IEEE SA
STANDARDS
ASSOCIATION

IEEE Standard for Edge/Fog Manageability and Orchestration

STANDARDS

IEEE Communications Society

Developed by the
Edge, Fog, Cloud Communications with IOT and Big Data Standards Committee

IEEE Std 1935™-2023

IEEE

IEEE Standard for Edge/Fog Manageability and Orchestration

Developed by the

Edge, Fog, Cloud Communications with IOT and Big Data Standards Committee
of the
IEEE Communications Society

Approved 15 February 2023

IEEE SA Standards Board

Abstract: With the innovation of mobile applications and the arrival of the fifth generation of telecommunication, edge computing has become a popular scheme due to its geographical proximity to end users. The overall architecture has the advantage of lower latency and improved user experience. However, because of physical limitations, management and orchestration in the edge system is necessary to maintain operations, common functions, and application lifecycle. Accordingly, a standardized, orchestration-wise design to simplify the process and offer better system performance and user experience was introduced by the IEEE P1935 working group. The framework, architecture, and procedures of the edge/fog system as well as its application lifecycle management are involved in this standard.

Keywords: edge computing, IEEE 1935, management and orchestration

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF:	ISBN 978-1-5044-9543-1	STD26033
Print:	ISBN 978-1-5044-9544-8	STDPD26033

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (https://standards.ieee.org/ipr/disclaimers.html), appear in all standards and may be found under the heading "Important Notices and Disclaimers Concerning IEEE Standards Documents."

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE SA) Standards Board. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA, and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning this standard, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents**.

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and Standards Coordinating Committees are not able to provide an instant response to comments, or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the IEEE SA myProject system.[1] An IEEE account is needed to access the application.

Comments on standards should be submitted using the Contact Us form.[2]

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under US and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

[1]Available at: https://development.standards.ieee.org/myproject-web/public/view.html#landing.
[2]Available at: https://standards.ieee.org/content/ieee-standards/en/about/contact/index.html.

[3]Available at: https://ieeexplore.ieee.org/browse/standards/collection/ieee.
[4]Available at: https://standards.ieee.org/standard/index.html.
[5]Available at: https://standards.ieee.org/about/sasb/patcom/materials.html.

reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

IMPORTANT NOTICE

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. IEEE Standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

Participants

At the time this standard was completed, the Standard for Edge/Fog Manageability and Orchestration Working Group had the following entity membership:

Hung-Yu Wei, *Chair*
Hazim Dahir, *Vice Chair*
Yao Chiang, *Secretary*

Organization Represented	*Name of Representative*
The Chinese University of Hong Kong	Raymond Yeung
Cisco	Hazim Dahir
Mitsubishi Electric Corporation	Tetsushi Matsuda
National Taiwan University	Hung-Yu Wei
National Yang Ming Chiao Tung University	John K. Zao

The Working Group gratefully acknowledges the contributions of the following participants. Without their assistance and dedication, this standard would not have been completed.

Yao Chiang
Tse-Yu Chen

The following members of the entity Standards Association balloting group voted on this guide. Balloters may have voted for approval, disapproval, or abstention.

0xSenses Corporation
1stCycle Corporation
Beckhoff Automation
Cadence Design Systems
China Electronic Standardization Institute
China Power Engineering Consulting Group Co., Ltd.
The Chinese University of Hong Kong
Institute of Biomedical Engineering, Chinese Academy of Medical Sciences & Peking Union Medical College
Institute of Geographical Sciences and Nature Resources Research
Mitsubishi Electric Corporation
National Institutes for Food and Drug Control, China
National Taiwan University
Shenzhen University
State Grid Corporation of China

Introduction

This introduction is not part of IEEE Std 1935-2023, IEEE Standard for Edge/Fog Manageability and Orchestration.

With the innovation of mobile applications and the arrival of the fifth generation of telecommunication, edge computing has become a popular scheme due to its geographical proximity to end users. The overall architecture has the advantage of lower latency and better user experience. However, because of physical limitations, management and orchestration in the edge system is necessary in order to maintain operations, common functions, and application lifecycle. Accordingly, the IEEE P1935 working group introduced a standardized, orchestration-wise design to simplify the process and offer better system performance and user experience. This standard involves the framework, architecture, and procedures of the edge/fog system as well as its application lifecycle management.

Contents

List of Figures

List of Tables

IEEE Standard for Edge/Fog Manageability and Orchestration

1. Overview

1.1 Scope

This standard specifies management and orchestration framework and mechanisms for edge/fog computing systems. It describes the edge/fog platform management procedures and edge/fog application management procedures.

1.2 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).[6,7]

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

1.3 Introduction

Edge computing is an emerging technology that can host the mobile applications closer to its users, provides lower latency, more efficient bandwidth and service delivery, as well as better user experience quality. Innovative mobile applications, such as augmented reality, facial detection, and interactive applications, evolve with mobile devices and attract great attention due to their ability to bring convenience and spice up people's lives. With a core concept similar to edge computing's placing the computing capacity at the local area network, fog computing is more often used in industrial internet of things (IIoT) scenarios. Edge/fog computing is a brand-new and promising paradigm offering an environment characterized by low latency and necessary resources for mobile devices to liberate them from computing-intensive and real-time applications.

[6]The use of the word *must* is deprecated and cannot be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.
[7]The use of *will* is deprecated and cannot be used when stating mandatory requirements; *will* is only used in statements of fact.

Although edge devices are closer to their users to provide the advantages described above, their reduced capacities of computation and storage compared to cloud computing is a fatal issue in practice. Edge/fog systems are responsible for maintaining common operations, handling application lifecycle, providing a corresponding environment for mobile services, performing storage and traffic control, and supporting service mobility. To meet these requirements, the management and orchestration of the edge/fog computing system become key topics in modern scenarios of networking and service delivery, aiming to leverage constrained resources more efficiently and coordinately.

The management and orchestration mentioned in this standard indicate the manual and automated configuration, control, and coordination of the edge/fog system and its services. The unified management and orchestration—such as regulating message transmission, controlling and allocating resources of the edge devices, and monitoring and arranging the mobile application lifecycle—can satisfy the mentioned needs and ensure better availability, flexibility, reliability, scalability, stability, service mobility, and performance. It means that the system shall be easy for multiple types of users to operate and configure, reliable while facing different challenges, and as efficient as possible. The edge/fog system—especially the orchestrator level in the three-level architecture that will be explained in later clauses—contains multiple entities and components responsible for executing the related management mechanism and coordinating the interaction among them. With a comprehensive management and orchestration scheme on top of it, the edge/fog system can handle multiple edge applications with various customers and scenarios.

To achieve the flexibility, scalability, security, and ability to manage its services and resources, it is necessary for an edge/fog system to follow the related specification consistently. Aside from the regular concepts and requirements, the edge/fog system and its components, as well as the management and orchestration designs, need to fulfill the demands and requirements listed below to achieve practicality:

a) It shall be possible to deploy the edge/fog system in various positions in the wired, wireless, and mobile network, as well as the data center of the edge service operators and edge service providers.
b) It shall be possible to deploy the components, including the edge/fog orchestrator (EFO), edge/fog control nodes, and edge/fog compute nodes, on the various hardware devices.
c) The edge/fog system shall fit in and communicate with the wireless communication network, dealing with the corresponding traffic routing to the correct components.
d) The edge/fog system shall be able to provide edge services and edge applications, as well as the corresponding environment for them.
e) The edge/fog system shall support the communications between the components and edge services and edge applications if authorized. This may include the different applications on different edge/fog compute nodes.
f) The edge/fog system shall support the full lifecycle and related operations of edge applications, such as hosting, onboarding, instantiation, operating, and termination.
g) The edge/fog system shall support edge service operators to manually and dynamically access the edge services and edge applications on their needs.
h) The edge/fog system shall be able to decide the service homing.
i) The edge/fog system shall support the compatibility with the authorized third-party entities, including processing the corresponding requests.
j) It shall be possible to deploy edge applications on different edge/fog compute nodes without specific arrangement.
k) The edge/fog system shall be able to authorize and authenticate the user requests about the edge applications.

l) The edge/fog system shall support the service mobility, that is, keeping connectivity while the user equipment (UE) is moving from the coverage of one edge/fog control node to another.

m) The edge/fog system shall utilize the related network information to improve its own performance in various matrices.

n) The edge/fog system shall support the edge service to declare its own availability during service discovery.

o) The edge/fog compute nodes shall be able to provide computing, networking, and storage resources.

This standard specifies the related components, requirements, procedures, and other necessary parts of management and orchestration in the edge/fog system. The following clauses illustrate and describe these specifications in detail. The framework and elements of the edge/fog system, as well as their functionalities and communication, are introduced in Clause 3. The ways to manage and orchestrate the platforms and related resources in edge/fog system are explained in Clause 4. The procedures and necessary formal files involved in launching and deploying the services and applications on the edge/fog system are elaborated in Clause 5. The concept of communication among multiple edge/fog systems are described in Clause 6.

2. Definitions, acronyms, and abbreviations

2.1 Definitions

For the purposes of this document, the following terms and definitions apply. The IEEE Standards Dictionary Online should be consulted for terms not defined in this clause.[8]

application context: Application context, or simply *context,* involves all the files that contain the custom information operators need, such as user information and statistical data.

application management element: The entity managing and coordinating the operation of edge applications, such as edge platform.

application package: One of the files that is involved in the application onboarding procedures. The application package is hierarchical, containing the necessary data and information of the application and its components, like virtual function (VF) images. An application package can be considered an installation file that will be deployed to a compute node to launch a specific edge application.

artifact: All the files that are generated by the edge app designer component and are distributed to the other components in the edge/fog orchestrator (EFO). They are the descriptor files for the components in the EFO.

attribute: The additional information of the resource.

blueprint file: A file that includes the orchestration rules, resource lifecycle management designs, etc. This file is used to configure the components in the edge/fog orchestrator (EFO).

compute node: One entity in the computer level that can handle the computing tasks on its own.

computer level: the bottom layer of the edge system that is composed of one or more compute nodes dealing with the computing tasks.

context: *See:* **application context**.

[8]IEEE Standards Dictionary Online is available at: http://dictionary.ieee.org. An IEEE account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

controller level: The middle layer of the edge system that is composed of one or more control nodes responsible for management and coordination of computer level.

control node: One entity in the controller level for management of the related compute nodes.

data plane: The functional entity that handles the traffic routes.

edge app: The applications running on the edge system.

edge app designer: A component in the edge/fog orchestrator (EFO) handling the application design and onboarding.

edge inventory manager: A component in the edge/fog orchestrator (EFO) for the management of the inventory in the edge system.

edge platform manager: An entity managing the edge platform.

edge platform: A platform that handles purposed computing tasks.

edge resource: The resources created in the edge system. This resource can be further sliced as the resource quota.

edge service operator: The service operator dealing with the management and operation of edge applications.

edge service provider: The designer and onboarder of edge applications.

edge/fog orchestrator (EFO): An orchestrator of the whole edge system. *Syn:* **edge orchestrator**.

EFO controllers: The components in the edge/fog orchestrator (EFO) responsible for coordinating and configuration of virtual network functions and related resources.

end user app: An interface for all end users.

end user proxy: A component in the edge/fog orchestrator (EFO) as an interface between the EFO and the end user app.

end users: All users that access the edge/fog orchestrator (EFO) via the end user apps.

external API: In the edge/fog orchestrator (EFO), it is provided for the edge service operators or other entities to access the functionalities of the EFO.

filtering rule: The restrictions and conditions to narrow down the search results.

infrastructure manager (IM): *See:* **virtualization infrastructure manager**.

inventory: The network information, topology, and metadata that are about the edge system environment and are stored in the edge inventory manager component.

lifecycle manager enabler: A component in the edge/fog orchestrator (EFO) that is responsible for the management of the application lifecycle.

manifest file: The files used in the application onboarding procedures containing an application package and a blueprint file.

network infrastructure: *See:* **virtual network infrastructure**.

orchestrator level: The top layer of the edge system for management and orchestration of the whole system, including controller level and computer level.

physical resource: The physical hardware of the edge system, for example, the physical servers.

property: The characteristics of the resource.

resource configuration: A part at the end of the resource creation operation.

resource creation: The operation used to create new resources in the edge system.

resource deletion: The operation used to delete resources in the edge system.

resource discovery: The operation used to search and return the list of resources matching the filtering rules.

resource management element: The entity managing the resources in the edge system, such as infrastructure manager/virtualization infrastructure manager (IM/VIM).

resource quota: After resource creation, the edge resources will be created and configured. In the configuration process or the resource reconfiguration operation later, the edge resource can be sliced into resource quota.

resource reconfiguration: The operation used to modify the information of resources. There are two types of this operation: resource replacement and resource update.

resource replacement: One of two types of modification operation that replaces whole information of the resource.

resource status query: The operation used to get the current information of the target resource.

resource update: One of two types of modification operation that only updates partial information of the resource.

resource URL (uniform resource locator): The unique addresses of the resources for the edge system to locate.

resource: The physical and virtual assets that can be leveraged by the edge system.

rule framework: A component in the edge/fog orchestrator (EFO) dealing with all conditions, requirements, and reaction policies.

VF (virtual function) image: The virtual functions providing an executing environment for the edge application. An application package is composed of one or more virtual functions called *VF images*.

virtual function management and orchestration (VF M&O): A component in the edge/fog orchestrator (EFO) responsible for the start-up and operation of the application.

virtual network infrastructure (VNI): The virtualized and non-virtualized resources that enable network operations.

virtual resource: The non-physical resources in the edge system, including computing resources, networking resources, storage resources, etc.

virtualization infrastructure manager (VIM): An entity managing the infrastructure resources named virtual network infrastructure that can be virtualized or non-virtualized.

2.2 Acronyms and abbreviations

AAA	authentication, authorization, and accounting
AME	application management element
CID	connectivity/interoperability domain
EAD	edge app designer
EFO	edge/fog orchestrator
EIM	edge inventory manager
EUA	end user application
EUP	end user proxy
IM	infrastructure manager
LCM	lifecycle manager
RME	resource management element
SAD	service and application domain
SLA	service level agreement
URL	uniform resource locator
VF	virtual function
VF M&O	virtual function management and orchestration
VIM	virtualization infrastructure manager
VNI	virtual network infrastructure
VNF	virtual network function

3. Architecture and framework for edge/fog manageability and orchestration

3.1 Introduction

In the edge/fog system, there are three levels responsible for different functionality and various entities in each level. This clause describes the overall edge/fog architecture and the responsibility of each entity and component. The interfaces between each entity are presented as well.

3.2 Edge/fog manageability and orchestration framework overview

Figure 1 is the basic architecture of the edge/fog system framework. The term *framework* here refers to the full architecture of the system and the components and entities in it. The entities refer to the elements, either inside the system or outside the system (e.g., some third-party software), and may be composed of one or more components. The system can be classified into three levels: orchestrator level, controller level, and computer level, and the more detailed information of the components introduced here can be found in 3.3. The orchestrator level includes the edge/fog orchestrator (EFO) as the main management and orchestration entity, and it is responsible for interacting with "users," that is, end users, edge service providers, and edge service operators. The end users can call all the users accessing the system via the end user app, which is another entity in the edge/fog system. The edge service providers are the designers and onboarders of edge applications. The edge service operators provide the edge/fog system functionalities and infrastructures and are responsible for managing and operating these applications properly. The controller level may include one or more edge/fog control nodes, which oversee the edge/fog compute nodes and manages the related resources. The edge platform manager and the (virtualization) infrastructure manager are at this level. The computer level may include one or more edge/fog compute nodes as well, which are responsible for the practical computing tasks. The edge platform, the edge application, and the (virtual) network infrastructure are at this level. The edge/fog compute nodes, as the most common and the most widely distributed entity in the system, can also be called *edge nodes* as an abbreviation.[9] All these entities and components will be described in detail in the following clauses.

Table 1 gives some brief descriptions and the related clause to each entity mentioned in Figure 1, as well as the terms used in Clause 3.

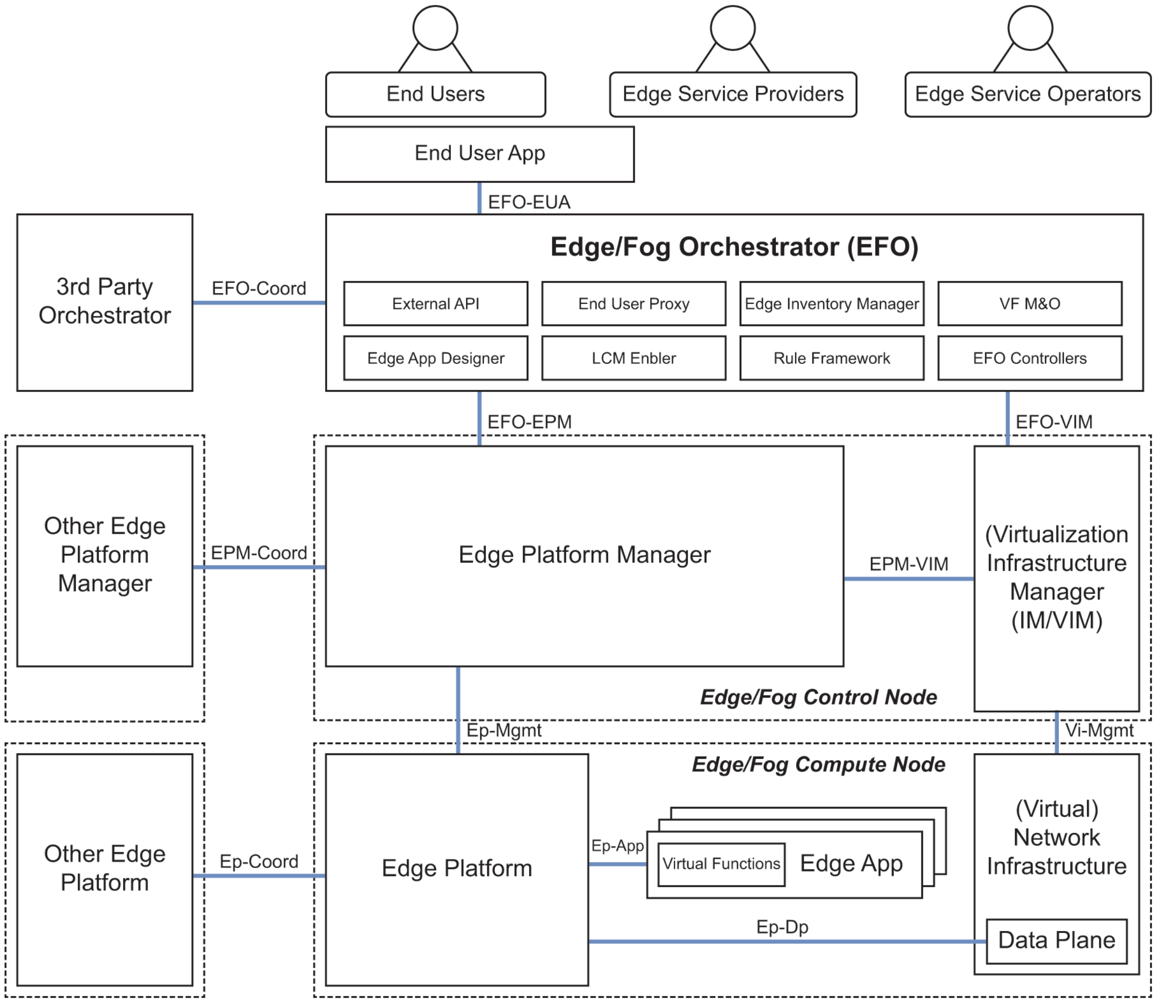

Figure 1—The basic architecture of the edge/fog system framework

[9]For ease of reading and description, we abbreviate the term *edge/fog* to *edge* in the remaining clauses of this standard. These terms refer to completely the same things. For example, an *edge/fog control node* can also be written as an *edge control node*.

Table 1—A brief description of the entities in Figure 1 and Clause 3

Entity	Brief description	Subclause
(Virtual) network infrastructure	The virtualized and non-virtualized resources that enable network operations. Usually abbreviated as VNI.	3.5.5
(Virtualization) infrastructure manager	An entity managing the infrastructure resources named virtual network infrastructure that can be virtualized or non-virtualized.	3.4.2
Application management element	The entity managing and coordinating the operation of edge applications, such as edge platform.	3.4.3
Compute node	Compute node is one entity in the computer level that is able to handle the computing tasks on its own.	3.2
Computer level	Computer level is the bottom layer of the edge system that is composed of one or more compute nodes dealing with the computing tasks.	3.2
Control level	Control level is the middle layer of the edge system that is composed of one or more control nodes, responsible for management and coordination of the computer level.	3.2
Control node	Control node is one entity in the controller level for management of the related compute nodes.	3.2
Data plane	The functional entity that handles the traffic routes.	3.5.4
Edge app	The applications running on the edge system.	3.5.3
Edge app designer	A component in the EFO handling the application design and onboarding.	3.3.2
Edge inventory manager	A component in the EFO for the management of the inventory in the edge system.	3.3.2
Edge platform	A platform that handles purposed computing tasks.	3.5.2
Edge platform manager	An entity managing the edge platform.	3.4.3
Edge service operator	The service operator dealing with the management and operation of edge applications.	3.2
Edge service provider	The designer and onboarder of edge applications.	3.2
Edge/fog orchestrator	An orchestrator of the whole edge system. It can be abbreviated as *edge orchestrator* or just *EFO*.	3.3.2
EFO controllers	The components in the EFO responsible for coordinating and configuration of virtual network functions (VNFs) and related resources.	3.3.2
End user app	An interface for all end users.	3.6.2
End user proxy	A component in the EFO as an interface between the EFO and the end user app.	3.3.2
End users	The users accessing the system via the end user apps.	3.6.2
End users	All users that access the EFO via the end user apps.	3.2
External API	External API in the EFO is provided for the edge service operators or other entities to access the functionalities of the EFO.	3.3.2
Inventory	Inventory is the network information, topology, and metadata that are about the edge system environment and are stored in the edge inventory manager component.	3.3.2
Lifecycle manager enabler (LCM enabler)	A component in the EFO that is responsible for the management of the application lifecycle.	3.3.2
Orchestrator level	The top layer of the edge system for management and orchestration of the whole system, including controller level and computer level.	3.2

Table continues

Table 1—A brief description of the entities in Figure 1 and Clause 3 *(continued)*

Entity	Brief description	Subclause
Resource management element	The entity managing the resources in the edge system, such as IM/VIM.	3.4.2
Rule framework	A component in the EFO dealing with all conditions, requirements, and reaction policies.	3.3.2
Virtual function management and orchestration	A component in the EFO responsible for the start-up and operation of the application.	3.3.2

3.3 Edge orchestrator level entities

3.3.1 Introduction

The edge/fog orchestrator (EFO) level can also be abbreviated as the edge orchestrator level. The entities and components in this level are responsible for the management and control of the whole edge system, in order to orchestrate and automate the network service running on the edge system. The edge orchestrator is the core functionality in orchestrator level management. There are multiple ways to access the EFO, detailed in the following subclause.

3.3.2 Edge orchestrator

The edge/fog orchestrator (EFO), or just edge orchestrator, is designed to provide real-time and rule-driven service orchestration and automation, including the start-up and configuration of virtual/physical network functions. The edge orchestrator allows various providers and developers involved in different fields—such as network, software, and service—to rapidly automate new service and support complete lifecycle management. With a common architecture as the base, the platform will help promote the development of the service.

The edge and fog orchestrator can be generally divided into three sections: the design time framework, the runtime framework, and the user interface. Each section contains one or more components that take different responsibilities.

The design time framework will be a comprehensive development environment with related tools, techniques, and repositories for defining and describing resources, services, and products.

— The edge app designer component in the platform will provide the functionality mentioned above to define/simulate/certify system assets as well as their associated processes and polices.
— Lifecycle management (LCM) enabler component is responsible for designing and managing the application lifecycle.

The runtime framework executes the rules and polices distributed by the design and creation environment, the edge app designer component in this case, and controllers manage resources corresponding to their assigned controlled domain.

— Virtual function management and orchestration (VF M&O) component will follow the business process model and notation (BPMN) flows that operate on the models distributed from edge app designer. VF M&O will use the models that describe the services and associated VNFs and other resources to automate the sequences of associated components needed for on-demand creation, modification, or removal of network, application, or infrastructure services and resources. VF M&O is able to drive any edge platform.
— Rule framework is responsible for dealing with all conditions, requirements, constraints, attributes, and needs that must be provided, maintained, and/or enforced.

- Edge inventory manager (EIM) provides real-time view of a system's various inventory, including resources, services, products, and their relationships with each other.
- End user proxy (EUP) is responsible for interacting with end user applications. It receives requests from the end user applications. The EUP allows end user applications to request onboarding, instantiation, and termination of the edge application. The EUP authorizes requests from end user applications and interacts with the other components in the EFO for further processing.
- The EFO controllers are responsible for network configuration for edge computing resources and network, VNF configurations and lifecycle management operations, and the lifecycle management of VNFs and network services based on VNF using VNF manager.

Finally, the user interface is for various platform users to design, operate, and manage the services and applications on the platform. The platform users have different access and operations allowed according to their assigned roles. In addition to the graphical user interfaces, external API are also provided to access the core network and use external user applications to execute specific operations and access the functionalities of the EFO.

3.4 Edge controller level entities

3.4.1 Introduction

The entities located in the edge controller level are responsible for the management and orchestration of the edge compute nodes. The entities can be generally divided into two classes according to their management targets: Resource management elements (RMEs) and application management elements (AMEs). The function and responsibility of each class of entities is described in the following subclauses.

3.4.2 Resource management elements

3.4.2.1 Introduction

Resource management elements focus on management and storage of network infrastructure. They can control and schedule the usage of the resources in the edge system. In the edge system, the resources can be classified into several categories, such as virtual or physical. More information about the resources in the edge system can be found in Clause 4.

3.4.2.2 Infrastructure manager

The infrastructure manager (IM) is responsible for managing both the virtualized infrastructure and the non-virtualized infrastructure as well as its virtualized and non-virtualized resources, including resource allocation, infrastructure preparation, and system information report. In many deployment scenarios, the computing infrastructure managers are implemented in virtual ways, such as virtual machines or containers. The management entity is usually called a *virtualization infrastructure manager* (abbreviated as VIM). It is termed IM/VIM in this document to emphasize its nature to support virtualized environments. Throughout the document, IM/VIM is used to describe the entity that manages the underlying computing infrastructure in a generic way. The implementation of the IM/VIM could be virtualized or non-virtualized according to the deployment scenarios.

3.4.3 Application management elements

3.4.3.1 Introduction

Application management elements focus on lifecycle management of the edge platform and the edge applications running on the platform.

3.4.3.2 Edge platform manager

The edge platform manager manages the rules, requirements, and lifecycle of applications, and provides element management functions to the edge platform. The edge platform manager receives virtualized resource fault reports and performance measurements from the IM/VIM for further processing.

Edge platform manager also includes the functional blocks that are responsible for the management of the edge platform and the edge applications with standard LCM procedures. edge application instances are considered VNF instances. It is possible to deploy more than one edge application LCM instance.

3.5 Edge computer level entities

3.5.1 Introduction

The edge computer entities are the functional elements that are involved in the computing tasks in an edge host. It includes the edge platform and a virtualization infrastructure that provides computational, storage, and network resources for the edge applications.

3.5.2 Edge platform

The edge platform is responsible for the edge functions that are necessary to run edge applications. It serves as a platform where edge applications are able to provide, discover, and consume certain edge services.

The edge platform also instructs data plane, which is further described in 3.5.4, and configures domain name system (DNS) proxy/server according to the traffic rules prescribed by the edge platform manager.

3.5.3 Edge applications

Edge applications are executed as virtual machines on top of the virtualization infrastructure provided by the edge host and can interact with the edge platform to consume and provide edge services.

An edge application is composed of one or multiple virtual function(s) that can be managed and orchestrated by the VF M&O component in EFO. It is possible either for one edge app to provide several services, or for multiple edge apps to provide a single service.

3.5.4 Data plane

The data plane in the edge host is responsible for dealing with the traffic rules, including routing the traffic among entities, as well as task offloading and installation of application from IM/VIM.

3.5.5 Virtual network infrastructure

The virtual network infrastructure (VNI) is the totality of all hardware and software that build up the entire environment where VNFs are deployed. The virtualization infrastructure is deployed as an VNI and is managed by an IM/VIM.

3.6 External entities

3.6.1 Introduction

This subclause describes the elements that connect the external environment. Explanations of how the edge architecture communicates with other existing standards are presented.

3.6.2 End user application

End user applications (EUAs) can interact with the end user app LCM component in EFO in orchestrator level via the standard API. The users—including end users, edge service operators, and edge service providers—can access the EFO services through the EUA.

3.7 Interfaces

3.7.1 Introduction

This subclause lists and describes the interfaces and reference points between entities inside and outside the edge system. The interfaces can be classified based on the highest level of the connected entities. That is, an interface between orchestrator level and controller level is classified as an orchestrator interface.

3.7.2 Orchestrator interfaces

The orchestrator interfaces encompass the connections between the orchestrator level and the controller level. These interfaces serve as crucial connectors, enabling the seamless flow of information and command execution for the EFO.

— The EFO-EPM reference point between the EFO and the edge platform manager is used for application management, including application lifecycle, rules and requirements, checking edge service availability, and exchanging related notifications between the EFO and the edge platform.

— The EFO-VIM reference point between the EFO and the IM/VIM is used for management of virtualized and non-virtualized resources in the edge system.

— The EFO-EUA reference point between the EFO and an end user app (EUA) is used to enable the runtime configuration to satisfy a set of quality of service (QoS) policies on workload, resource usage, or network performance. It also supports the operation of necessary files, such as manifest files and packages.

— The EFO-Coord reference point between the EFO and a third-party orchestrator is used to exchange necessary information about the underlying edge nodes, the implemented rules, and the approach to interpret the messages.

Table 2—The related subclauses of the orchestrator interfaces

Interface	Related subclauses	Responsibility
EFO-EPM	5.3.3	Service deploy requests, workload instantiation, application activation, inventory update
	5.3.4, 5.3.5	Context creation requests and responses, context deletion requests and responses
	5.3.6	Service termination requests, workload termination, application deactivation, inventory deletion
	5.3.7	Application reconfiguration, inventory update
EFO-VIM	4.3.2 through 4.3.6	Resource requests and responses (including resource creation, resource status query, resource discovery, resource reconfiguration, resource deletion)
	5.3.2	Store the images
	5.3.3	Resource allocation, infrastructure instantiation
	5.3.6	Resource release

Table continues

Table 2—The related subclauses of the orchestrator interfaces *(continued)*

Interface	Related subclauses	Responsibility
EFO-EUA	5.3.2	Manifest file onboarding and queries, package distribution requests, notification
	5.3.4	Application list queries, context creation requests
	5.3.2 through 5.3.7	Application requests (including onboarding, instantiation, termination, and reconfiguration) and context requests (including context creation and deletion)
EFO-Coord	6.3	Communication between two EFOs

3.7.3 Controller interfaces

Within this subclause, we will delve into the controller interfaces that encompass both the interfaces within controller level and those between controller level and computer level. These interfaces serve as the conduits for exchanging vital information and facilitating the seamless coordination of edge applications.

— The Ep-Mgmt reference point between the edge platform manager and the edge platform is used for the configuration of edge platform, including the rules, requirements, and lifecycle support of applications.

— The Vi-Mgmt reference point between the IM/VIM and the VNI is used for the management and coordination of the VNI and the data plane in the edge system.

— The EPM-VIM reference point between the edge platform manager and the IM/VIM is responsible for the communication between these two entities.

— The EPM-Coord reference point between two edge platform managers is used to exchange the necessary control information about the underlying edge compute nodes. The interoperation of two EPMs belonging to different edge service operators can also leverage this interface to share their information.

Table 3—The related subclauses of the controller interfaces

Interface	Related subclauses	Responsibility
Ep-Mgmt	5.3.3	Application creation, configuration requests, inventory update
	5.3.4	Context creation requests and responses
	5.3.5	Context deletion requests and responses
	5.3.6	Configuration requests, application deletion, inventory deletion
	5.3.7	Application reconfiguration
Vi-Mgmt	4.3.2	Resource creation
	4.3.5	Resource replacement, resource update
	4.3.6	Resource deletion
	5.3.2	Store the images
	5.3.3	Assign application packages
	5.3.6	Infrastructure termination
EPM-VIM	5.3.6	Infrastructure termination requests
	5.3.7	Infrastructure update requests
EPM-Coord	6.3	Communication between two EPMs

3.7.4 Computer interfaces

In this subclause, we will explore the computer interfaces that primarily encompass the interfaces within the computer level. These interfaces serve as essential communication channels utilized by the edge platform to interact with various entities, including edge applications, data plane, and even other edge platforms.

— The Ep-App reference point between the edge platform and the edge applications provides the registration, discovery, and communication support of services.

— The Ep-Dp reference point between the edge platform and the data plane of the virtual network infrastructure is used to set the data plane on how to route traffic among applications, network, service, etc.

— The Ep-Coord reference point between two edge platforms is used to exchange necessary information or perform the handover during the operation of applications. The interoperation of two edge platforms belonging to different edge service operators can also leverage this interface to transmit messages.

Table 4—The related subclauses of the computer interfaces

Interface	Related subclauses	Responsibility
Ep-App	5.3.3 through 5.3.7	Application creation and deletion, configuration, context creation and deletion
Ep-Dp	5.3.3	Assign application packages It is also involved in some pre-advance preparation, such as images onboarding, application scaling, and container installation
Ep-Coord	6.3	Communication between two edge platforms

4. Edge platform management and orchestration

4.1 Introduction

The edge platforms are responsible for handling various tasks, including computing, networking, and storage. In order to process these tasks in a coordinated and scalable manner, the platforms need to communicate with other entities to share necessary information and collect needed resources. Aside from the edge platforms, the edge and fog orchestrator (EFO) and the edge platform managers are also involved in the process. This clause describes the RESTful service APIs of the resource operations that are used for the communication of these components.

4.2 Overview of edge platform resource management

The service APIs are used for the resource management and communication between the components. As shown in Figure 2, these service APIs can be used to do several common operations on the resources, including resource creation, resource status query, resource search, resource reconfiguration, and resource deletion. The resources we mention here include physical resources and virtual resources. The physical resources are the hardware equipment available for the edge system, including physical servers, facilities, and infrastructure. The virtual resources can be considered as the software, computing, networking, and storage resources in the system, as well as the virtualized infrastructure.

There are five types of API supported:

a) Resource creation: This operation API is used to create the target resource in the edge system. It includes registration and configuration of both physical resources and virtual resources. The HTTP method POST is supported.

b) Resource status query: This operation API is used to fetch the information of the certain resource. The HTTP method GET is supported.

c) Resource discovery: This operation API is used to search the resources that meet the specific filtering rules. The HTTP method GET is supported.

d) Resource reconfiguration: This operation API is used to modify the properties and attributes of the target resource after its creation. There are two types of HTTP methods supported, PUT and PATCH, that are used to replace the whole information and to update partial information, respectively.

e) Resource deletion: This operation API is used to remove the target resource in the edge system. The HTTP method DELETE is supported.

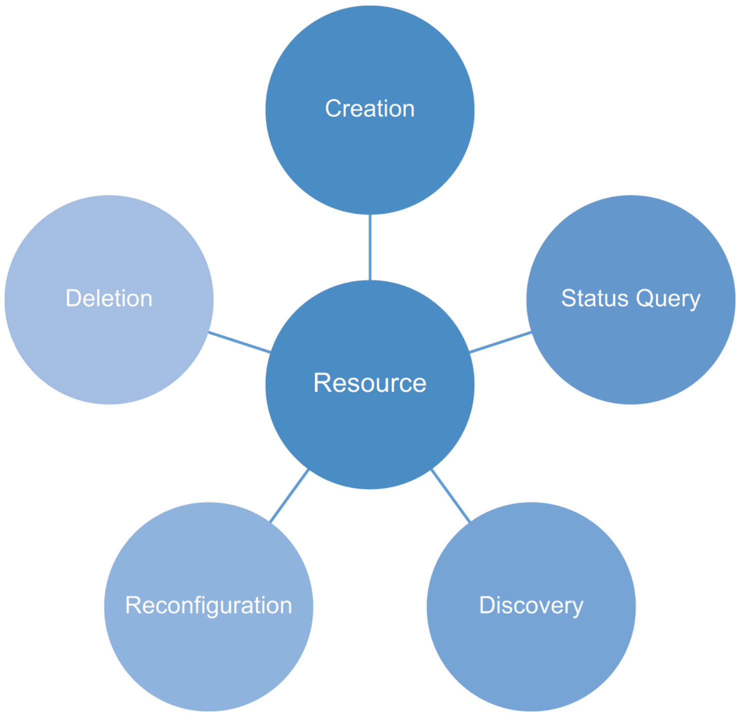

Figure 2—The common operations of the edge system resource management

These service APIs for the edge platforms are RESTful HTTP APIs, which shall be independent of technology implementation. All the resources in the edge system have their uniform resource locators (URLs) to be called by the APIs. The specification of service APIs shall include the following information:

— The purpose of the API: The description and usage of the operation API.

— Request format: The format of the request. The format includes the HTTP methods, headers, and payload.

— Response format: The format of the response, including the supported HTTP status codes.

— HTTP methods supported: The HTTP method(s) used in the operation API. An operation API may support multiple HTTP methods.

— Representation supported: Most of the operation API payload should be in JSON format. The exceptions shall be specified, if they exist.

The components mentioned in 4.3 are listed in Table 5. The operation APIs can be sent from the internal VF M&O. It may also be possible for some external API clients that are able to send HTTP requests to send the requests, but additional verification and authorization are needed. This clause only discusses the situation where the request is sent from the VF M&O.

Table 5—The components mentioned in 4.3

Types	Components	4.3.2	4.3.3	4.3.4	4.3.5	4.3.6
Users	Service provider					
	Operator					
	End user app					
EFO	External API					
	End user proxy					
	Edge app designer					
	VF M&O	√	√	√	√	√
	Edge inventory manager					
	LCM enabler					
	Rule framework					
	EFO controllers					
Control nodes	Edge platform manager					
	IM/VIM	√	√	√	√	√
Compute nodes	Edge platform					
	Edge app					
	VNI	√				√
	Data plane					

Table 6—The terms used in the operation APIs

Terms	Descriptions	Subclause
Attribute	Attributes are the additional information of the resource.	4.3.5
Edge resource	Edge resources are the resources created in the edge system. This resource can be further sliced as the resource quota.	4.3.2
Filtering rule	Filtering rules are the restrictions and conditions to narrow down the search results.	4.3.4
Physical resource	Physical resources are the physical hardware of the edge system, for example, the physical servers.	4.2
Property	Property means the characteristics of the resource.	4.3.5
Resource	Resources are the physical and virtual assets that can be leveraged by the edge system.	4.2
Resource configuration	The resource configuration is a part at the end of the resource creation operation.	4.3.2
Resource creation	It is the operation used to create new resources in the edge system.	4.3.2
Resource deletion	It is the operation used to delete resources in the edge system.	4.3.6
Resource discovery	It is the operation used to search and return the list of resources matching the filtering rules.	4.3.4
Resource quota	After resource creation, the edge resources will be created and configured. In the configuration process, or later in the resource reconfiguration operation, the edge resource can be sliced into resource quota.	4.3.2
Resource reconfiguration	It is the operation used to modify the information of resources. There are two types of this operation: resource replacement and resource update.	4.3.5

Table continues

Table 6—The terms used in the operation APIs *(continued)*

Terms	Descriptions	Subclause
Resource replacement	It is one of the two types of modification operation which replaces whole information of the resource.	4.3.5
Resource status query	It is the operation used to get the current information of the target resource.	4.3.3
Resource update	It is one of two types of modification operation that only updates partial information of the resource.	4.3.5
Resource URL	Resource URLs are the unique addresses of the resources for the edge system to locate.	4.2
Virtual resource	Virtual resources are the non-physical resources in the edge system, including computing resources, networking resources, storage resources, etc.	4.2

4.3 Resource manageability procedures

4.3.1 Introduction

This subclause describes the service APIs that can be used to do several resource operations, including resource creation, status query, discovery, reconfiguration, and deletion. These resources need to be registered by the edge service operators or edge service providers via the resource creation operation before being used. The request procedures start from the VF M&O in the EFO and are processed by the IM/VIM in most cases. Subclauses 4.3.2 through 4.3.6 describe the usage, procedures, as well as the format of the requests and responses of each operation. Subclause 4.3.7 lists the corresponding HTTP status code for the failed requests.

4.3.2 Resource creation

4.3.2.1 Introduction

New resources can be created by sending an HTTP POST request to the system. The POST request shall contain the information about the new resource that will be created. The new resources called *edge resources* will be given unique identifiers (IDs) named resource URLs. After creation, these edge resources will be configured automatically based on the request payload. The resource may be split to several pieces or slices, called *resource quota*, providing higher flexibility and system utilization. For example, the virtual resources contained in one physical server can be cut into slices to be leveraged in different fields, and these slices are called the *resource quota of the physical resource* (that physical server). The other details of the configuration process vary from system to system and are therefore beyond the scope of this standard.

Such resource creation operation can be made for both physical and virtual resources. It is also the necessary procedure for the edge service operators and edge service providers to register their resources into the system, or the system will not be able to recognize them and perform other operations. The resource creation operation registers the real information of the physical resources to the edge system to make them visible and usable for the system and connects the virtual resources to the physical resources that host them. In such connections, the virtual resources are like the child resources of the physical resources, which are like parent resources. The connections can be modified by the resource reconfiguration operation.

VF M&O seldom makes the resource creation operation on its own, but it is possible for it to create and connect the virtual resources to the physical resources automatically after the creation of the physical resources.

4.3.2.2 Conditions and information involved

The pre-condition of the procedure:

— The target resource is not yet created in the edge system.

The post-condition of the procedure:

— The resource has been created and configured as network infrastructure in the edge system.

4.3.2.3 Procedures

The procedures outlined in this subclause detail the steps involved in resource creation as depicted in Figure 3. The procedures are as follows:

a) The VF M&O sends a resource creation request to the IM/VIM (Table 7).

b) The IM/VIM computes the resource URL and then creates the resource as the network infrastructure. The IM/VIM configures the properties and attributes of the resource based on the payload of the request, the process of which differs from system to system.

c) After the resource is created either successfully or failingly, the IM/VIM sends the response to the VF M&O (Table 8 and Table 20).

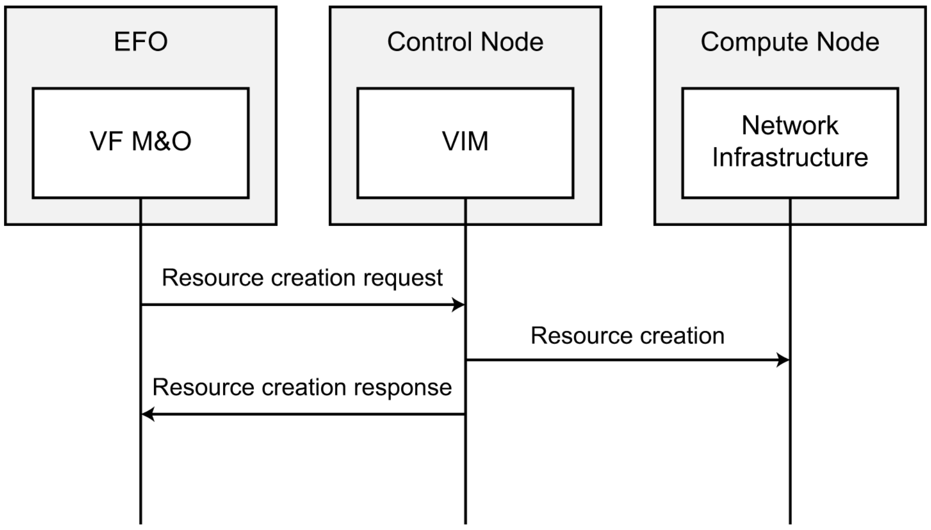

Figure 3—The workflow of the resource creation

4.3.2.4 Requests and responses

If the server creates the new resource successfully, the HTTP response status "201 created" shall be returned, which includes the path information of the created resource (Table 7). On failure, the specific status error code shall be returned (Table 8).

Table 7—Resource creation request

Element	Description
Request type	HTTP POST
Parent resource ID	The identifier of the parent resource of the created resource It shall be contained in the URL
Request payload	The payload contains the information and data of the created resource

Table 8—Resource creation successful response

Element	Description
HTTP status code	201 Created
HTTP header	The URL of the created resource
Response payload	(Empty)

4.3.3 Resource status query

4.3.3.1 Introduction

The resource information can be obtained via an HTTP GET request that ought to be available for most resources and contains the empty payload. This operation is usually used when the entity already knows the URL of the target resource, which can be obtained via the resource discovery operation described in 4.3.4. The resource status query operation can be used for both physical and virtual resources. In all cases, this operation only returns the information of the target resource, and it will give corresponding error messages if the requested resource is not found in the system. This operation is also the necessary step before the resource reconfiguration operation described in 4.3.5 to avoid race condition.

4.3.3.2 Conditions and information involved

The pre-condition of the procedure:

— The target resource has been created in the edge system.

The post-condition of the procedure:

— The entity that sent the request gets the information of the target resource.

4.3.3.3 Procedures

The procedures outlined in this subclause detail the steps involved in resource status query as depicted in Figure 4. The procedures are as follows:

a) The VF M&O sends the resource status query request to the IM/VIM in order to query the target resource.
b) If the specific resource is found, the information of the requested resource shall be returned to the VF M&O as an HTTP response (Table 10). On failure, the specific error status code shall be returned (Table 20).

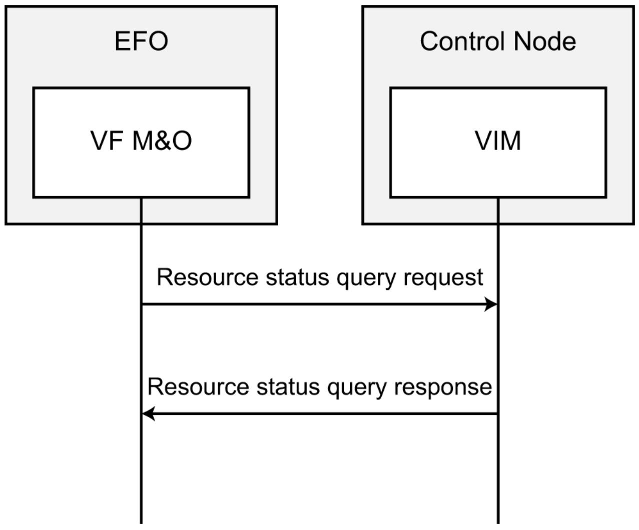

Figure 4—The workflow of the resource status query

4.3.3.4 Requests and responses

On success, the "200 OK" status code shall be returned with information of the requested resource (Table 10); on failure, the specific error status code shall be returned (Table 20).

Table 9—Resource status query request

Element	Description
Request type	HTTP GET
Resource ID	The identifier of the target resource It shall be contained in the URL
Request payload	(Empty)

Table 10—Resource status query successful response

Element	Description
HTTP status code	200 OK
HTTP header	(Empty)
Response payload	The information of the requested resource, including the version number of it

4.3.4 Resource discovery

4.3.4.1 Introduction

The resource searching with the filtering rules can be done by an HTTP GET method with query parameters that can be used to control the content of the query result. The filtering rules are the restrictions and conditions to narrow down the search results. The GET request is with the empty payload. Compared to the resource status query operation, the resource discovery operation can return all the resources that match the filtering rules. If there is no resource that matches the filtering rules, an empty list shall be returned.

As with the other operations, the resource discovery operation is suitable for both physical and virtual resources. The users and the system can use this operation to find all the resources they need via the VF M&O respectively.

4.3.4.2 Conditions and involved information

The pre-condition of the procedures:

— The target resource has been created in the edge system.

The post-condition of the procedures:

— The entity that sent the request receives all the resources matching the filtering rules as well as their information.

4.3.4.3 Procedures

The procedures outlined in this subclause detail the steps involved in resource discovery as depicted in Figure 5. The procedures are as follows:

a) The VF M&O sends a resource discovery request to the IM/VIM (Table 11).
b) The IM/VIM searches and filters the resources to return a list of resources that satisfies the given filtering rules as an HTTP response to the VF M&O (Table 12). On failure, the specific error status code shall be returned (Table 20).

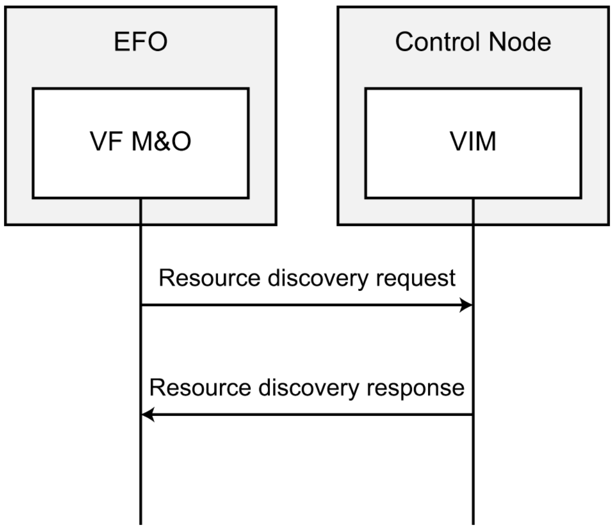

Figure 5—The workflow of the resource discovery

4.3.4.4 Requests and responses

On success, the "200 OK" status code shall be returned with the payload that contains the specific resource data, adjusted according to the passed parameters (Table 11). On failure, the specific error status code shall be returned (Table 20). If no resource matches the filtering rules, it is considered a failure case.

Table 11—Resource discovery request

Element	Description
Request type	HTTP GET
Filtering rules	The restrictions and conditions to narrow down the search results They shall be contained in the URL
Request payload	(Empty)

Table 12—Resource discovery successful response

Element	Description
HTTP status code	200 OK
HTTP header	(Empty)
Response payload	A list of the resources that match the filtering rules It shall be in JSON format

4.3.5 Resource reconfiguration

4.3.5.1 Introduction

Resource reconfiguration means the modification of the properties or attributes of the target resource. These properties and attributes are automatically generated during the resource creation operation, based on the payload of the resource creation operation request. There are two methods to reconfigure a resource: replacement and update. For ease of differentiation, these two methods can be referred to as *resource replacement* and *resource update,* respectively. The resource replacement operation replaces the whole information of the target resource, and the resource update operation only updates the information that is contained in the request payload.

For resource replacement, resources can be overwritten by the HTTP PUT method with resource representation. That is, all old resource content will be omitted after resource replacement. For resource update, the HTTP PATCH method is used to update a resource by only changing the part described in the payload from the client, leaving the other part unchanged. PATCH requests are available for all situations where PUT requests are also available. In both cases, before the modification, the resource status query operation shall be performed to ensure the version of the resource to avoid race conditions.

Regarding the other operations, the resource discovery operation is suitable for both physical and virtual resources. For example, the edge service operators may need to either update the information of their physical servers, change the settings of the computing resources, or re-slice the virtual resources. The VF M&O can also be setup to update the information automatically based on the feedback from the system.

4.3.5.2 Conditions and information involved

The pre-condition of the procedure:

— The target resource has been created in the edge system.

The post-condition of the procedure:

— The properties and attributes of the target resource will be modified. If needed, the related resources will be reconfigured as well.

4.3.5.3 Procedures—replacement

The procedures outlined in this subclause detail the steps involved in resource replacement as depicted in Figure 6. The procedures are as follows:

a) Before the resource replacement, the VF M&O needs to make the resource status query operation to get the version number of the target resource as described in 4.3.3.3. If the specific resource is found, the information of the requested resource shall be returned. If not, the process is terminated as there is no resource to replace.

b) The VF M&O sends the resource replacement request (Table 13) that contains the version number obtained in step a) to the IM/VIM.

c) The IM/VIM checks the version number of the resource. If the version number is valid, the IM/VIM will replace the target resource. If not, an error message will be returned. If needed, the related resources will be reconfigured as well.

d) The IM/VIM returns the success or error message to the VF M&O (Table 15, Table 16, or Table 20).

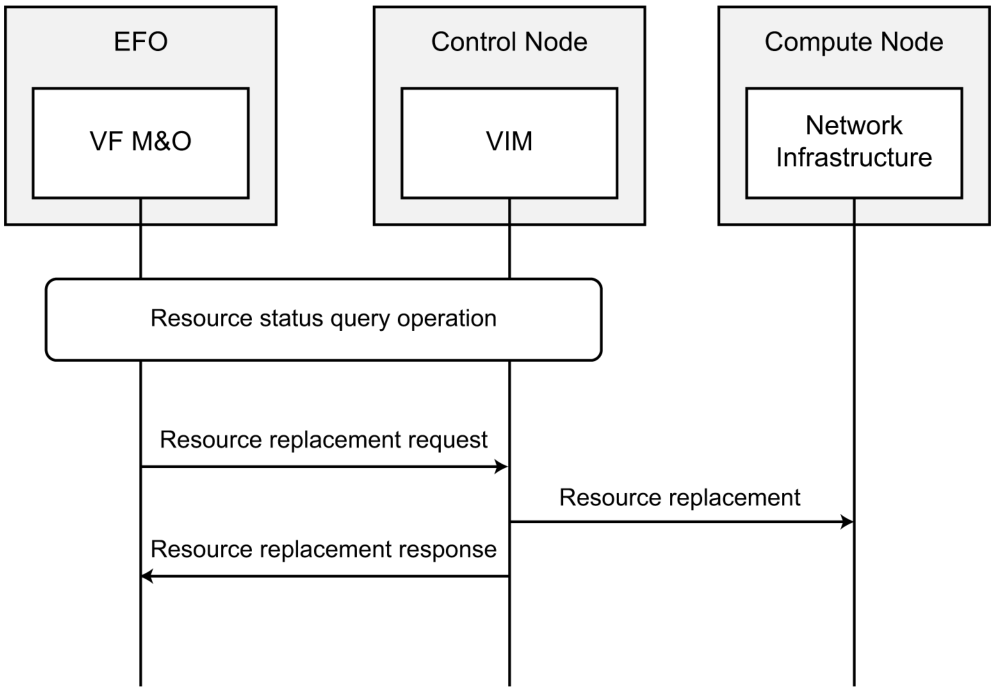
EFO
VF M&O
Control Node
VIM
Compute Node
Network Infrastructure
Resource status query operation
Resource replacement request
Resource replacement
Resource replacement response

Figure 6—The workflow of the resource replacement

4.3.5.4 Procedures—update

The procedures outlined in this subclause detail the steps involved in resource update as depicted in Figure 7. The procedures are as follows:

a) Before the resource replacement, the VF M&O needs to make the resource status query operation to get the version number of the target resource as described in 4.3.3.3. If the specific resource is found, the information of the requested resource shall be returned. If not, the process is terminated as there is no resource to replace.

b) The VF M&O sends the resource update request (Table 13) that contains the version number obtained in step a) to the IM/VIM.

c) The VIM checks the version number of the resource. If the version number is valid, the IM/VIM will update the target resource. If not, an error message will be returned. If needed, the related resources will be reconfigured as well.

d) The IM/VIM returns the success or error message to the VF M&O (Table 15, Table 16, or Table 20).

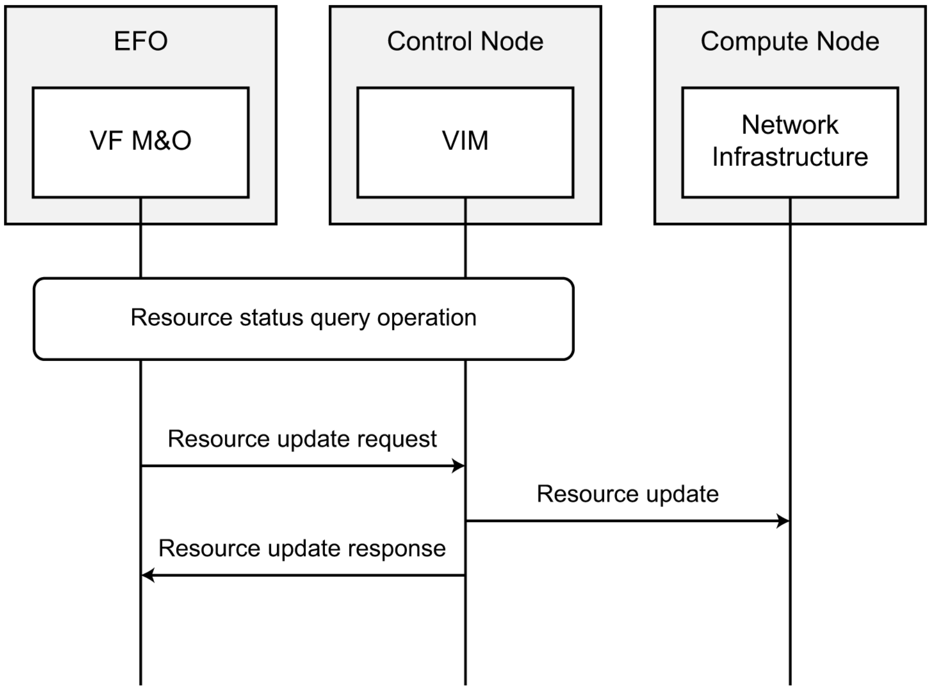

Figure 7—The workflow of the resource update

4.3.5.5 Requests and responses

On success, either "200 OK" or "204 No Content" shall be returned (Table 15 and Table 16); if a client attempts to modify the resource with an expired version number, "412 Precondition Failed" shall be returned. On other failures, the specific status error code shall be returned (Table 20).

Table 13—Resource replacement request

Element	Description
Request type	HTTP PUT
Request payload	The new data and information of the target resource All old content of the target resource will be replaced by the content of this payload
Version number	The version number of the target resource It is used to avoid the race conditions

Table 14—Resource update request

Element	Description
Request type	HTTP PATCH
Request payload	The new data and information of the target resource Only the resource attributes that are included in the payload will be updated, and all other attributes will remain unchanged
Version number	The version number of the target resource It is used to avoid race conditions

Table 15—Resource reconfiguration successful response with non-empty payload

Element	Description
HTTP status code	200 OK
HTTP header	(Empty)
Response payload	The new information of the updated resource

Table 16—Resource reconfiguration successful response with empty payload

Element	Description
HTTP status code	204 No Content
HTTP header	(Empty)
Response payload	(Empty)

4.3.6 Resource deletion

4.3.6.1 Introduction

The HTTP DELETE method is used to delete the requested resource. The request and the response are both with an empty payload, but it is also possible for the response to return the final information of the resource. After deletion, all the information about the deleted resource will be removed, and the related resources will be updated and rearranged.

As with the other operations, the resource discovery operation is suitable for both physical and virtual resources. The system can delete the target resource based on the needs and feedback. The deletion of physical resources means that these resources are removed from and no longer visible to the system, but the physical servers may be still at the same location. The deletion of a virtual resource actually removes the resource; however, it is possible to re-create the resource if the physical resource is still in the edge system.

4.3.6.2 Conditions and information involved

The pre-condition of the procedure:

— The target resource has been created in the edge system.

The post-condition of the procedure:

— The target resource will be removed from the edge system. The related resources are reconfigured.

4.3.6.3 Procedures

The procedures outlined in this subclause detail the steps involved in resource update as depicted in Figure 8. The procedures are as follows:

a) The VF M&O sends a resource deletion request to the IM/VIM (Table 17).

b) The IM/VIM deletes the target resource in the network infrastructure and cleans up the related information. If the resource is the parent of other resources, it is optional to delete those resources as well. The related resources will be reconfigured.

c) The IM/VIM returns the deletion success or error message to the VF M&O (Table 18, Table 19, or Table 20).

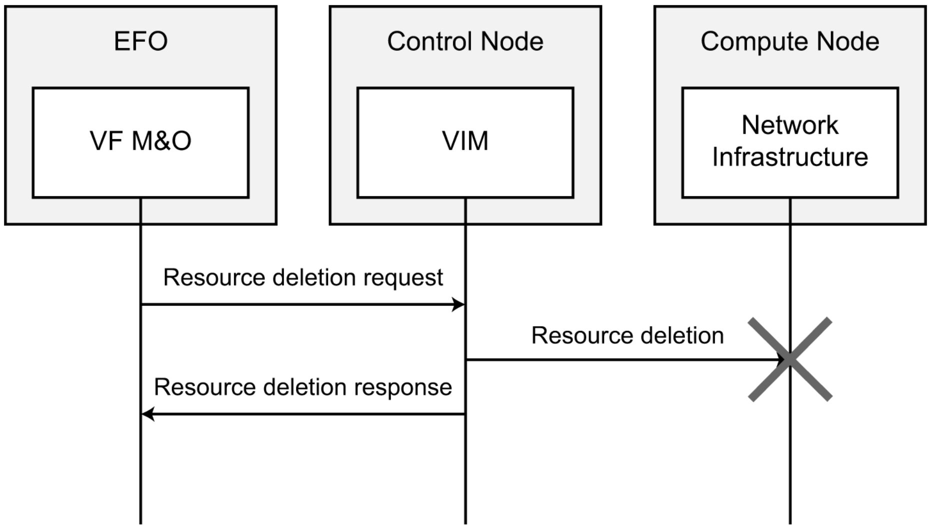

Figure 8—The workflow of the resource deletion

4.3.6.4 Requests and responses

On success, if the payload is empty, “204 No Content” shall be returned, otherwise “200 OK” shall be returned (Table 18 and Table 19). On failure, the specific error code shall be returned (Table 20).

Table 17—Resource deletion request

Element	Description
Request type	HTTP DELETE
Request payload	(Empty)

Table 18—Resource deletion successful response with non-empty payload

Element	Description
HTTP status code	200 OK
HTTP header	(Empty)
Response payload	The information of the deleted resource

Table 19—Resource deletion successful response with empty payload

Element	Description
HTTP status code	204 No Content
HTTP header	(Empty)
Response payload	(Empty)

4.3.7 Responses for request failure

Previous subclauses listed the requests and responses in success cases. This subclause discusses the failure cases and lists the corresponding status codes for such responses. If not specified, these responses are common for all operations mentioned in Clause 4.

Table 20—Status codes for request failure cases

HTTP status code	Description
400 Bad Request	This status code is used for non-specific errors.
401 Unauthorized	This status code is used when the clients have no authority to perform the operations.
403 Forbidden	This status code is used when the client is authorized but the server still refuses.
404 Not Found	This status code is used when the requested URL is invalid in the system, including the case where there is no resource meeting the filtering rules.
405 Method Not Allowed	This status code is used when the HTTP method of the request is not supported for the target resource.
409 Conflict	This status code is used when there is a conflict so that the server cannot handle the request, such as an edit conflict.
412 Precondition Failed	This status code is used if a client attempts to modify the resource with an expired version number.

5. Edge application management and orchestration

5.1 Introduction

To provide the complete lifecycle management of edge applications, the system has to have knowledge about the application runtime information, thus it is necessary to specify the procedures and the message flows between all components involved. This clause describes the managing procedures and message flows of applications at edge. Compared to Clause 4, this clause focuses on how to run and coordinate edge applications in the edge system.

5.2 Overview of edge application management

5.2.1 Introduction

Management and orchestration (M&O) means the management and coordination of the resources, including network resources and computing resources, in order to provide the lifecycle control of the edge applications. With such management, it is possible for edge service operators (or simply *operators*) to manage their edge services and applications—which is provided by edge service providers (or simply *service providers*)—and make adjustments and modifications based on the scenario, network condition, and user information. An edge service is composed of one or more edge applications, and an edge application is composed of one or more virtual functions (VFs).

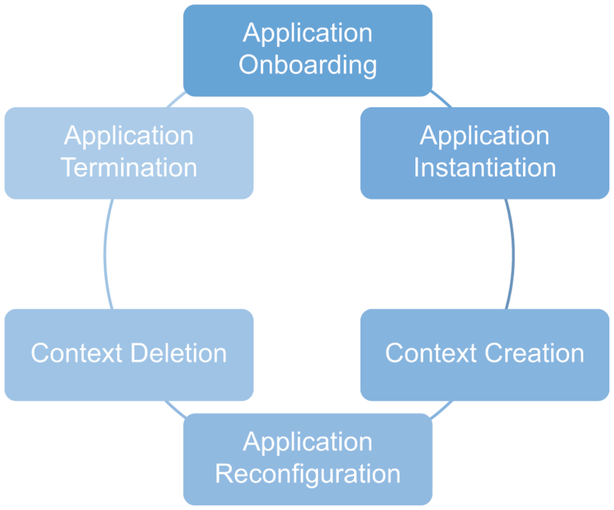

Figure 9—The diagram of lifecycle procedures of edge applications

As shown in Figure 9, a typical application has the following lifecycle management procedures:

a) Application onboarding: In the application onboarding procedure, the edge management needs to obtain the necessary software files and description files that are necessary to start running applications.

b) Application instantiation: Based on the application blueprint file that describes the management and orchestration rules for an application, the EFO is able to instantiate an application with edge app designer, making the homing decision of the edge application.

c) Context creation: After instantiation, the edge application will run on the decided edge platform. In the process, it is common for the edge service providers to collect certain user data or for the users to upload their own data. The context, or *application context,* involves all the files that contain the custom information operators need, such as user information and statistical data.

d) Context deletion: edge service providers are able to remove the certain application context (such as user data) when no longer needed.

e) Application reconfiguration: Based on the collected data and information, the edge service providers can reconfigure their edge applications. For example, it is possible to increase the accuracy of the service in the cost of latency to meet the requirements of the specific users.

f) Application termination: If an edge application is no longer needed (at least in the edge system coverage), the edge service operators can remove the edge application from the edge platform and the edge platform manager.

It is common for a standard edge application to undergo various lifecycle procedures, including onboarding, instantiation, termination, and reconfiguration. Also, for the application context, it is necessary for the edge service providers and the edge service operators to modify the data and information contained in their services that correspond to context creation and deletion procedures. An instance is a concrete occurrence of an edge application that has gone through the instantiation process. It is possible to do context creation, context deletion, and application reconfiguration whenever the edge application is active, that is, when the edge application has been instantiated and has not been terminated.

In the process, edge service providers offer their applications and services onto the edge system by uploading the specific manifest files, infrastructure owners provide their edge devices, and edge service operators authorize the edge service providers and the third-party users to leverage the functionality of their edge system. All these roles can be performed by the same group or a different group of people.

The manifest file is necessary for edge service providers to onboard their applications and service. The manifest file contains an application package and a blueprint file. The application package is hierarchical, containing the necessary data and information of the application and its components, like VF images. The blueprint file includes the orchestration rules, the resource lifecycle management designs, etc. The edge app designer component in the EFO will process the files and output the artifacts that will be distributed to the other components in the EFO.

There are also various terminologies involved in the lifecycle procedures, such as VF images and inventory. All these terms are described in Table 23 and the related subclauses.

In addition, it is common for the entities and components to send requests and responses to ensure the progress goes properly. Table 21 shows the default format of a request/response. The format of those with specific needs is described in each respective subclause.

Table 21—Default format of a request/response

Element	Description
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.

Table 22—The components mentioned in 5.3

Types	Components	4.3.2	4.3.3	4.3.4	4.3.5	4.3.6	4.3.7
Users	Service provider	√					
	Operator	√	√			√	√
	End user app			√	√		
EFO	External API		O			√	√
	End user proxy			√	√		
	Edge app designer	√					
	VF M&O	√	√	√	√	√	√
	Edge inventory manager	√	√	√	√	√	√
	LCM enabler	√					
	Rule framework	√					
	EFO controllers	√					
Control nodes	Edge platform manager	√	√	√	√	√	√
	IM/VIM	√	√			√	√
Compute nodes	Edge platform		√	√	√	√	√
	Edge app		√	√	√	√	√
	VNI and data plane	√	√			√	

Table 23—The files used in the lifecycle procedures

Terms	Descriptions	Subclause
Application context	Application context, or simply *context,* involves all the files that contain the custom information operators need, such as user information and statistical data.	5.3.4.2

Table continues

Table 23—The files used in the lifecycle procedures *(continued)*

Terms	Descriptions	Subclause
Application package	An application package is one of the files involved in the application onboarding procedures. The application package is hierarchical, containing the necessary data and information of the application and its components, like VF images. An application package can be considered as an installation file that will be deployed to a compute node to launch a specific edge application.	5.3.2.2
Artifact	Artifacts are all the files that are generated by the edge app designer component and are distributed to the other components in the EFO. They are the descriptor files for the components in the EFO.	5.3.2.2
Blueprint file	A blueprint file is a file that includes the orchestration rules, the resource lifecycle management designs, etc. This file is used to configure the components in the EFO.	5.3.2.2
Inventory	Inventory is the network information, topology, and metadata that are about the edge system environment and are stored in the edge inventory manager component.	5.3.5.2
Manifest file	Manifest files are the files used in the application onboarding procedures. A manifest file contains an application package and a blueprint file.	5.3.2.2
VF image	VF images are the virtual functions providing an executing environment for the edge application. An application package is composed of one or more virtual functions called *VF images*.	5.3.2.2

5.3 Manageability and orchestration procedures

5.3.1 Introduction

The details of the service and message design flow of each procedure are presented in this subclause. The term *application* means the edge applications, and the term *context* means the application context, including user information.

The six lifecycle management and orchestration procedures are described in the following subclause in detail. Each subclause contains four necessary parts of a procedure: (1) an introduction, (2) conditions and involved information, (3) a detailed workflow, and (4) tables of the format and necessary data of the requests and responses.

5.3.2 Application onboarding

5.3.2.1 Introduction

The purpose of the application onboarding procedure is to obtain the needed application software package and the corresponding application blueprint that describes how the application shall be deployed. The message flow of application onboarding is designed for uploading and onboarding the necessary packages of applications to the edge system.

Operators and service providers mentioned in this subclause can be assumed to be the same or different people/groups/organizations. That is, if they are assumed to be the same people, they do not need to perform the query request (step b) in 5.3.2.3) before the distribution request (step c) in 5.3.2.3). It is also possible for the edge service operators to get information of the uploaded manifest files, for example, directly from the edge service providers.

5.3.2.2 Conditions and information involved

The pre-condition of the procedure:

— The edge service providers shall prepare the manifest files.

— Once they upload their manifest files, the edge service operators are able to trigger the distribution among the EFO components.

The post-condition of the procedure:

— The artifacts are generated by the edge app designer and are distributed to the components in the EFO based on their functionality. The example can be found in 5.3.2.5.

— The application packages are saved to the VNI via the IM/VIM.

During the application onboarding procedures, the following files are involved:

— Manifest file: a file that includes an application package and a blueprint file.

— Application package: a hierarchical file containing the necessary data and information of the application and its components, VF images. An application package is an installation file that will be deployed to a compute node to launch a specific edge application.

— VF image: a virtual function. An application package is composed of one or more virtual functions called *VF images*.

— Blueprint file: a file that includes the orchestration rules, the resource lifecycle management designs, etc. This file is used to configure the components in the EFO.

— Artifacts: all the files generated by the edge app designer component after the manifest file is passed to the edge app designer. Artifacts are the descriptor files for the components in the EFO and will be distributed to the other components in the EFO.

5.3.2.3 Procedures

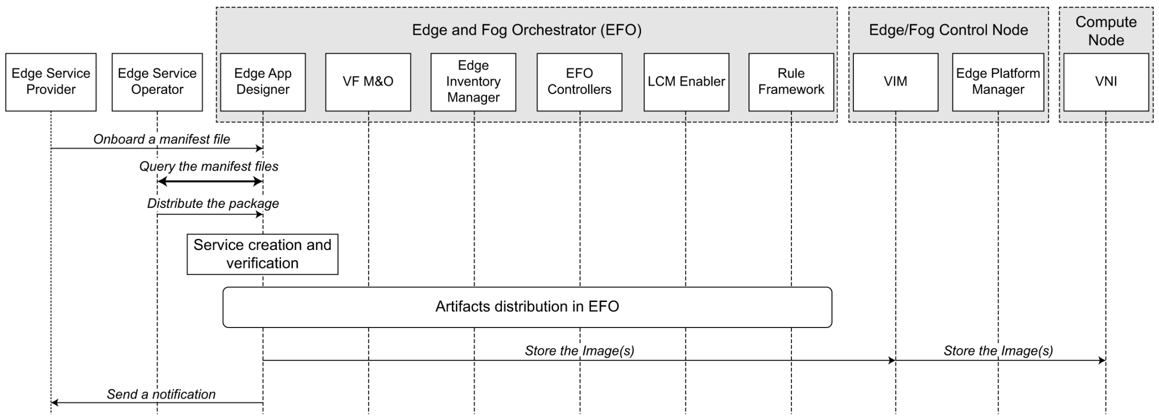

Figure 10—The workflow of application onboarding procedures

The following provides a detailed description of the flow illustrated in Figure 10:

a) The edge service provider onboards a manifest file that includes an application package and a blueprint file via the edge app designer component of the EFO (Table 24). The edge app designer checks if the edge service provider is authorized to perform the uploading operation. A response shall be made if successful (Table 25).

b) (Optional) the edge service operator sends a query request (Table 26 and Table 27) to the edge app designer component to get the list of the uploaded application manifest files. Then the edge service operator selects and filters these files based on their needs and triggers the distribution of the packages and blueprint files.

c) The edge service operator sends a distribution request (Table 28) to trigger the distribution of the packages and blueprint files contained in the selected manifest file.

d) The edge app designer component verifies the request, checking if the edge service operator is authorized to perform the distribution operation and if the related procedures and the information are ready.

e) If the request passes the authorization by the edge app designer component, the component starts to create the edge application. The edge app designer generates the artifacts as the output.

f) All the artifacts are distributed to their responsible components in the EFO from the edge app designer component. The example can be found in 5.3.2.5.

g) The application package is saved to the VNI that is responsible for storage via the corresponding IM/VIM.

h) The edge app designer sends a notification to the edge service operator for the success of the distribution (Table 29).

5.3.2.4 Requests and responses

Table 24 through Table 29 describe the elements contained in the request.

Table 24—Application manifest file uploading request

Element	Description
Service provider ID	This is the edge service provider identifier.
Manifest file	A manifest file includes an application package and a blueprint file. It is the payload of the request.

Table 25—Application manifest file uploading response

Element	Description
Operation status	A 201 Created code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Manifest file ID	This is the generated identifier of the uploaded manifest file. This field will be blank if the upload fails.

Table 26—Uploaded manifest files query request

Element	Description
Operator ID	This is the edge service operator identifier. The ID shall be authorized by the EFO.
Query filter	This is the filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations.

Table 27—Uploaded manifest files query response

Element	Description
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Query result	This is a list of the manifest files that satisfy the query conditions.

Table 28—Artifact distribution request

Element	Description
Operator ID	This is the edge service operator identifier. The ID shall be authorized by the EFO.
Manifest file ID	This is the identifier of the manifest file that the edge service operator supposes to distribute.

Table 29—Distribution success notification

Element	Description
Operator ID	This is the edge service operator identifier.
Application ID	When packages are distributed successfully, the application is given an identifier for the edge service operators to instantiate it in the next subclause.

5.3.2.5 Artifact distribution

This subclause describes an example of the process of artifact distribution, which is depicted in Figure 11. The artifact distribution is mentioned in step f) of 5.3.2.3. Its purpose is to classify and send the relevant files to the corresponding component in the EFO.

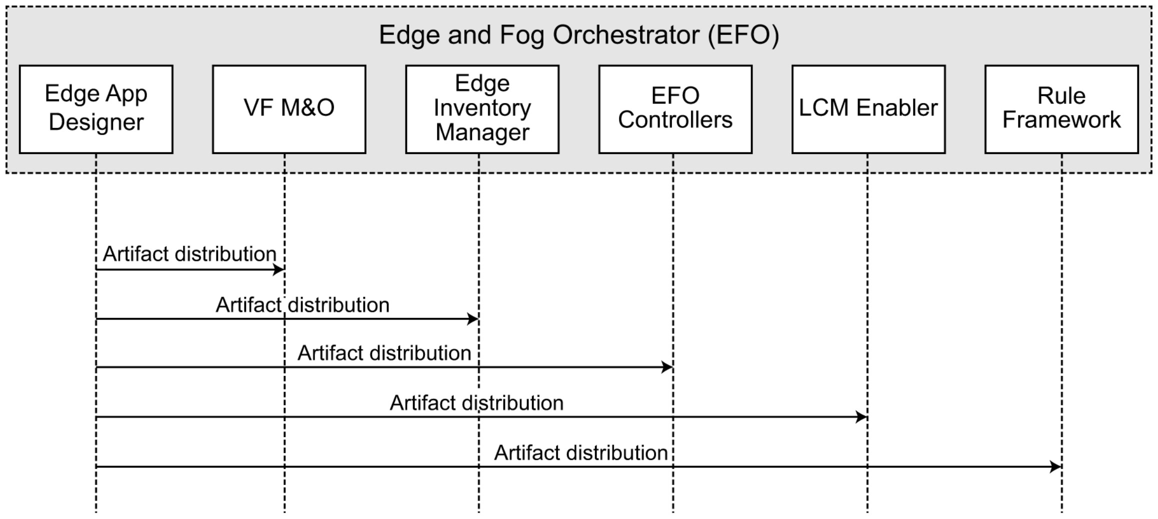

Figure 11—A schematic diagram of the workflow of artifact distribution

The resource lifecycle management designs will be distributed to the LCM enabler component, and the orchestration rules will be distributed to the rule framework component. The VF M&O is responsible for the management and orchestration of edge applications in their runtime. EFO controllers assist EFO to control and communicate with the entities in control nodes and compute nodes. Edge inventory manager stores the information and the topology of edge applications and the related environment. The related files contained in the manifest file will be distributed to the corresponding components according to their functionality.

5.3.3 Application instantiation

5.3.3.1 Introduction

The purpose of application instantiation is to launch the service (that is, the edge application) on a certain edge platform to serve the customers. The message flow of application instantiation and start-up is used to instantiate an application instance in the edge system and have it properly configured.

5.3.3.2 Conditions and information involved

The pre-condition of the procedure:

— The artifacts have been distributed to each component in the EFO.

— The application package has been saved to the VNI.

The post-condition of the procedure:

— The edge application has been instantiated following the steps described below.

— The inventory is created and updated based on the information of the created edge application instance.

— The edge platform has been configured to allocate its computing and networking resources properly.

During the application instantiation procedures, the following files are involved:

— Workload: the workload covers all the pre-work needed for the application to launch properly.

— The inventory: the network information, topology, and metadata that are about the edge system environment and are stored in the edge inventory manager component.

5.3.3.3 Procedures

The workflow of the application instantiation procedures, as depicted in Figure 12, unfolds as follows:

a) The edge service operator sends an application instantiation request (Table 30) to the VF M&O component through external API. A response shall be made (Table 21).

b) The VF M&O makes the homing decision of the instance to be instantiated. The homing decision decides on which edge platform the edge application will be instantiated.

c) The VF M&O creates the empty inventory in the edge inventory manager (Table 31). The empty inventory will be fulfilled with real values after the edge application is created.

d) The VF M&O sends a service deployment request to the target edge platform manager (Table 33). A response shall be made if successful (Table 21).

e) The VF M&O requests the IM/VIM to allocate resources, including computing resource and networking resource (Table 34). A response shall be made if successful (Table 21).

f) The VF M&O requests the edge platform manager entity to instantiate the workload Table 35). A response shall be made if successful (Table 21).

g) The IM/VIM deals with the instantiation work on the infrastructure (Table 36). In the process, the IM/VIM loads the needed application package from the VNI and deploys it onto the edge platform (Table 37 and Table 38). A response shall be made if successful (Table 21).

h) The edge platform manager informs the edge platform assigned by the EFO to create the edge application (Table 39). A response shall be made if successful (Table 21).

i) The VF M&O requests the edge platform manager to activate the instance (Table 40). The edge platform manager informs the corresponding edge platform to deal with the configuration work on the instance (Table 41). A response shall be made if successful (Table 21).

j) The edge platform returns the information to the edge platform manager to update the inventory in the edge inventory manager (Table 32).

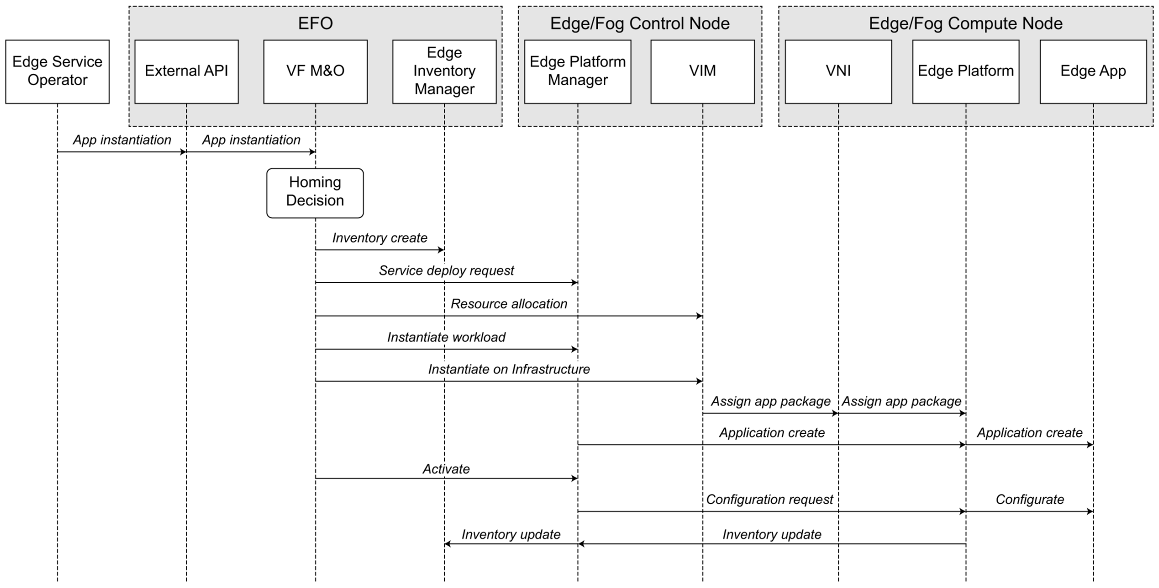

Figure 12—The workflow of application instantiation procedures

5.3.3.4 Requests and responses

Table 30 through Table 41 describe the elements contain in the requests and the responses.

Table 30—Application instantiation request

Element	Description
Operator ID	This is the edge service operator identifier. The ID shall be authorized by the EFO.
Application ID	The identifier of the application that the edge service operator supposes to instantiate.

Table 31—Inventory update request from an edge platform to an edge platform manager

Element	Description
Edge platform ID	This is the edge platform identifier.
Application ID	This is the identifier of the application that has been instantiated successfully.
New inventory	This is the empty inventory that will be updated after instantiation finished.

Table 32—Inventory update request from an edge platform manager to the edge inventory manager in an EFO

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
Updated inventory	This is the updated inventory. It shall fulfill the empty inventory that has been created with parameters and values of the real environment.

Table 33—Service deployment request

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.

Table 34—Resource allocation request

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
VIM ID	This is the IM/VIM identifier.
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.

Table 35—Application manifest file uploading request

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.

Table 36—Application manifest file uploading request

Element	Description
VIM ID	This is the IM/VIM identifier.
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.

Table 37—Application manifest file uploading request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.
Edge platform ID	This is the edge platform identifier that the IM/VIM tells the VNI to assign the application package to.

Table 38—Application manifest file uploading request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.
Application package	This is the application package for the target application to be instantiated.

Table 39—Application manifest file uploading request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.

Table 40—Application manifest file uploading request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.

Table 41—Application manifest file uploading request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to instantiate.
Configuration details	This is the configuration of the edge app. The details of the configuration vary depending on the system and the edge app itself.

5.3.4 Application context creation

5.3.4.1 Introduction

The purpose of context creation is to add necessary data and information to the system. The application context may be regarded as some information that helps edge service operators make decisions such as streaming resolution for each user in the case of a video-streaming application. If there is already some context in the system, the request is sent for the new context to join with the already existing context; if not, a new space to store the context needs to be instantiated. For example, an end user updates its selected streaming resolution to the system.

In practice, it is possible for both end users and edge service operators to do application context creation, and the procedures are different in these two cases. The end users report and update their personal data regularly, and these data may be used to make some better decisions by the system and the edge service operators. The edge service operators are also able to recreate or add the information according to their own needs, like they may update the user data to reconfigure the performance of the edge application every season. The procedures for end users are described in 5.3.4.3, and procedures for edge service operators are described in 5.3.4.4.

5.3.4.2 Conditions and information involved

The pre-condition of the procedure:

- The edge app has been instantiated.
- The inventory has been created and updated in the edge inventory manager.

The post-condition of the procedure:

- The application context has been added and stored into the edge application.

During the application context creation procedures, the following files are involved:

- Application context: application context involves all the files that contain the custom information operators need, such as user information and statistical data. The application context also includes the data that can assist the edge service operators to make decisions.
- The inventory: the network information, topology, and metadata that are about the edge system environment and are stored in the edge inventory manager component.

5.3.4.3 Procedures (for end users)

The procedures of application context creation, as depicted in Figure 13, unfold as follows:

a) The end user app sends the application list query request (Table 42) to the end user proxy (EUP).
b) Upon receiving the request, the EUP verifies who sends this request and for what purpose and authorizes the request if it is valid; that is, from a valid end user and with valid request content.

c) If the request is valid, the EUP retrieves the list of edge applications available to the end user app from the edge inventory manager in the EFO (Table 43 and Table 44).

d) The EUP sends the query results to the end user app (Table 45). The query results shall include a list of all applications that satisfy the end user's requirements.

e) The end user app sends the application context creation request, which contains the application context to be created (Table 46), to the EUP.

f) The EUP verifies the request from the user app to check who sends this request and for what purpose, and then forwards the request to the VF M&O in the EFO.

g) The VF M&O grants the request, and if necessary (for example, there is no application available in nearby area), does the application instantiation with the other components in the EFO and the edge platform manager. The details of instantiation are the same as described in 5.3.3.3.

h) The EFO forwards the context creation request (Table 46) to the instance, and the instance registers the user information inside.

i) The context creation complete response is replied to the end user app (Table 47).

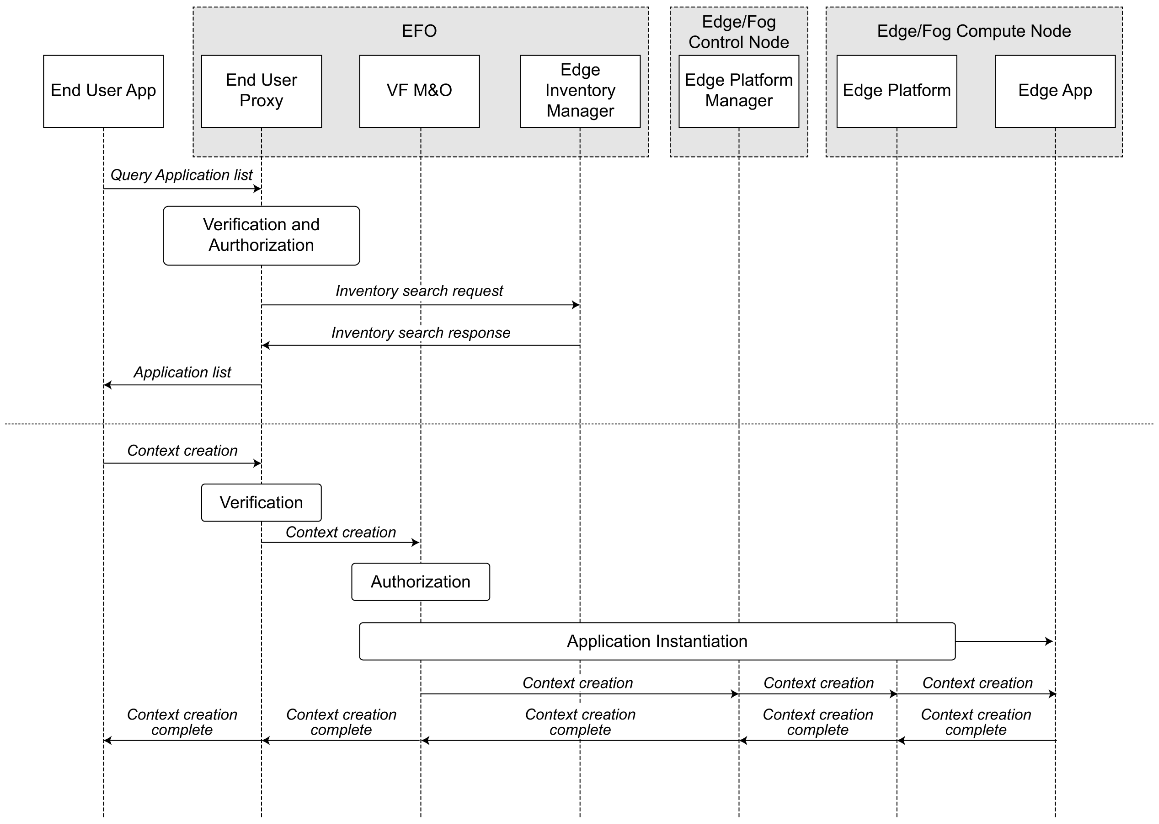

Figure 13—The workflow of application context creation procedures

5.3.4.4 Procedures (for edge service operators)

The procedures of application context creation, as shown in Figure 14, are the following:

a) The edge service operator sends the application list query request (Table 42) to the edge inventory manager via the external API.

b) The request retrieves the list of edge applications available to the end user app from the edge inventory manager in the EFO (Table 43 and Table 44).

c) The query results are sent to the edge service operator (Table 45). The query results shall include a list of all applications that satisfy the end user's requirements.

d) The edge service operator sends the application context creation request, which contains the application context to be created (Table 46), to the VF M&O via the external API.

e) The VF M&O grants the request and, if necessary (for example, there is no application available in nearby area), does the application instantiation with the other components in the EFO and the edge platform manager. The details of instantiation are the same as described in 5.3.3.3.

f) The EFO forwards the context creation request (Table 46) to the instance, and the instance registers the user information inside.

g) The context creation complete response is replied to the end user app (Table 47).

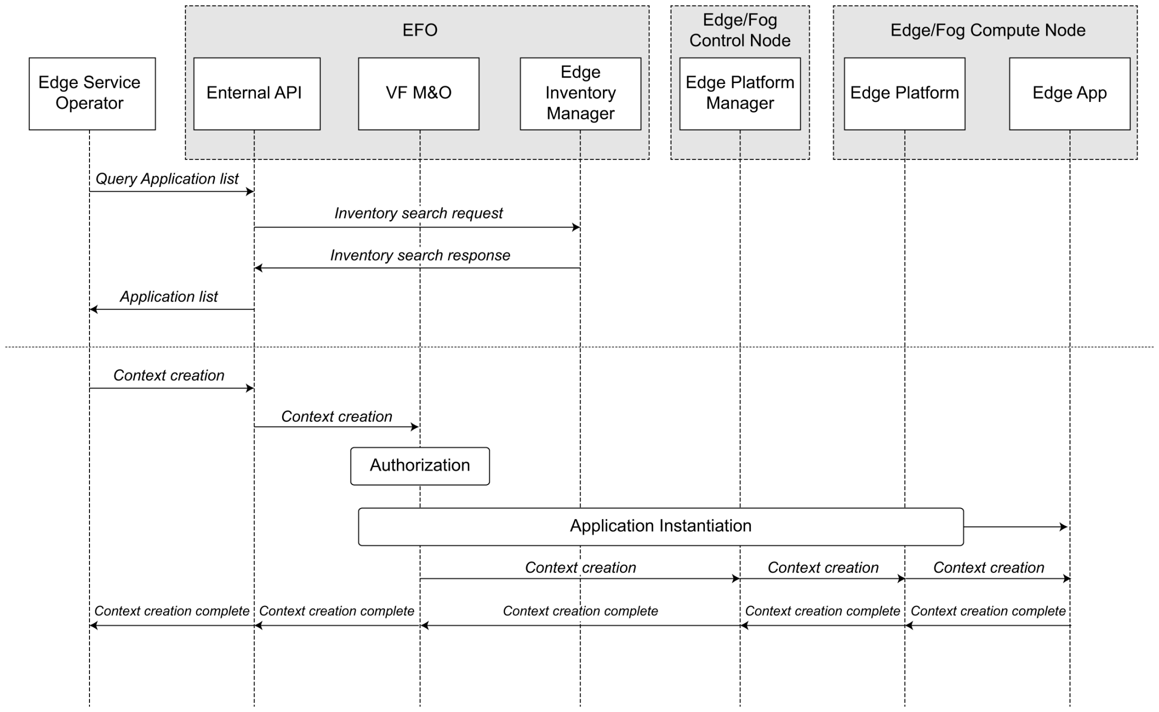

Figure 14—The workflow of application context creation procedures

5.3.4.5 Requests and responses

Table 42 through Table 47 describe the elements contained in the requests and the responses.

Table 42—Application list query request

Element	Description
End user app ID	This is the end user app identifier. The ID shall be authorized by the EFO.
Query filter	This is the filter conditions for the query. End user apps can only query the applications that satisfy these conditions and limitations.

Table 43—Inventory search request

Element	Description
Query ID	This is the query identifier.
Query filter	This is the filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations.

Table 44—Inventory search response

Element	Description
Query identifier	This is the query identifier.
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Query result	This is a list of the manifest files that satisfy the query conditions.

Table 45—Application list query response

Element	Description
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Query results	This is a list of the applications that satisfy the query conditions.

Table 46—Application context creation request

Element	Description
End user app ID	This is the end user app identifier. The ID shall be authorized by the EFO.
Application ID	This is the identifier of the application that the end user app supposes to do context creation.

Table 47—Application context creation response

Element	Description
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Application ID	This is the identifier of the application that the end user app supposes to do context creation.

5.3.5 Application context deletion

5.3.5.1 Introduction

The purpose of application context deletion is to delete the application context that is added or collected in the process of application context creation. The application context may be regarded as some information that helps edge service operators to make the decision, such as streaming resolution for each user in the case of a video-streaming application. As context creation, it is possible for both end users and edge service operators to perform an application context deletion operation. For example, the edge service operators may clean up the user data stored in the edge system every season. The procedures for end users are described in 5.3.5.3, and the procedures for edge service operators are described in 5.3.5.4.

5.3.5.2 Conditions and information involved

The pre-condition of the procedure:

— The edge app has been instantiated.

— The inventory has been created and updated in the edge inventory manager.

— The application context has been created in the target edge application.

The post-condition of the procedure:

— The application context has been removed from the system.

During the application context deletion procedures, the following files are involved:

— Application context: application context involves all the files that contain the custom information operators need, such as user information and statistical data.

— The inventory: the network information, topology, and metadata that are about the edge system environment and are stored in the edge inventory manager component.

5.3.5.3 Procedures (for end users)

The procedures of application context deletion performed by end users, as depicted in Figure 15, are as follows:

a) The end user app sends the application context deletion request (Table 48) to the end user proxy (EUP) that contains the application context to be deleted.

b) The EUP verifies the request from the end user app to check who sends this request and for what purpose, and then forwards the request to the VF M&O.

c) The VF M&O grants the request and forwards the context deletion request to the instance, and the instance erases the related user information.

d) The context deletion complete response is returned to the end user app (Table 49).

e) (Optional) If necessary, the EFO and the corresponding entities do the application termination as mentioned in 5.3.6.3.

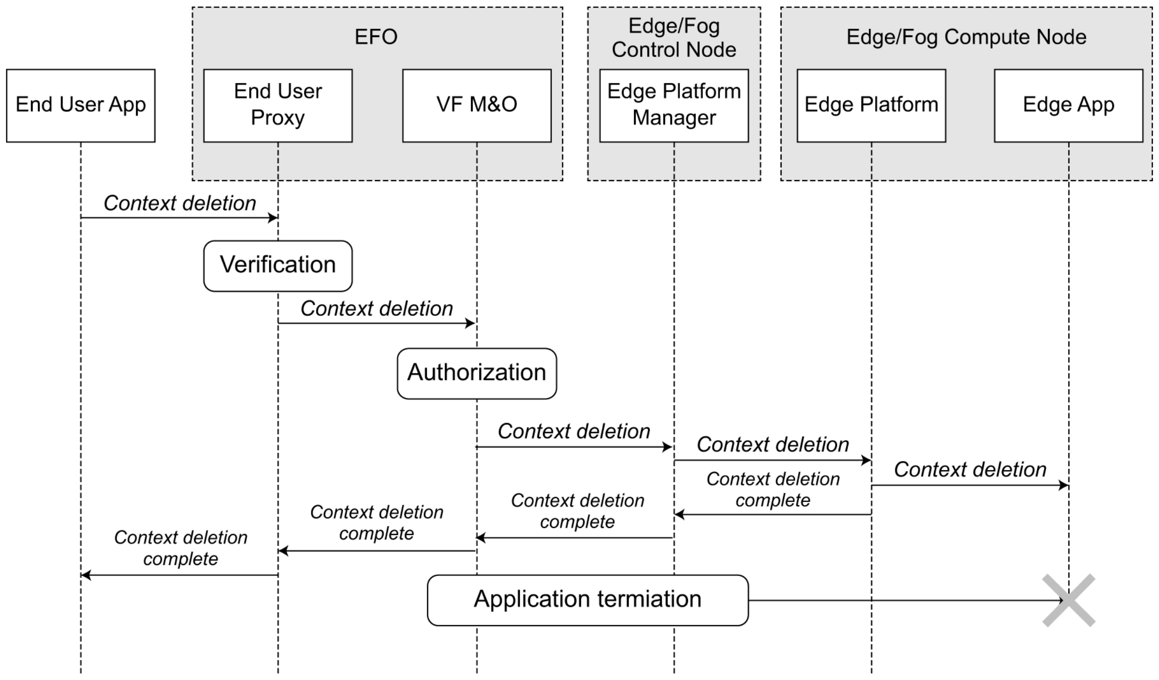

Figure 15—The workflow of application context deletion procedures

5.3.5.4 Procedures (for edge service operators)

The procedures of application context deletion performed by edge service operators, as depicted in Figure 16, are as follows:

a) The end user app sends the application context deletion request (Table 48), which contains the application context to be deleted, to the VF M&O via the external API.

b) The VF M&O grants the request and forwards the context deletion request to the instance, and the instance erases the related user information.

c) The context deletion complete response is returned to the edge service operator (Table 49).

d) (Optional) If necessary, the EFO and the corresponding entities do the application termination as mentioned in 5.3.6.3.

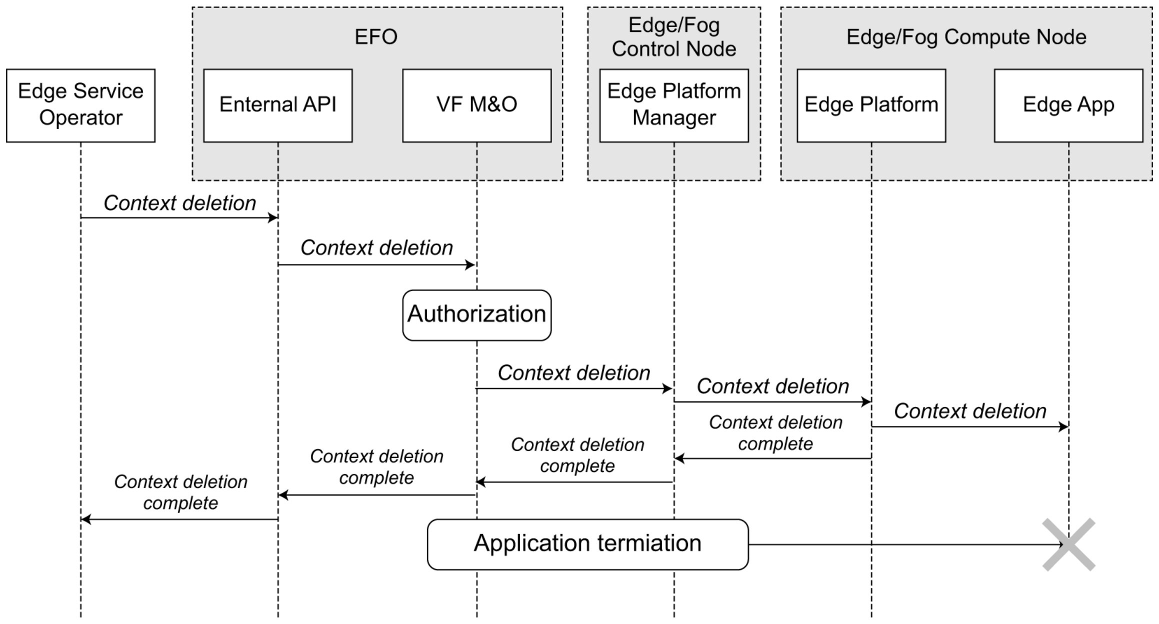

Figure 16—The workflow of application context deletion procedures

5.3.5.5 Requests and responses

Table 48 and Table 49 describe the elements contained in the requests and the responses.

Table 48—Application context deletion request

Element	Description
End user app ID	This is the end user app identifier. The ID shall be authorized by the EFO.
Application ID	This is the identifier of the application that the end user app supposes to do context deletion.

Table 49—Application context deletion response

Element	Description
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Application ID	This is the identifier of the application that the end user app supposes to do context deletion.

5.3.6 Application termination

5.3.6.1 Introduction

The purpose of application termination is to terminate an application instance in the edge system. If the application is no longer used, that is, if there is no user in the region accessing the application, the edge service operator can terminate and remove the application from the control and compute nodes.

5.3.6.2 Conditions and information involved

The pre-condition of the procedure:

— The edge app has been instantiated.

— The inventory has been created and updated in the edge inventory manager.

The post-condition of the procedure:

— The edge app has been terminated.

— The related inventory is removed from the edge inventory manager.

— The artifacts, which are distributed to each component in the process of onboarding, are still saved in each component in the EFO.

During the application termination procedures, the following files are involved:

— Workload: all the work that needs to be done before the edge application instance actually starts up.

5.3.6.3 Procedures

The detailed description of the application termination workflow, as depicted in Figure 17, is as follows:

a) The edge service operator sends an application termination request (Table 50) to the VF M&O component in the EFO through the external API.

b) The VF M&O retrieves the location of the instance from edge inventory manager (Table 51 and Table 52).

c) The VF M&O sends service terminate request (Table 50) to the target edge platform manager. A response shall be made if successful (Table 21).

d) The VF M&O requests the edge platform manager to deactivate the instance (Table 53).

e) The edge platform manager informs the corresponding edge platform to do the configuration work to the edge application (Table 54). The configuration may include the necessary settings that shall be made before the edge application is actually shut down, such as terminate the internal VFs in order. A response shall be made if successful (Table 21).

f) The VF M&O requests the edge platform manager to terminate the workload (Table 53).

g) The edge platform manager informs the IM/VIM to deal with the termination work on the infrastructure (Table 55). The IM/VIM terminates the related VNI according to the request. A response shall be made if successful (Table 21). Note that the default response is also generated and sent to the VF M&O.

h) The edge platform manager informs the edge platform to actually terminate and delete the edge application (Table 56). A response shall be made if successful (Table 21).

i) The edge platform manager returns the information to update the inventory in the edge inventory manager component (Table 57).

j) The VF M&O requests the IM/VIM to release corresponding resources, including computing and networking resources (Table 58). A response shall be made if successful (Table 21).

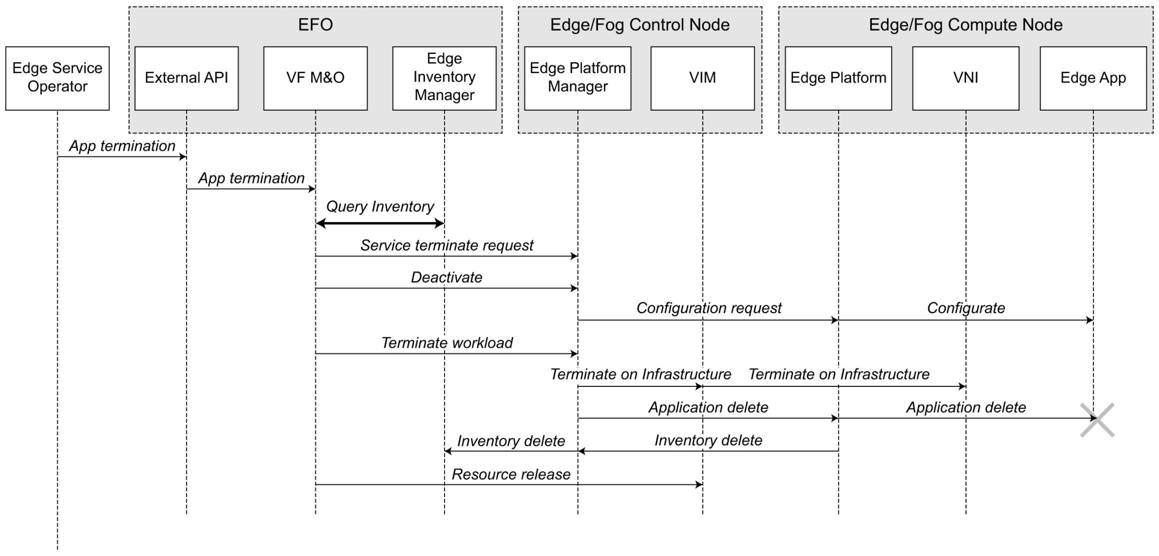

Figure 17—The workflow of application termination procedures

5.3.6.4 Requests and responses

Table 50 through Table 58 describe the elements contained in the requests and the responses.

Table 50—Application termination request

Element	Description
Operator ID	This is the edge service operator identifier. The ID shall be authorized by the EFO.
Application ID	This is the identifier of the application that the edge service operator supposes to terminate.

Table 51—Instance location retrieval request

Element	Description
Query ID	This is the query identifier.
Query filter	This is the filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations. In this case, there shall be only one edge application that satisfies the conditions.

Table 52—Instance location retrieval response

Element	Description
Query identifier	This is the query identifier.
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Query result	This is the location information of the target edge application.

Table 53—Deactivation and workload termination request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to terminate. The edge platform manager will deactivate and terminate the related workload according to the request.

Table 54—Edge app configuration request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to terminate.
Configuration details	This is the configuration of the edge app. The details of the configuration vary depending on the system and the edge app itself.

Table 55—Infrastructure termination request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to terminate. The IM/VIM will terminate the use of infrastructure according to the request.

Table 56—Application termination and deletion request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to terminate. The edge platform will terminate and delete the edge application according to the request.

Table 57—Inventory update request

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
Updated inventory	This is the updated inventory. After application termination, the edge application will be no longer available and thus the connections to its end users will be cut off, and some of the edge platform may be idle now. All this information needs to be updated.

Table 58—Resource release request

Element	Description
Application ID	This is the identifier of the edge application that has been terminated.

5.3.7 Application configuration/re-configuration

5.3.7.1 Introduction

The purpose of application reconfiguration is to update the settings of the edge applications in the edge system. The message flow of application reconfiguration is used to update the settings and configuration of the target edge application.

5.3.7.2 Conditions and information involved

The pre-condition of the procedure:

— The edge application has been instantiated.

— The inventory has been created and updated in the edge inventory manager.

The post-condition of the procedure:

— The configurations of IM/VIM, edge platform, and edge application have been updated.

— The inventory in the edge inventory manager has been updated based on the reconfiguration result.

5.3.7.3 Procedures

The detailed description of the application reconfiguration workflow, as depicted in Figure 18, is as follows:

a) The edge service operator sends an application reconfiguration request (Table 59) to the VF M&O component in the edge and fog orchestrator (EFO) through the external API.

b) The VF M&O retrieves the location of the instance from the edge inventory manager (Table 60 and Table 61).

c) The VF M&O sends an application reconfiguration request to the target edge platform manager (Table 62). A response shall be made if successful (Table 21).

d) The edge platform manager informs the IM/VIM to update the configuration on the infrastructure (Table 62). A response shall be made if successful (Table 21).

e) The edge platform manager requests the corresponding edge platform to configure the target edge application (Table 63). A response shall be made if successful (Table 21).

f) (Optional) If there is any change in the system inventory, the edge platform manager returns the information to update the inventory in the edge inventory manager component (Table 64).

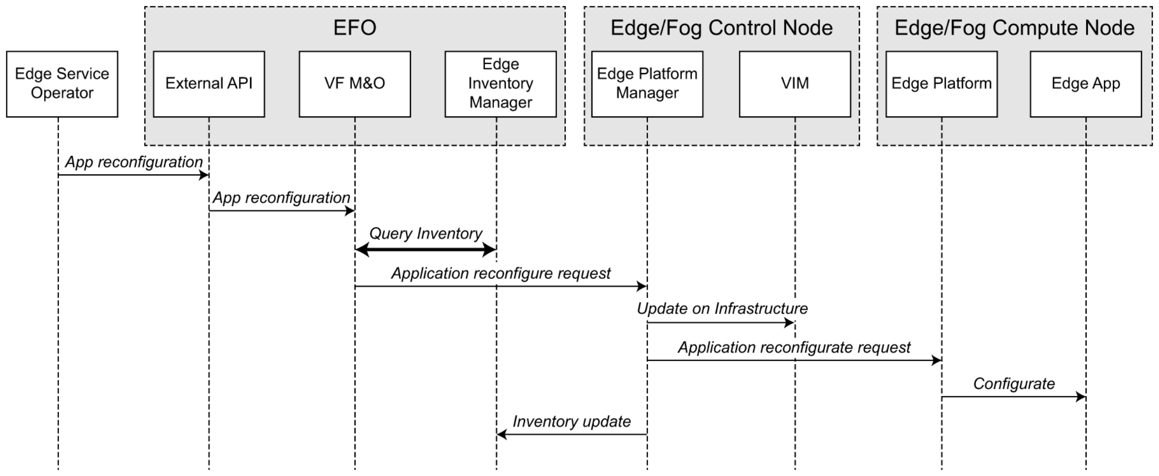

Figure 18—The workflow of application reconfiguration procedures

5.3.7.4 Requests and responses

Table 59 through Table 64 describe the elements contained in the requests and the responses.

Table 59—Application termination request

Element	Description
Operator ID	This is the edge service operator identifier. The ID shall be authorized by the EFO.
Application ID	This is the identifier of the application that the edge service operator supposes to reconfigure.
New configuration	Thes are the new settings or the new configuration files of the application.

Table 60—Instance location retrieval request

Element	Description
Query ID	This is the query identifier.
Query filter	This is the filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations. In this case, there shall be only one edge application that satisfies the conditions.

Table 61—Instance location retrieval response

Element	Description
Query identifier	This is the query identifier.
Operation status	A 200 OK code shall be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code shall be returned if failed.
Query result	This is the location information of the target edge application.

Table 62—Application termination request

Element	Description
Application ID	This is the identifier of the application that the edge service operator supposes to reconfigure.
New configuration	This is the new settings or the new configuration files of the application.

Table 63—Application termination request

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
Application ID	This is the identifier of the application that the edge service operator supposes to reconfigure.
New configuration	This is the new settings or the new configuration files of the application.

Table 64—Inventory update request

Element	Description
Edge platform manager ID	This is the edge platform manager identifier.
New inventory	This is the inventory that is updated after the application reconfiguration.

6. Management domains

6.1 Introduction

As shown in Figure 19, an edge computing system may be distributed across multi-tiers of edge computing nodes and EFOs managed by different service providers. Thus, there is a need to establish an administrative superstructure to enforce interconnectivity and interoperability among these edge entities managed by the same edge service provider and orchestrate the workflows of applications and virtual functions running on these nodes. This clause defines two types of administrative domains related to the multiple edge system scenario, known as the connectivity/interoperability domains (CIDs) and the service/application domains (SADs), to achieve these objectives.

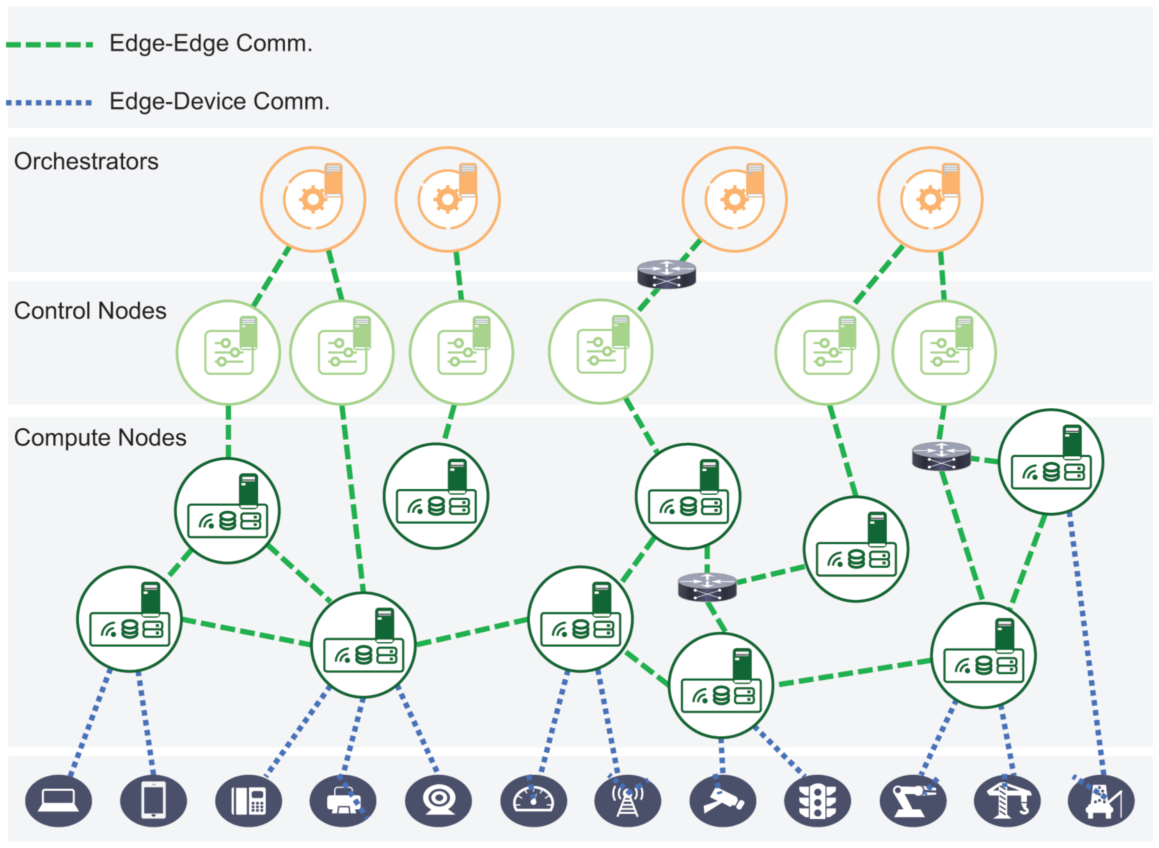

Figure 19—Multi-tier edge computing infrastructure

6.2 Domain components

6.2.1 Introduction

6.2.2 Physical edge nodes

A *physical edge node* refers to a physical computing entity that can run edge applications and virtual functions in the form of virtual machine (VM) appliances or containers and support their management and orchestration through an edge control node as a management/orchestration agent.

Physical edge nodes may also provide management of other end devices, such as sensors and actuators, connected to those nodes. In this case, these physical edge nodes act as proxy agents to expose abstractions of these devices to the higher-level edge management system, including the edge control node and the edge orchestrator (EFO).

6.2.3 Logical edge nodes

A logical edge node represents a group of edge nodes in the same cluster that are managed and orchestrated as a single logical entity at the network edge.

A logical edge node shall be managed by an edge control node as a logical node management agent. The edge control node, as the master of the cluster, represents all the edge nodes in the cluster to the higher-level management, the EFO. It works by presenting a managed object model of the logical edge node. This object model includes an aggregated abstraction of hardware components, software modules, and other resources of individual nodes, along with the metadata specifying the functional and structural composition of the cluster.

The control node also helps the EFO perform the orchestration of microservices and applications running in a logical edge node. In this regard, the control node shall expose the capability, resources, workload status, and telemetry of individual nodes and the entire cluster to the EFO.

6.2.4 Virtual edge nodes

A virtual edge node is a software runtime environment that presents itself as an edge node and can thus be managed as a distinct edge compute node. It may be a virtual machine (VM), a container, or any other possible environment. Multiple virtual edge nodes can be deployed on a single physical or logical edge node. A virtualized industrial controller running on a physical edge node is a typical example of a virtual edge node.

An edge control node shall also manage a virtual edge node. The management behaves similarly to the physical case except that it allocates the virtual resources rather than the physical ones of its host. The host of a virtual edge node—which may be physical, logical, or even another virtual edge node—shall be aware of the presence and the characteristics of the virtual edge node, including the physical resources associated with that virtual node.

Orchestration of applications and virtual functions running in a virtual edge node works in the same way as in a physical edge node with the exception that the underlying resources are virtual. The control node and its API shall be the sole point of contact between the virtual edge node and the EFO.

6.3 Connectivity/interoperability domains (CIDs)

6.3.1 Introduction

A connectivity/interoperability domain (CID) is a collection of end nodes (e.g., IoT devices), edge nodes, and EFOs that belong to the same edge service operator and can interconnect[10] and interoperate[11] with one another. That is, two EFOs in two different CIDs may still be able to interconnect or interoperate with each other but belong to two different edge service operators. Entities that belong to the same CID are managed by the same EFO and governed by the same set of operating rules produced by rule framework. They also implement a common set of communication, computing, and storage functions.

[10]Interconnectivity is the capability of networked entities to exchange information with one another.
[11]Built upon interconnectivity, interoperability is the capability of networked entities to interpret the information exchanged with one another consistently according to a specified set of syntactic and semantic rules.

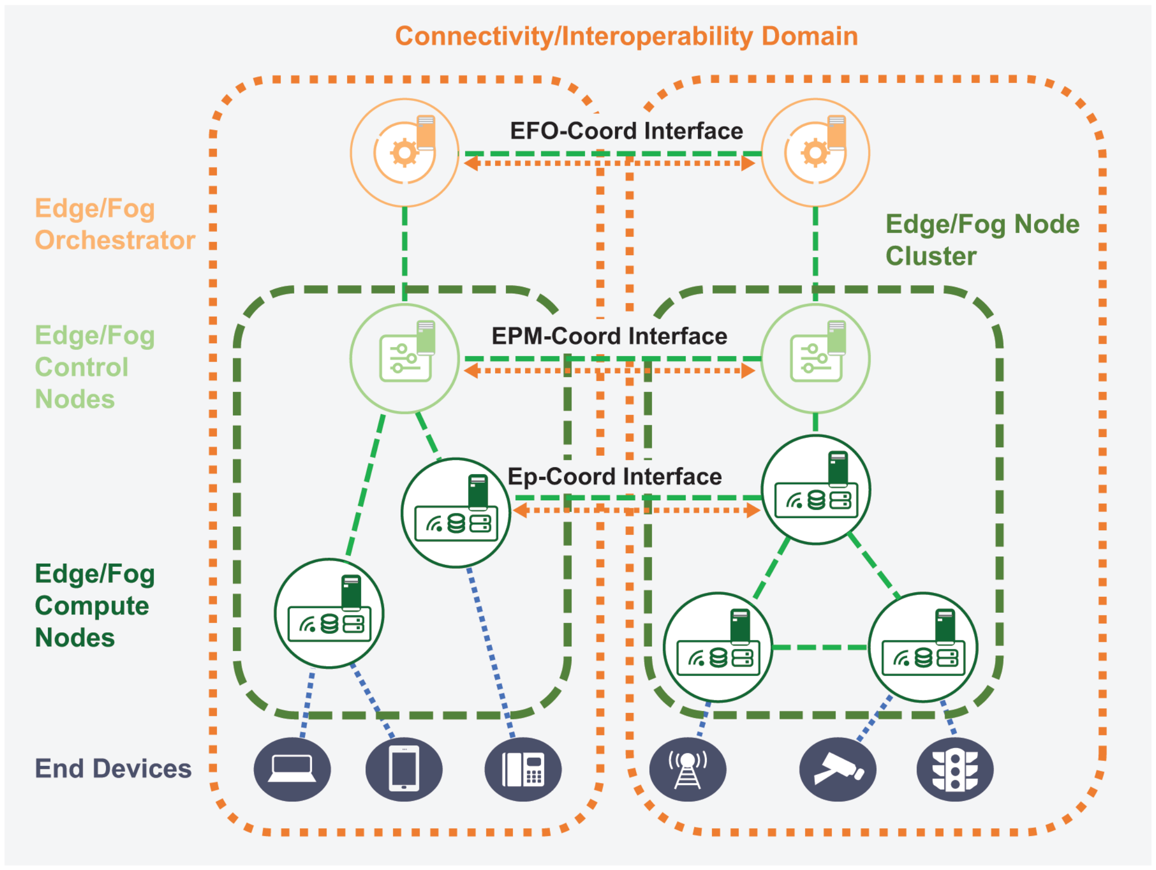

Figure 20—Structure of connectivity/interoperability domains (CIDs)

6.3.2 Organization

6.3.2.1 Introduction

A CID must consist of an edge orchestrator, one or more edge control nodes, and one or more edge compute nodes. In addition to them, it may also involve some end devices, such as smartphones, laptops, and cameras. The control and compute nodes can be organized into several clusters based on their geographical characteristics or runtime environments. Also, two CIDs can communicate with each other, either sharing data or information, via inter-CID communications. The following subclauses explain the concept of these terms and components.

6.3.2.2 Edge node clusters

The edge nodes in a CID may organize themselves into multiple clusters, each of which may implement the common set of communication, computing, and storage functions differently. Every cluster can be considered as one or more logical edge nodes, having an edge control node to manage them. For example, the edge nodes running docker containers in a CID can form a cluster with a control node for docker and distinguish themselves from those running open container initiative (OCI) containers in the same CID.

6.3.2.3 Border edge nodes

The edge nodes in a CID may organize themselves into multiple clusters, each of which may implement the common set of communication, computing, and storage functions differently. Every cluster can be considered as one or more logical edge nodes, having an edge control node to manage them. For example, the edge nodes running docker containers in a CID can form a cluster with a control node for docker and distinguish themselves from those running open container initiative (OCI) containers in the same CID.

6.3.2.4 Inter-CID communication

Inter-CID communication refers to the communication between two different CIDs. The communication can go through several specific interfaces, such as Ep-Coord, EPM-Coord, and EFO-Coord. For example, two edge compute nodes can exchange some necessary information of the running applications by the Ep-Coord interface; two edge control nodes can share the runtime information of the underlying edge platforms via the EPM-Coord interface.

6.3.3 Use scenario: regional cryptographic implementation

Edge nodes in two different CIDs—either located in different geopolitical regions or belonging to different business enterprises—may employ different cryptographic algorithms and enforce specific security policies. Edge nodes within each CID can interoperate with one another seamlessly. However, a border edge node (BEN) must be installed at the boundary of the two CIDs in order to facilitate the interoperation among the edge nodes in the two CIDs. This BEN must be able to leverage the specific interfaces connected to each of the two CIDs and be trusted by both. Each interface shall implement the set of cryptographic algorithms employed by the CID and enforce the same security policies.

6.4 Service/application domains (SADs)

6.4.1 Introduction

A service/application domain (SAD) is a collection of resources provided on a set of edge nodes to support a specific service/application provided by the same edge service provider. The resources in a SAD follow a set of service level agreements (SLA) and operating rules established by rule framework. Resources including data, analytics models, and virtual functions are the essential elements of a SAD. These resources can reside in one or more physical/logical/virtual edge node(s). Some of them can migrate among these nodes to satisfy a set of QoS policies on workload, resource usage, or network performance.

For example, a video-streaming service provided by the same enterprise may consist of some edge applications hosted by several edge nodes in multiple CIDs. The edge applications and the involved resources are considered an SAD.

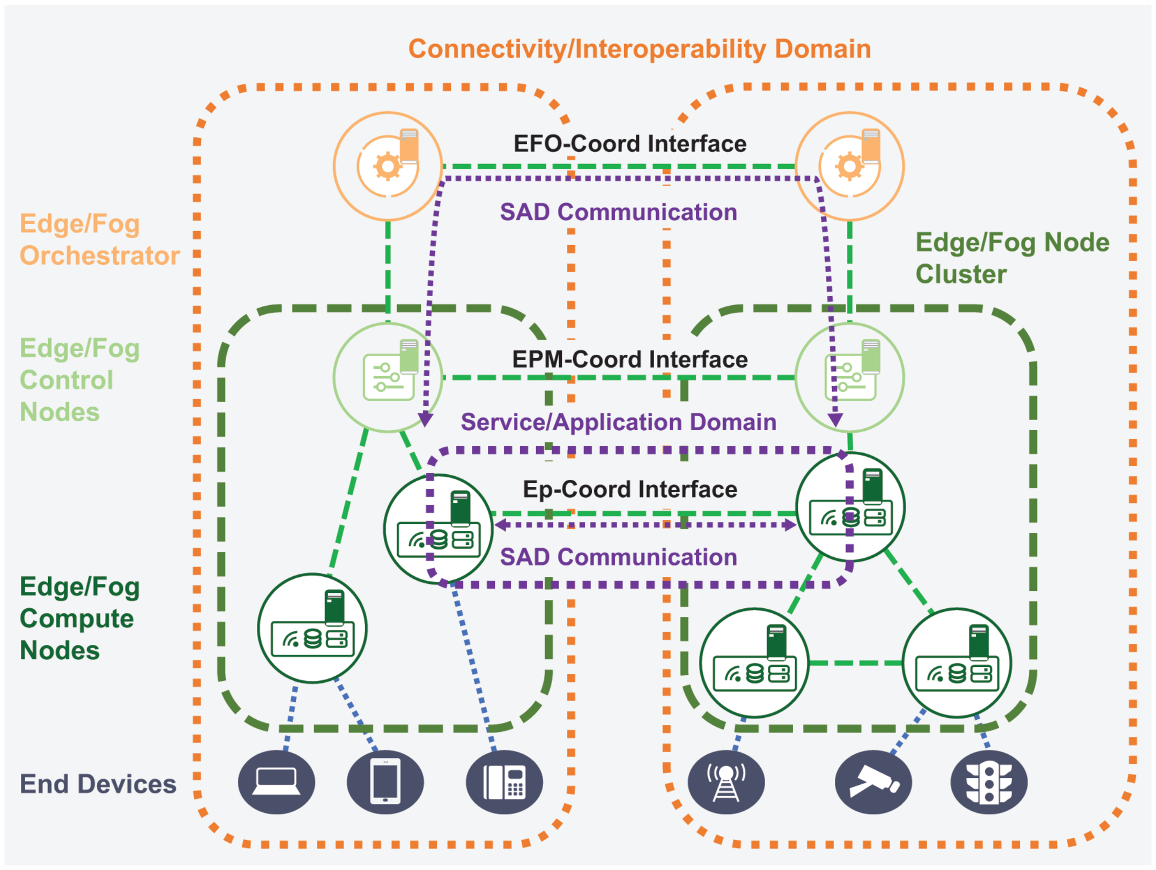

Figure 21—Structure of service/application domain (SAD)

6.4.2 Organization

A SAD may span across multiple CIDs, given that the bridging edge nodes among the CIDs can perform the necessary information translation and interchange. Conversely, a CID can also host multiple SADs; each manages the resources necessary to support a specific service/application. Together, CIDs and SADs form a nested hierarchy for managing the functional and operational behaviors of networked entities in an IoT continuum.

The resources in a SAD for supporting a specific service/application shall be grouped in a distinct namespace that enables the resources allocated to different SADs to reside on the same set of edge nodes while separate from one another in their usage. In this progression, two edge application instances on different edge compute nodes in a SAD can exchange their information and handover computing tasks by performing SAD communication, either via the Ep-Coord or the EFO-Coord interfaces.

6.4.3 Use scenario: application-specific information isolation

An edge service operator may lease the resources in its radio and core networks to several edge service providers. Each edge service provider can offer and onboard its virtual network functions—such as radio resource allocation, software-defined routing, accounting, and billing—by creating its own SAD and running its services in an isolated namespace. They can also enforce different networking policies, authentication, authorization, and accounting (AAA) rules at a fine-grained level in each SAD belonging to different edge service providers by configurating via the EFO.

IEEE SA
STANDARDS
ASSOCIATION

RAISING THE WORLD'S STANDARDS