge, contains an MEC platform and the compute, storage, and network resources for applications in VMs or containers. The MEC platform offers a secure environment where MEC applications may, via RESTful APIs, discover, advertise, consume and offer services.

Appendix II

Use cases of microservices based edge computing

(This appendix does not form an integral part of this Recommendation.)

II.1 Infotainment services

II.1.1 Immersive multimedia service delivery in a high-speed scenario

Immersive multimedia services, such as virtual reality (VR) and augment reality (AR), are anticipated to be key offerings in the fifth-generation mobile network (5G) mobile communication [b-3GPP TR 22.891]. These services require higher throughput and reduced latency compared to legacy 4G mobile communication systems, so it is important to ensure service continuity without interruption of service irrespective of the user's movement and network architecture. However, since the service coverage that can be provided by an IEC is limited on multiple IECs deployment scenarios, it may be difficult to guarantee service continuity in high-speed scenarios where a user has high-speed mobility. To solve this problem, it is necessary for the IEC to perform fast and intelligent horizontal service migration between IECs, which is achieved by the topology management function.

The topology management function has a role for IEC chaining that makes a logical map between nearby IECs and current serving IECs and for data delivery that selects and forwards contents among IECs configured by the IEC chaining map, considering user mobility, network architecture and service requirement. With topology management function, since IEC could logically chain among the nearby IECs and intelligently distribute service contents to selected IECs, service continuity could be guaranteed in a high-speed scenario.

An operation of topology management block may be performed by prediction-based chaining with machine learning taking into accounts user mobility, network architecture and service requirements. Figure II.1 shows an example operation of a topology management block in a high-speed scenario.

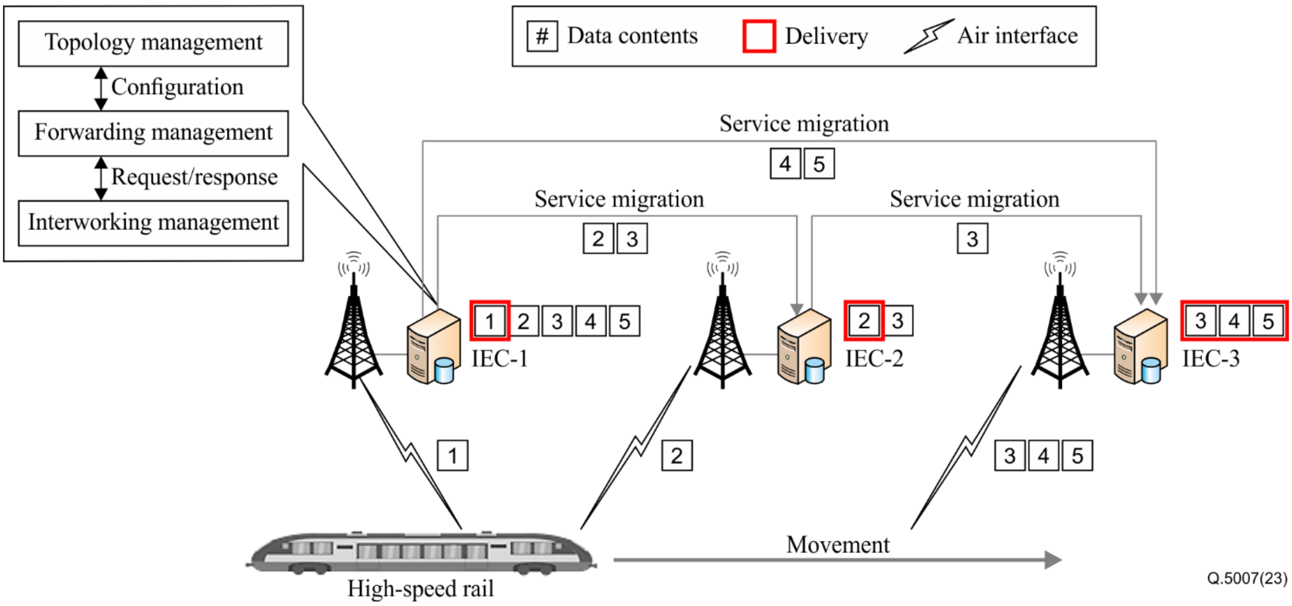

Figure II.1 – A use case of topology management block

II.1.2 Analysis and distribution of operation and management data in a high-speed scenario

Mobile communication service in high-speed scenarios, such as high-speed trains and self-driving vehicles are recognized as a key scenario of 5G services. In particular it is required to a high reliability and low latency for data transmission for the services, since the majority of traffic is critical information for passenger safety, advanced driving assistance for autonomous driving and operation of the railway system. For that reason, these kinds of services are classified as an ultra-reliable and low latency communications (URLLC) services in the 5G use cases [b-ITU-R M.2083-0]. The URLLC service requires sub-millisecond latency with error rates that are lower than one packet loss in 10^5 packets.

In a high-speed scenario, various types of sensors and cameras can be equipped with a vehicle[1] to make control and measurement information. In addition, multimedia information gathered from the camera is especially useful for situation recognition in emergency cases, such as car accidents, fault detections and fire on the carriages. Therefore, it is necessary to deliver the information to proper entities on time with low latency and high reliability since the data is tightly related to passenger safety and system operation.

Based on the above description, the following data processing is required in a high-speed scenario:

– storing data temporarily,

– analysing the data to recognize a situation,

– and distributing the data to entities that are necessary for it.

These procedures can be divided into two parts, such as data analysis and distribution block. Namely, the first and second operations belong to the data analysis function, and the third operation corresponds to the data distribution function.

The IEC serves as a data storage and processing entity that is located close to where data is generated and used, for low latency services with an intelligent manner. Consequently, the IEC is a suitable entity to perform the data analysis that determines whether an operation status is normal or not, and the data distribution which transmits the data to the relevant entities if necessary. In order to provide passenger safety, autonomous driving and operation of the railway system, it is necessary to define functional blocks for data analysis and distribution function in the microservices based IEC signalling framework.

Figure II.2 shows a scenario for data analysis and distribution block in operation and management traffic transfer. In this example, each IEC performs analysis from the gathered information, and then it transmits the data to other nodes where the information is needed, with priority. For example, if data from step 3 and step 5 have critical information for safety, those have higher transmission priority than other data. So, the data analysis and distribution block make a decision on the delivery order, which should be transmitted after data analysis.

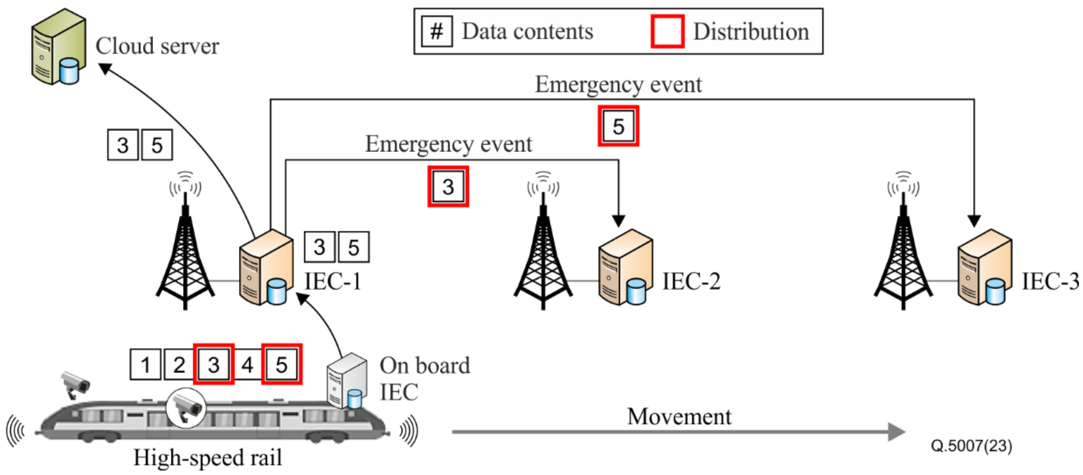

Figure II.2 – A use case of data analysis and distribution block in a high-speed scenario

1 In this use case, vehicle means various types of transportation such as cars, high-speed trains and metros.

II.1.3 Infotainment and autonomous driving service in vehicle-to-everything (V2X)

5G mobile communication system is expected to offer an applicable platform for vehicle-to-everything (V2X) service, leveraging the emerging technologies related to new radio (NR), edge computing, network softwarization, service-based architecture and artificial intelligence. Within this framework, integration of V2X with edge computing is able to provide a viable and effective solution that can accelerate V2X service.

In V2X, there are two major types of use cases, such as infotainment and autonomous driving. To achieve these use cases, various types of vehicles, user devices and sensors are needed to be fully connected.

Regarding infotainment service, both onboard users and nearby roadside will also be able to generate, share, and retrieve diverse content such as video streaming, AR/VR (augmented reality/virtual reality) services, or simple messaging transferring. For example, let us assume a scenario where passengers are onboarding and they are receiving streaming VR service connected to a cloud server. In this use case, IEC can facilitate content processing functions such as video rendering for computational overhead reduction of end devices. IEC is also able to provide a role of an intermediate node between the cloud server and the user device for contents' delivery. Since high-speed mobility is basically required in V2X service, IEC is able to support a fast data distribution function for service migration between either IECs or IEC and cloud servers considering vehicle speed, IEC deployment and quality of service (QoS) requirement.

In the context of autonomous driving, it is imperative that vehicles, devices and sensors are capable of communicating and exchanging information related to traffic conditions, surrounding environment and maps. In the service, it is essential to provide low latency and reliable communication since the data is directly related to safety. For the use case, IEC makes it possible to provide in time delivery to nodes where information is required, which can be achieved by situation recognition based on the intelligent data analysis. For example, if there is a case where VR contents and measurement data for auto driving are simultaneously transmitted over a radio link on the same IEC, it would be reasonable for measurement data to have higher priority over user content, especially in emergency situations such as car accidents. At the moment, IEC can provide transmission policy functions for data delivery collaboration with artificial intelligence algorithms such as machine learning.

Based on the above use cases, IEC is expected to be an essential function for infotainment and autonomous driving services in vehicular type communications networks as shown in Figure II.3.

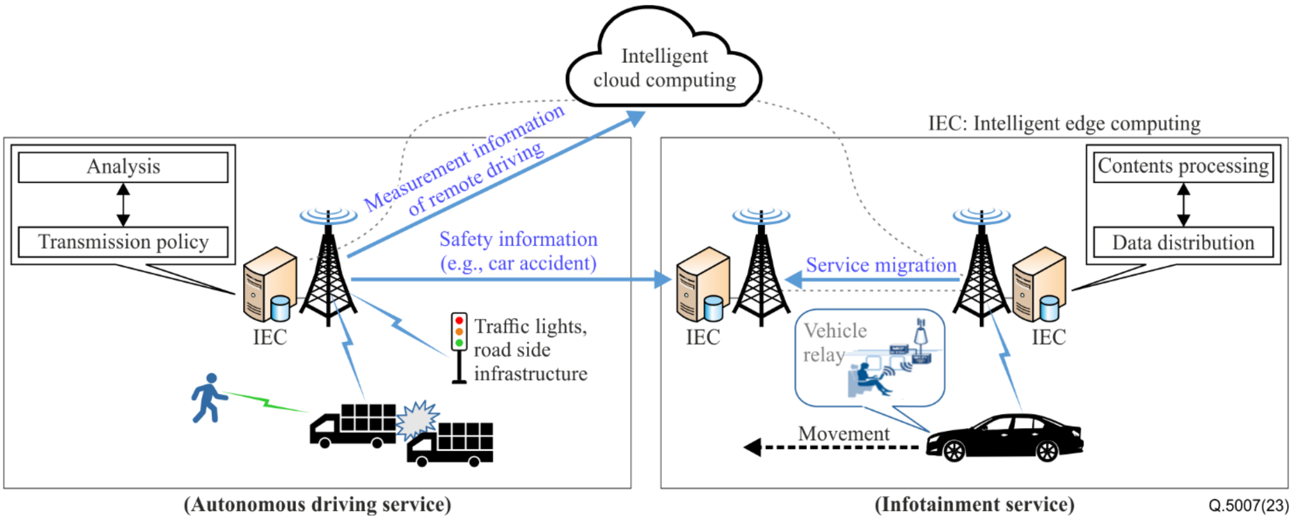

Figure II.3 – Infotainment and autonomous driving service in V2X

II.2 Factory services

II.2.1 General reinforcement learning based device (sorter, stacker, etc.) control service on smart factory scenarios

Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment to maximize some notion of cumulative reward within a given environment. It stands as one of three fundamental paradigms in machine learning paradigms, alongside supervised learning and unsupervised learning [b-Reinforcement Learning]. It differs from supervised learning in that labelled input/output pairs need not be presented, and sub-optimal actions need not be explicitly corrected. Instead, it emphasizes on striking a balance between exploration (of uncharted territory) and exploitation (of current knowledge). The environment is typically formulated as a Markov decision process (MDP), as many reinforcement learning algorithms for this context utilize dynamic programming techniques. The main difference between the classical dynamic programming methods and reinforcement learning algorithms is that the latter do not assume knowledge of an exact mathematical model of the MDP, and they target large MDPs where exact methods become infeasible. Programming logic controller (PLC) is an industrial digital computer that continuously monitors the state of input devices and makes decisions to control the state of output devices.

This use case shows the application of multiple smart factory tasks such as multi stacker, grid sorter, etc. based on the intelligent edge computing as shown in Figure II.4. For the purpose of implementing this scenario, we need to add a general RL controller which consists of the environment, agents and so on.

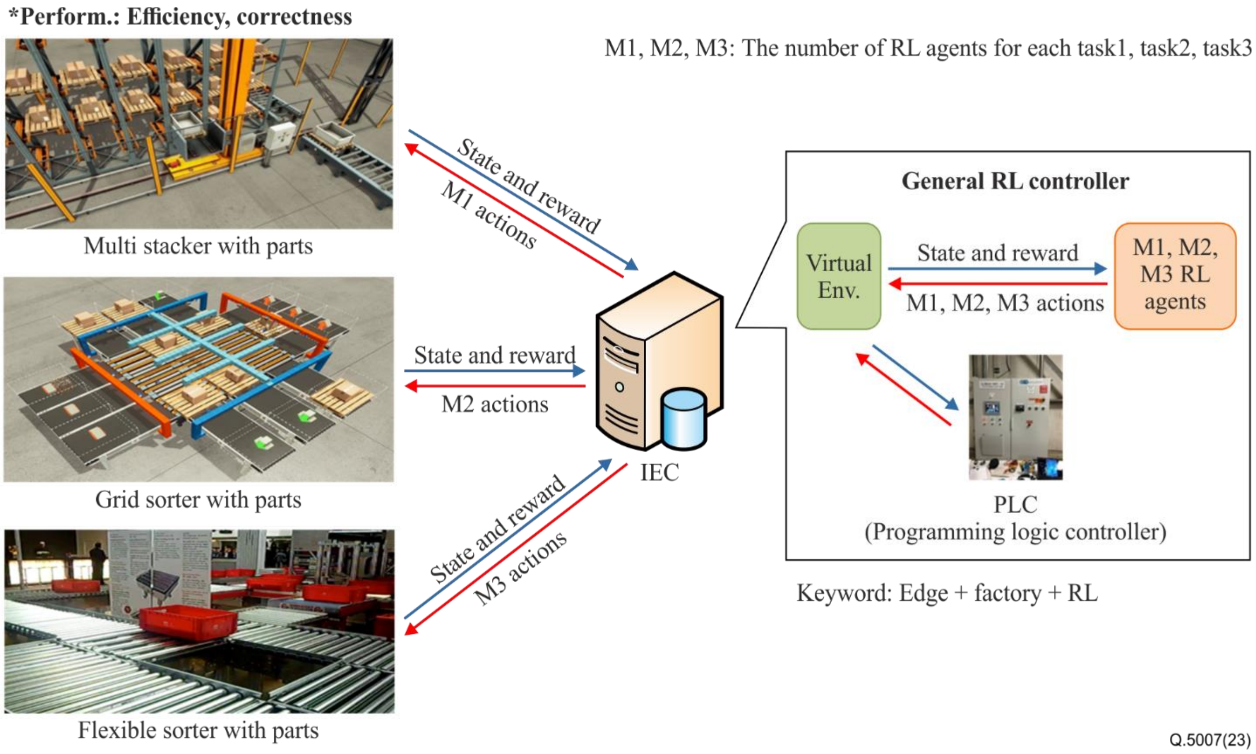

Figure II.4 – A use case of general reinforcement learning based device control service

II.3 Education services

II.3.1 Recommendation and scoring system for online judge on smart education scenario

The easiest and certain way to improve programming skills is repetitive coding. To this end, users can solve various programming problems and submit their codes to the online judge (OJ)[2] system to easily check its answer and efficiency. The users would consider the solution to problems and more effective programming codes, and the competencies as a programmer would also be enhanced in this process.

Figure II.5 illustrates a use case where the online judge system which is expanded as a problem recommendation system applying IEC. Existing online judge systems can have trouble with crashing or delayed servers in the situation that multiple coding competitions are held concurrently. And problems are sorted simply in the level order in the existing online judge system. Also, personal identity is not for consideration in the system. Thus, it needs to be considered to analyse about metadata such as type of school, student's background, strengths, and weaknesses for customized education. In IEC, the judge controller and recommendation service can be microservices to overcome these weak points. It can be used as a unit of school or event, and it is possible to use the entire participants' data from the intelligent cloud computing or partial unit data from the IEC.

In this use case, students solve the programming problems and submit their codes to the scoring and recommendation system. The system analyses the codes to score and judge their level. Also, the system recommends the next programming quiz from the problem DB by analysing information such as the overall correct answer rate, individual problem-solving history, strengths and weaknesses. Teachers can be provided with students' achievement data for teaching and evaluation.

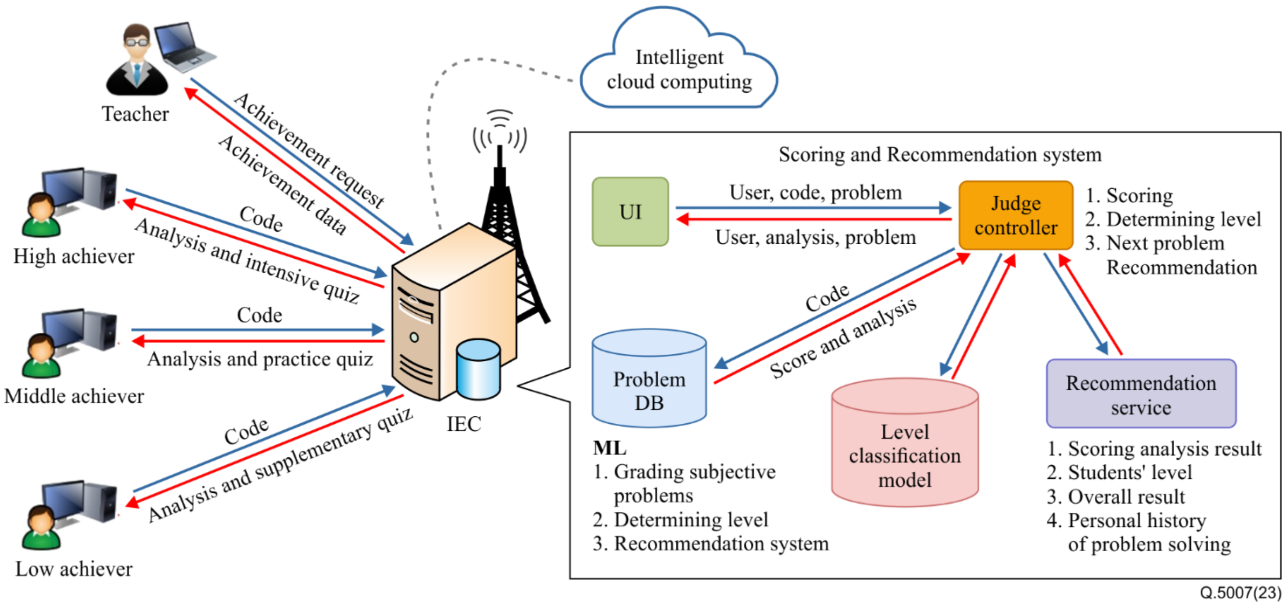

Figure II.5 – Recommendation and scoring system for online judge

II.3.2 Blockchain scenario for online judge

Blockchain technology has the potential to improve IEC's performance in three ways. The first is trustworthiness. External hacking or internal administrators can tamper with IEC information. When information is stored in the blockchain, the integrity of the data can be ensured by detecting and recovering information that has been falsified by the IEC. The second factor is accessibility. Due to a hardware/software malfunction or a denial of service (DOS) attack, IEC may perform erratically. If

---

2 Online judge: System for evaluating accuracy and efficiency of algorithms that are submitted by users automatically.

the information is recorded in the blockchain, people can receive the same service as before through various IECs. The third factor is compatibility. It is required to specify the information exchange method and format in advance in order to use the information handled by one IEC in another IEC. Any IEC can easily link to the blockchain and use the information if the manner of storing and exchanging information in the blockchain is standardized. It can, however be implemented in a variety of ways depending on the type of blockchain (public, private, consortium). For school, we will use private or a consortium type.

Figure II.6 depicts an IEC example of an online deterrent using blockchain technology. In this use case, users (teachers, students) continue to use the IEC's online deterrence service. Instead of the current IEC storing the problem / solution information and using it during service provision, the new IEC saves the information in the blockchain and uses the blockchain's information during service supply. Other IECs can access the information recorded on the blockchain if one IEC stores it on the blockchain. By using access control or encryption techniques, selective sharing of information stored in the blockchain is possible. If users set their account information stored in the blockchain to be accessible to other IECs, they can use the same online deterrence service independently of a specific IEC. Even if a teacher or student's affiliation changes, the history of using the online judge service can be maintained. Selective sharing of information recorded in the blockchain can be accomplished via access control or encryption mechanisms. Users can use the same online deterrence service regardless of which IEC they use if they set their account information in the blockchain to be available to other IECs. The history of using the online judge service can be kept even if a teacher's or student's affiliation changes. Individual transactions are used to store and retrieve information on the blockchain, and transaction logs can be used to conduct audits. All blockchain-connected IECs can detect and respond to unexpected access to information by specified individuals or IECs. The time it is registered in the blockchain can be used to verify whether or not the code used to solve the problem is plagiarized, as well as the copyright for the problem itself.

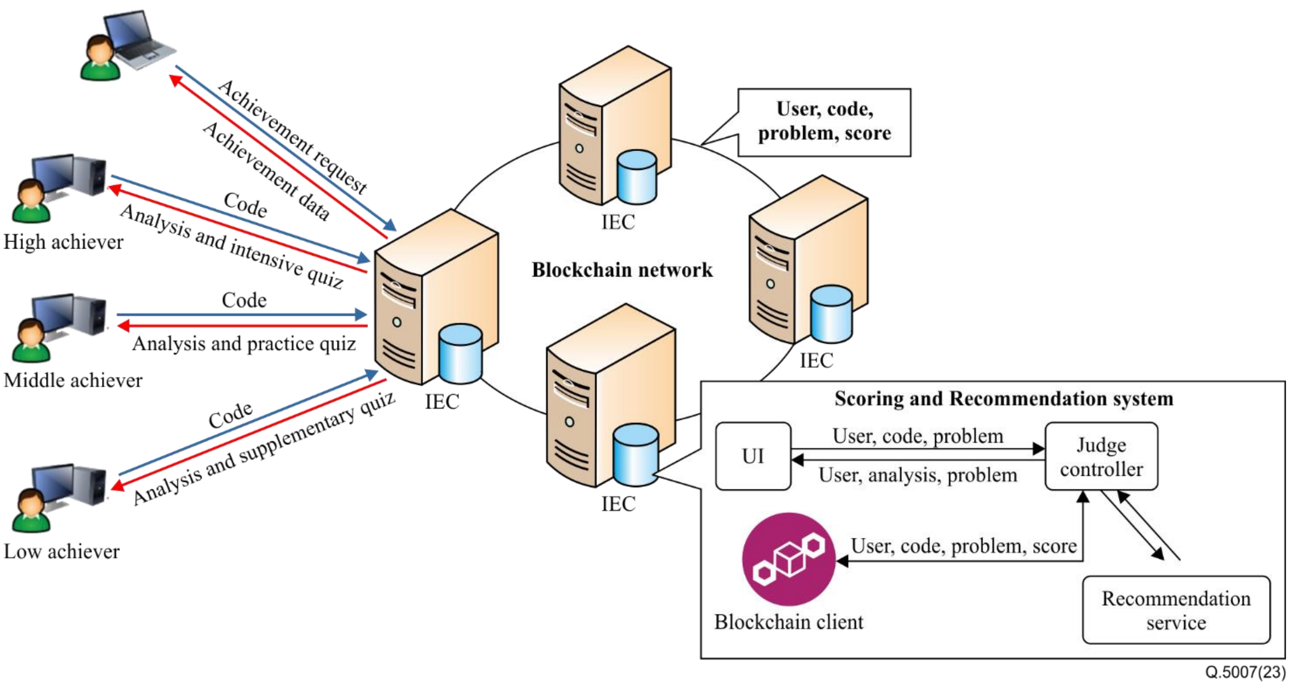

Figure II.6 – Blockchain scenario for online judge

II.4 Convergence services

II.4.1 Convergence intelligence edge service scenario by cross-domain workflow framework

Recently, there have been emerging manufacturing technologies such as smart manufacturing, which is based on a cross-domain workflow framework that derives optimal results by linking different tasks or different purposed domains in different physical spaces. For instance, some areas between where the customer's demand occurs and where customized manufacturing occurs are physically located in different areas.

Thus, the cross-domain framework offers various workflow configurations, allowing for combining different special domains, different business domains, or different purpose domains. Moreover, it adapts seamlessly to alterations in data generation status or environment. Therefore, the framework finally achieves an intelligent service that organizes, executes, and controls various workflows. As a result of building a practical intelligent service, it can be represented in the convergence intelligence service.

The core of the cross-domain workflow configuration technology is the orchestration technology, which can configure, execute, and control microservice-type workflows in order to compose various microservices. In this scenario, it describes a cooperative intelligence service that composes one or more workflows from different spatial domains either inside edge or outside the edge area, and applies each workflows' process hierarchically by connecting pipelining using intelligent microservice edge computing (IEC) technology.

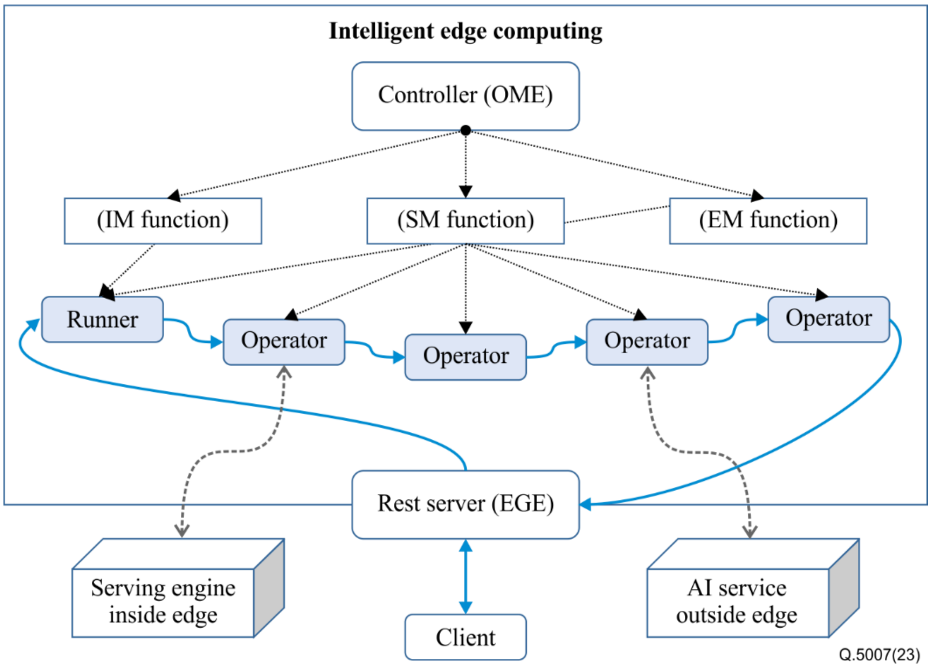

Figure II.7 – Cross-domain scalable workflow engine container composition method for convergence intelligence edge service

Figure II.7 shows components of configured various edge engine containers to build convergence intelligence service. The intelligence service composes various microservice engines representing an operator controlled by an orchestration method as OME (orchestration management entity) to provide a convergence intelligence service suitable for a specific purpose by compiling pretrained knowledge such as AI service in one or more business domains. In addition, the runner can be implemented by IM (ingress management) and EM (egress management) functions, and all of these operators can be implemented by SM (service management) functions.

The REST server which is represented by EGE, is running as a runner microservice, so that the server receives the client's request and calls the operator inside the edge area sequentially. Then the operator returns the result of the final process by the last operator to the service client. Here, each operator is interworking by REST API using the URI information.

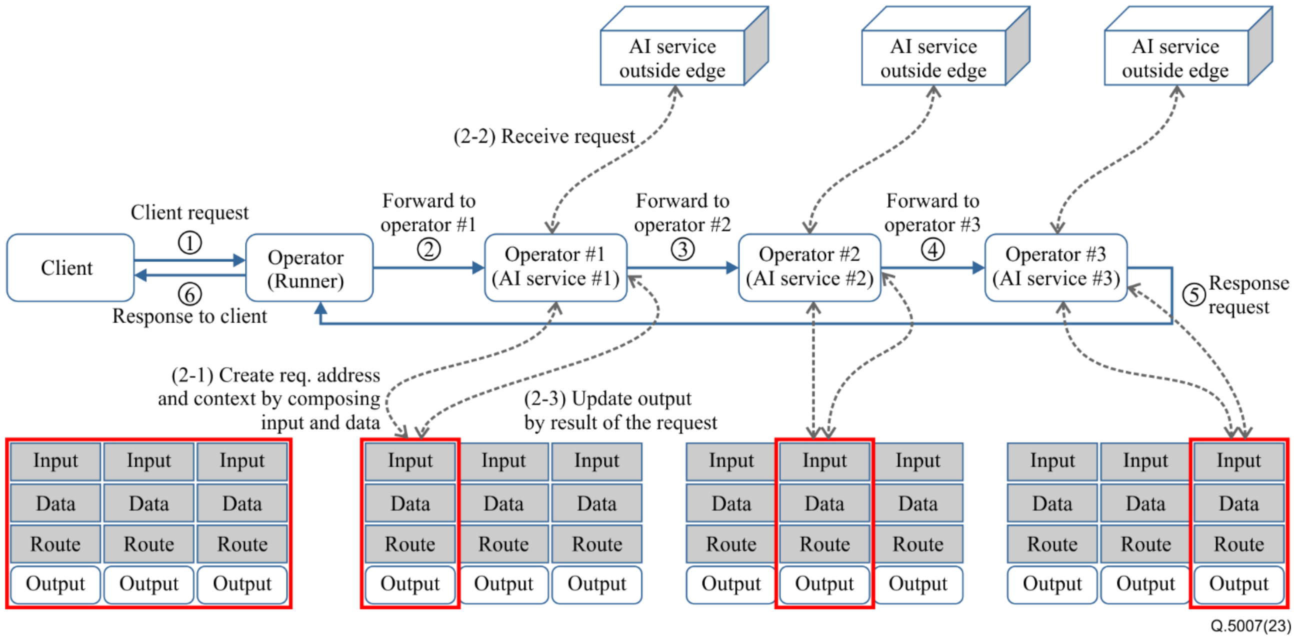

Figure II.8 – Internal processing procedure of engine container as operators for convergence intelligence edge service

Figure II.8 shows the internal processing procedure for composing engine containers to build pipelining workflows.

– 1. A client requests a service to the runner as operator.

– 2. Runner forwards the request to the operator #1.

– 2.1 Operator #1 transfers query data that creates URI address and context for request by composing input, data, and route.

– 2.2 The request is proceeded by using AI service through the Rest API.

– 2.3 Operator #1 updates output through results of 2.2 including input field, data field, route field, and output field in the responding data.

– 3. The updated request forwards it to operator #2 for composing the second column, and it is proceeded just as step 2.

– 4. The updated request forwards it to operator #3 for composing the third column, and it is proceeded just as step 3.

– 5. Operator #3 responds to the result of the process that aggregates columns to the runner.

– 6. Runner forwards the result to the client.

Bibliography

[b-ITU-T X.1046] Recommendation ITU-T X.1046 (2020), *Framework of software-defined security in software-defined networks/network functions virtualization networks.*

[b-ITU-T X.1047] Recommendation ITU-T X.1047 (2021), *Security requirements and architecture for network slice management and orchestration*.

[b-ITU-T Y.3500] Recommendation ITU-T Y.3500 (2014), *Information technology – Cloud computing – Overview and vocabulary*.

[b-ITU-R M.2083-0] Recommendation ITU-R M.2083-0 (2015), *IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond.*

[b-EdgeXFoundry] Edge X Foundry Homepage. (n.d), *The Preferred Open Source Edge Platform*. <https://www.edgexfoundry.org/>

[b-ETSI GS NFV-SEC 022] ETSI GS NFV-SEC 022 V2.7.1 (2020), *Network Functions Virtualisation (NFV) Release 2; Security; Access Token Specification for API Access.* <https://www.etsi.org/deliver/etsi_gs/NFV-SEC/001_099/022/02.07.01_60/gs_NFV-SEC022v020701p.pdf>

[b-ETSI MEC] Sabella, D., Sukhomlinov, V., Trang, L., Kekki, S., Paglierani, P., Rossbach, R., Li, X., Fang, Y., Druta, D., Giust, F., Cominardi, L., Featherstone, W., Pike, B., and Hadad, S. (2019), *Developing Software for Multi-Access Edge Computing*. ETSI White Paper No. 20. <https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp20ed2_MEC_SoftwareDevelopment.pdf>

[b-IETF RFC 7231] IETF RFC 7231 (2014), *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content.* <https://datatracker.ietf.org/doc/html/rfc7231>

[b-3GPP TR 22.891] 3GPP Technical Report 22.891 V14.2.0 (2016), *Feasibility Study on New Services and Markets Technology Enablers; Stage 1 (Release 14).* <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2897>

[b-Kubernetes] Kubernetes Homepage. <https://kubernetes.io/ko/>

[b-Reinforcement Learning] Kaelbling, L.P., Littman, M.L., and Moore, A.W. (1996), *Reinforcement Learning: A Survey*. Journal of Artificial Intelligence Research Vol.4. <https://www.jair.org/index.php/jair/article/view/10166>

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	Tariff and accounting principles and international telecommunication/ICT economic and policy issues
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Environment and ICTs, climate change, e-waste, energy efficiency; construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	**Switching and signalling, and associated measurements and tests**
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects, next-generation networks, Internet of Things and smart cities
Series Z	Languages and general software aspects for telecommunication systems

Published in Switzerland
Geneva, 2024