IEEE P1935 Contribution:

Draft for Edge/Fog Manageability and Orchestration Standard

Contributors:

Working Group draft document (Hung-Yu Wei, as Chair of P1935)

John Zao, Yao Chiang, Tze-Yu Chen, Gary Gan

Version:

Last Update: 2022/02/10

(Draft) IEEE P1935 Standard for Edge/Fog Manageability and Orchestration

1. Introduction

Edge computing as an emerging technology that can host the mobile applications closer to its users, provides lower latency, higher efficient bandwidth and service delivery, as well as better user quality of experience. The innovative mobile applications, such as augmented reality, facial detection, and interactive applications, evolve as 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 of placing the computing capacity at the local area network, fog computing is more often used in Industrialy Internet of Things (IIoT) scenarios. Coming up as a modern solution to catch up with such needs, edge/fog computing is a brand-new and promising paradigm to offer an environment characterized by low latency and necessary resources for mobile devices to liberate them from the computing-intensive and real-time applications.

However, even though the edge devices are closer to their users and thus may provide the advantages described above, their fewer capacities of computation and storage than cloud computing is another fatal issue in practice. Edge/Fog systems are also responsible for maintaining the common operations, handling the application lifecycle, providing a corresponding environment for the mobile services, performing the functionalities of storage and traffic control, and supporting service mobility. To meet these requirements, the management and orchestration of the Edge/Fog computing system become a key topic in the modern scenarios of networking and service delivery, aiming to leverage the 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 the regulation of the transmissions of messages, the controlling and allocation of the resources of the edge devices, and the monitoring and arrangement in 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, be hard to down while facing different challenges, and be as efficient as possible. The Edge/Fog system, especially the Orchestrator Level in the 3-level architecture, which will be further explained in the following sections, 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:

1. 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.
2. It shall be possible to deploy the components, including the EFO, Edge/Fog control nodes, and Edge/Fog compute nodes, on the various hardware devices.
3. The Edge/Fog system shall fit in and communicate with the wireless communication network, dealing with the corresponding traffic routing to the correct components.
4. The Edge/Fog system shall be able to provide Edge services and Edge applications, as well as the corresponding environment for them.
5. 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.
6. The Edge/Fog system shall support the full lifecycle and related operations of Edge applications, such as hosting, onboarding, instantiation, operating, and termination.
7. The Edge/Fog system shall support Edge Service Operators to manually and dynamically access the Edge services and Edge applications on their needs.
8. The Edge/Fog system shall be able to decide the service homing.
9. The Edge/Fog system shall support the compatibility with the authorized 3rd party entities, including processing the corresponding requests.
10. It shall be possible to deploy Edge applications on different Edge/Fog compute nodes without specific arrangement.
11. The Edge/Fog system shall be able to authorize and authenticate the user requests about the Edge applications.
12. The Edge/Fog system shall support the service mobility, that is, keeping connectivity during the UE is moving from the coverage of an Edge/Fog control node to another.
13. The Edge/Fog system shall utilize the related network information to improve its own performance in various matrices.
14. The Edge/Fog system shall support the Edge service to declare its own availability during service discovery.
15. The Edge/Fog compute nodes shall be able to provide computing, networking, and storage resources.

This standard is going to specify the related components, requirements, procedures, and other necessary parts of the management and orchestration in the Edge/Fog system. In the following sections of the standard, we 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 Section 2. The ways to manage and orchestrate the platforms and related resources in Edge/Fog system are explained in Section 3. The procedures and necessary formal files involved in launching and deploying the services and applications on the Edge/Fog system are elaborated in Section 4. The concept of communication among multiple Edge/Fog system are described in Section 5.

2 Architecture and Framework for Edge/Fog Manageability and Orchestration

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

2.1 Edge/Fog Manageability and Orchestration Framework Overview

Figure 2.1-1 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 3rd 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. 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 is (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 [1]. All these entities and components will be described in detail in the following sections.

Table 2.1-1 gives some brief descriptions and the related section to each entity mentioned in Figure 2.1-1, as well as the terms used in Section 2.

[1] For ease of reading and description, we abbreviate the term "Edge/Fog" to "Edge" in the remaining sections of this standard. These terms refer to the completely same things. For example, an Edge/Fog control node can also be written as an Edge control node.

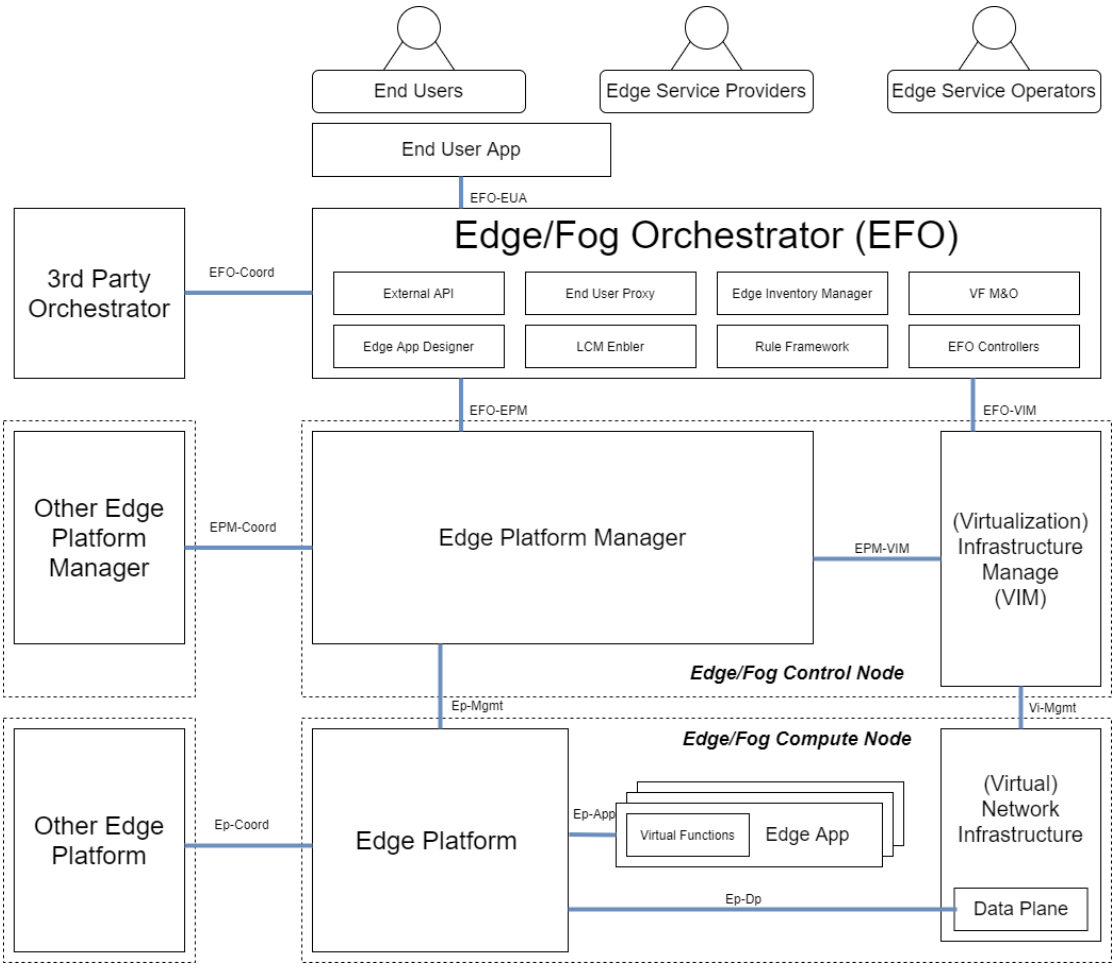

Figure 2.1-1 The basic architecture of the Edge/Fog system framework

Table 2.1-1 The brief description of the entities in the Figure 2.1-1 and Section 2

Entity	Brief Description	Section
End User App	An interface for all end users.	2.5.1
End Users	The users accessing the system via the End User Apps.	2.5.1
Edge/Fog Orchestrator	An orchestrator of the whole Edge system. It can be abbreviated as Edge Orchestrator or just EFO.	2.2.1
Edge Platform Manager	An entity managing the Edge Platform.	2.3.2
(Virtualization) Infrastructure Manager	An entity managing the infrastructure resources named Virtual Network Infrastructure, which can be virtualized or non-virtualized. Usually abbreviated as VIM.	2.3.1
Edge Platform	A platform which handles purposed computing tasks.	2.4.1
Edge App	The applications running on Edge system.	2.4.2
(Virtual) Network Infrastructure	The virtualized and non-virtualized resources which enable network operations. Usually abbreviated as VNI.	2.4.4
Data Plane	The functional entity that handles the	2.4.3

	traffic routes.	
Application Management Entity	The entity managing and coordinating the operation of Edge Applications, such as Edge Platform.	2.3.2
Compute node	Compute node is one entity in the Computer Level that is able to handle the computing tasks on its own.	2.1
Computer Level	Computer Level is the bottom layer of the Edge system which is composed of one or more compute nodes, dealing with the computing tasks.	2.1
Control node	Control node is one entity in the Control Level for management of the related compute nodes.	2.1
Control Level	Control Level is the middle layer of the Edge system which is composed of one or more control nodes, responsible for management and coordination of Computer Level.	2.1
Edge App Designer	A component in the EFO handling the application design and onboarding.	2.1.1
Edge Inventory Manager	A component in the EFO for the management of the inventory in the Edge system.	2.1.1
Edge Service Operator	The service operator dealing with the management and operation of Edge Applications.	2.1
Edge Service Provider	The designer and onboarder of Edge Applications.	2.1
EFO Controllers	The components in the EFO responsible for coordinating and configuration of virtual network functions (VNFs) and related resources.	2.1.1
External API	External API in the EFO is provided for the Edge Service Operators or other entities to access the functionalities of the EFO.	2.1.1
End User Proxy	A component in the EFO as an interface between the EFO and the End User App.	2.1.1
End Users	All users that access the EFO via the End User Apps.	2.1
Inventory	Inventory is the general name for the assets, resources, services, and products in the Edge system.	2.1.1
Lifecycle Manager Enabler (LCM Enabler)	A component in the EFO which is responsible for the management of the application lifecycle.	2.1.1

Orchestrator Level	The top layer of the Edge system for management and orchestration of the whole system, including Controller Level and Computer Level.	2.1
Resource Management Entity	The entity managing the resources in the Edge system, such as VIM.	2.3.1
Rule Framework	A component in the EFO dealing with all conditions, requirements, and reaction policies.	2.1.1
Virtual Function Management and Orchestration	A component in the EFO responsible for the start-up and operation of the application. It can be abbreviated as VF M&O.	2.1.1

2.2 Edge Orchestrator Level Entities

The Edge/Fog orchestrator 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 (EFO) is the core functionality in orchestrator level management. There are multiple ways to access the EFO described in the following section in details.

2.2.1 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 the 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, which 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 Enabler (LCM Enabler) component, is responsible for designing and managing the application lifecycle.

The run time framework executes the rules and polices distributed by the design and creation environment, the Edge App Designer component in this case, and controllers which manage resources corresponding to their assigned controlled domain.

- Virtual Function Management and Orchestration (VF M&O) component will follow the BPMN (Business Process Model and Notation) flows which operate on the models distributed from Edge App Designer. VF M&O will use the models which 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) component 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 on-boarding, instantiation, termination of 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 allowed operations according to their assigned roles.

Additionally, external API are also provided for the access to core network and from external user applications in order to execute specific operations and access the functionalities of the EFO.

2.3 Edge Controller Level Entities

These entities located in 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 (RME) and Application Management Element (AME). The function and responsibility of each class of entities is described in following sub-sections.

2.3.1 Resource Management Elements

Resource management elements focus on management and storage of network infrastructure. They are able to 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. The more information about the resources in the Edge system can be found in Section 3.

2.3.1.1 (Virtualization) Infrastructure Manager

The (Virtualization) Infrastructure Manager (VIM) 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.

2.3.2 Application Management Elements

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

2.3.2.1 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 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 as VNF instances. It is possible to deploy more than one Edge Application LCM instance.

2.4 Edge Computer Level Entities

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 which provides compute, storage, and network resource for the Edge applications.

2.4.1 Edge Platform

The Edge platform is responsible for the Edge functions which 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 and configures DNS proxy/server according to the traffic rules prescribed by the Edge platform manager.

2.4.2 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 single or multiple virtual function(s), which can be managed and orchestrated by the VF M&O component in EFO. It is possible either for an Edge App to provide several services, or for multiple Edge Apps to provide a single service.

2.4.3 Data Plane

The NFVI 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 VIM.

2.4.4 (Virtual) Network Infrastructure

The (Virtual) Network Infrastructure (VNI) is 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 a VIM.

2.5 External Entities

This section describes the elements which connect the external environment. Explanations of how the Edge architecture communicates with other existing standards are presented.

2.5.1 End User Application

End User Applications (EUA) are able to 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.

2.6 Interfaces

This section lists and describes the interfaces and the reference points between entities inside and outside of Edge system. The interfaces can be classified based on the highest level of the connected entities. That is, an interface between Orchestrator Level and Control Level is classified as an Orchestrator interface.

2.6.1 Orchestrator Interfaces

- The “EFO-EPM” reference point between the EFO and the Edge platform manager is used for the application management, including application lifecycle, rules and requirements, checking Edge service availability, and exchanging related notification between the EFO and the Edge platform.
- The “EFO-VIM” reference point between the EFO and the VIM is used for management of the 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 run-time configuration to satisfy a set of QoS policies on workload, resource usage, or network performance. It also

support the operation of the necessary files, such as manifest files and packages.

- The “EFO-Coord” reference point between the EFO and a 3rd party orchestrator is used to exchange necessary information about the underlying Edge nodes, the implemented rules, and the approach to interpret the messages.

Table 2.6.1-1 The related sections of the Orchestrator interfaces

Interface	Related Sections	Responsibility
EFO-EPM	4.2.2	Service deploy requests, workload instantiation, application activation, inventory update
	4.2.3, 4.2.4	Context creation requests and responses, context deletion requests and responses
	4.2.5	Service termination requests, workload termination, application deactivation, inventory deletion,
	4.2.6	Application reconfiguration, inventory update
EFO-VIM	3.2.1-3.2.6	Resource requests and responses (including resource creation, resource status query, resource discovery, resource reconfiguration, resource deletion)
	4.2.1	Store the images
	4.2.2	Resource allocation, infrastructure instantiation
	4.2.5	Resource release
EFO-EUA	4.2.1	Manifest file onboarding and queries, package distribution requests, notification
	4.2.3	Application list queries, context creation requests
	4.2.1-4.2.6	Application requests (including onboarding, instantiation, termination, and reconfiguration) and context requests (including context creation and deletion)
EFO-Coord	5.2	Communication between two EFOs

2.6.2 Controller Interfaces

- 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 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 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 EPM belonging to different Edge Service Operators can also leverage this interface to share their information.

Table 2.6.2-1 The related sections of the controller interfaces

Interface	Related Sections	Responsibility
Ep-Mgmt	4.2.2	Application creation, configuration requests, inventory update
	4.2.3	Context creation requests and responses
	4.2.4	Context deletion requests and responses
	4.2.5	Configuration requests, application deletion, inventory deletion
	4.2.6	Application reconfiguration
Vi-Mgmt	3.2.1	Resource creation
	3.2.4	Resource replacement, resource update
	3.2.5	Resource deletion
	4.2.1	Store the images
	4.2.2	Assign application packages
	4.2.5	Infrastructure termination
EPM-VIM	4.2.5	Infrastructure termination requests
	4.2.6	Infrastructure update requests
EPM-Coord	5.2	Communication between two EPMs

2.6.3 Computer Interfaces

- 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 the 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 2.6.3-1 The related sections of the Orchestrator interfaces

Interface	Related Sections	Responsibility

Ep-App	4.2.2-4.2.6	Application creation and deletion, configuration, context creation and deletion
Ep-Dp	4.2.2	Assign application packages It is also involved in some pre-advance preparation, such as images onboarding, application scaling, and container installation.
Ep-Coord	5.2	Communication between two Edge platforms

3 Edge Platform Management and Orchestration

The Edge platforms are responsible for handling various tasks, including computing, networking, and storage tasks. 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 section describes the RESTful service APIs of the resource operations which are used for the communication of these components.

3.1 Overview of Edge Platform Resource Management

The service APIs are used for the resource management and communication between the components. As shown in Figure 3.1-1, 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 resources, networking resources, and storage resources in the system, as well as the virtualized infrastructure.

There are totally five types of API supported:

(1) 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.
(2) Resource status query: This operation API is used to fetch the information of the certain resource. The HTTP method GET is supported.
(3) Resource discovery: This operation API is used to search the resources that meet the specific filtering rules. The HTTP method GET is supported.
(4) 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, which are used to replace the whole information and to update partial information respectively.
(5) Resource deletion: This operation API is used to remove the target resource in the Edge system. The HTTP method DELETE is supported.

Figure 3.1-1 The common operations of the Edge system resource management

These service APIs for the Edge platforms are RESTful HTTP APIs, which should be independent of technology implementation. All the resources in the Edge system have their URLs to be called by the APIs. The specification of service APIs should 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 should be specified if exist.

The components that are mentioned in Section 3.2 are listed in Table 3.1-1. The operation APIs can be sent from the internal VF M&O. It may also possible for some external API clients which are able to send HTTP requests to send the requests, but additional verification and authorization are needed in that case. In this section, we only discuss the situation where the request is sent from the VF M&O.

Table 3.1-1 The components mentioned in Section 3.2

Types	**Components**	**3.2.1**	**3.2.2**	**3.2.3**	**3.2.4**	**3.2.5**
Users	Service Provider					
	Operator					
	End User App					
EFO	External API					
	End User Proxy					
	Edge App Designer					
	VF M&O	V	V	V	V	V
	Edge Inventory Manager					
	LCM Enabler					
	Rule Framework					
	EFO Controllers					
Control nodes	Edge Platform Manager					
	VIM	V	V	V	V	V
Compute nodes	Edge Platform					
	Edge App					
	VNI	V				V
	Data Plane					

Table 3.1-2 The terms used in the operation APIs

Terms	**Descriptions**	**Sections**
Attribute	Attributes are the additional information of the resource.	3.2.4
Edge resource	Edge resources are the resources created in	3.2.1

	the Edge system. These resource can be further sliced as the resource quota.	
Filtering rule	Filtering rules are the restrictions and conditions to narrow down the search results.	3.3
Physical resource	Physical resources are the physical hardware of the Edge system, for example, the physical servers.	3.1
Property	Property means the characteristics of the resource.	3.2.4
Resource	Resources are the physical and virtual assets that can be leveraged by the Edge system.	3.1
Resource configuration	The resource configuration is a part at the end of the resource creation operation.	3.2.1
Resource creation	It is the operation used to create new resources in the Edge system.	3.2.1
Resource deletion	It is the operation used to delete resources in the Edge system.	3.2.5
Resource discovery	It is the operation used to search and return the list of resources matching the filtering rules.	3.2.3
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.	3.2.1
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.	3.2.4
Resource replacement	It is one of the two types of modification operation which replaces whole information of the resource.	3.2.4
Resource status query	It is the operation used to get the current information of the target resource.	3.2.2
Resource update	It is one of the two types of modification operation which only updates partial information of the resource.	3.2.4
Resource URL	Resource URLs are the unique addresses of the resources for the Edge system to locate.	3.1
Virtual resource	Virtual resources are the non-physical resources in the Edge system, including computing resources, networking resources, storage resources, etc.	3.1

3.2 Resource Manageability Procedures

This section describes the service APIs which can be used to do several resource

operations, including resource creation, resource status query, resource discovery, resource reconfiguration, and resource 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 is processed by the VIM in the most cases. In Section 3.2.1 to 3.2.5, we will describe the usage, procedures, as well as the format of the requests and responses of each operation. In Section 3.2.6, we list the corresponding HTTP status code for the failed requests.

3.2.1 Resource Creation

New resources can be created by sending an HTTP POST request to the system. The POST request should contain the information about the new resource that will be created. The new resources called as “Edge resources” will be given unique identifier (ID) named the resource URL. After creation, these Edge resources will be configured automatically based on the request payload as well. The resource may be split to several pieces or slices, called as “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 as the resource quota of the physical resource (that physical server). The other details of the configuration process vary from system to system, and is 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 connect the virtual resources to the physical resources which host them. Such connection is shown in a way where the virtual resources are the child resources of the physical resources as the 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.

3.2.1.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The target resource is not yet created in the Edge system.

The post-condition of the procedure:

(1) The resource has been created and configured as network infrastructure in the Edge system.

3.2.1.2 Procedures

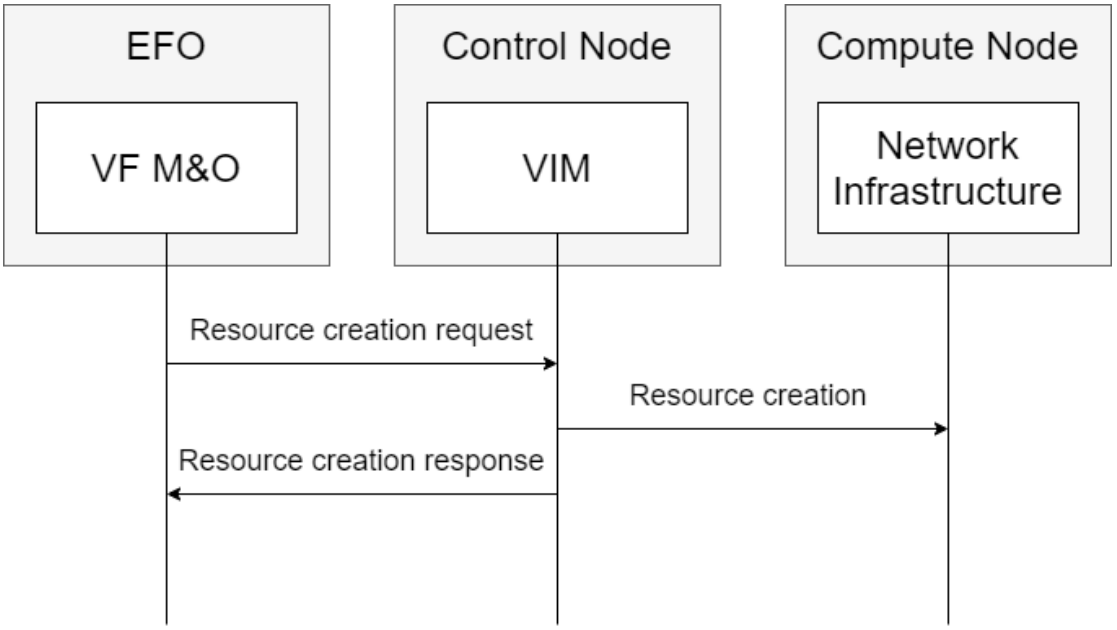

Figure 3.2.1.2-1 The workflow of the resource creation

1. The VF M&O sends a resource creation request to the VIM (Table 3.2.1.3-1).
2. The VIM computes the resource URL and then creates the resource as the Network Infrastructure. The 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.
3. After the resource is created either successfully or failingly, the VIM sends the response to the VF M&O (Table 3.2.1.3-2 & Table 3.2.6-1).

3.2.1.3 Requests and Responses

If the server creates the new resource successfully, the HTTP response status “201 created” should be returned, which includes the path information of the created resource (Table 3.2.1.3-1). On failure, the specific status error code should be returned (Table 3.2.6-1).

Table 3.2.1.3-1 Resource creation request

Element	Description
Request Type	HTTP POST
Parent Resource ID	The identifier of the parent resource of the created resource. It should be contained in the URL.
Request Payload	The payload contains the information and data of the created resource.

Table 3.2.1.3-2 Resource creation successful response

Element	Description	
HTTP Status Code	201 Created	
HTTP Header	URL	The URL of the created resource.
Response Payload	(Empty)	

3.2.2 Resource Status Query

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 Section 3.3. 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 Section 3.4 to avoid race condition.

3.2.2.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The target resource has been created in the Edge system.

The post-condition of the procedure:

(1) The entity which sent the request gets the information of the target resource.

3.2.2.2 Procedures

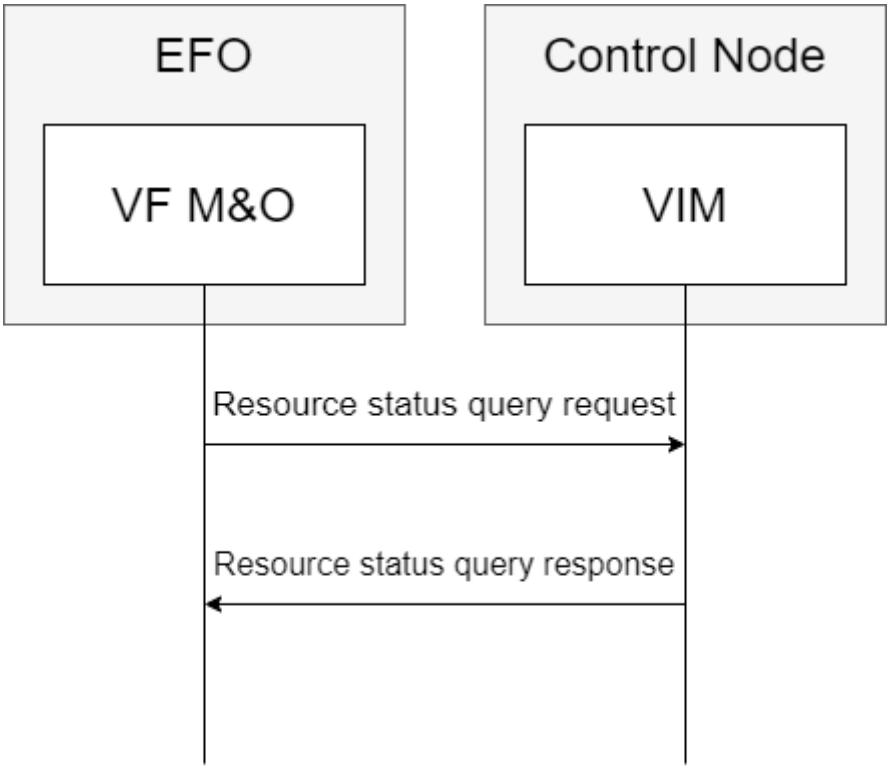

Figure 3.2.2.2-1 The workflow of the resource status query

1. The VF M&O sends the resource status query request to the VIM in order to query the target resource.
2. If the specific resource is found, the information of the requested resource should be returned to the VF M&O as an HTTP response (Table 3.2.2.3-2). On

failure, the specific error status code should be returned (Table 3.2.6-1).

3.2.2.3 Requests and Responses

On success, the “200 OK” status code should be returned with information of the requested resource (Table 3.2.2.3-2); on failure, the specific error status code should be returned (Table 3.2.6-1).

Table 3.2.2.3-1 Resource status query request

Element	Description
Request Type	HTTP GET
Resource ID	The identifier of the target resource. It should be contained in the URL.
Request Payload	(Empty)

Table 3.2.2.3-2 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.

3.2.3 Resource Discovery

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 is able to return all the resources that match the filtering rules. If there is no resource that matches the filtering rules, an empty list should be returned.

As 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.

3.2.3.1 Conditions and Involved Information

The pre-condition of the procedures:

(1) The target resource has been created in the Edge system.

The post-condition of the procedures:

(1) The entity which sent the request receives all the resources matching the filtering rules as well as their information.

3.2.3.2 Procedures

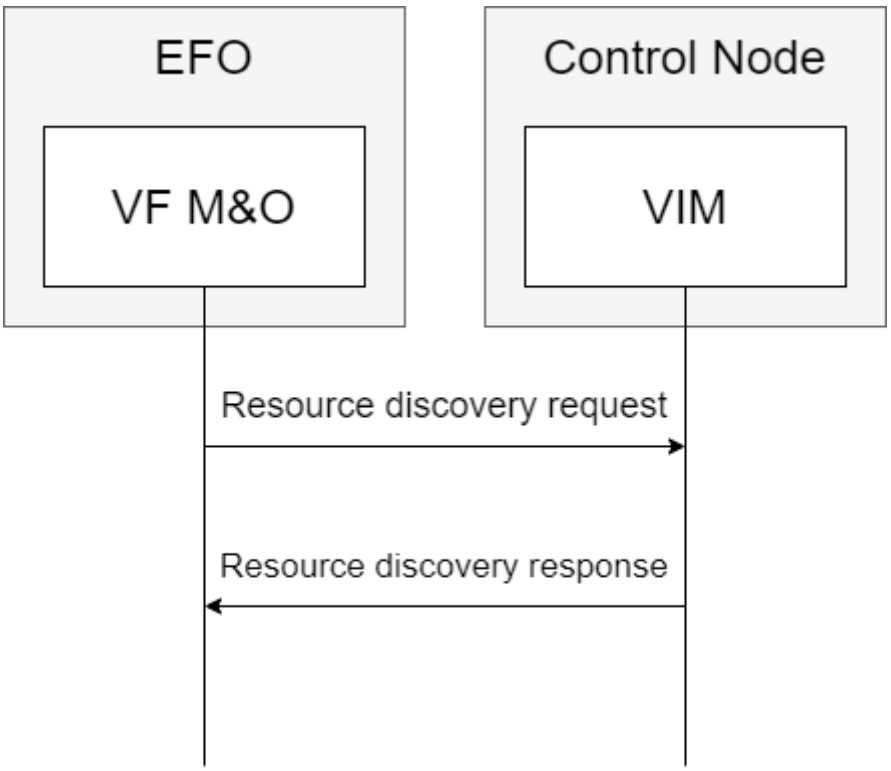

Figure 3.2.3.2-1 The workflow of the resource discovery

1. The VF M&O sends a resource discovery request to the VIM (Table 3.2.3.3-1).
2. The VIM searches and filter the resources to return a list of resources which satisfy the given filtering rules as an HTTP response to the VF M&O (Table 3.2.3.3-2). On failure, the specific error status code should be returned (Table 3.2.6-1).

3.2.3.3 Requests and Responses

On success, the “200 OK” status code should be returned with the payload that contains the specific resource data, adjusted according to the passed parameters (Table 3.2.3.3-1). On failure, the specific error status code should be returned (Table 3.2.6-1). If no resource matches the filtering rules, it is considered as the failure case.

Table 3.2.3.3-1 Resource discovery request

Element	Description
Request Type	HTTP GET
Filtering Rules	The restrictions and conditions to narrow down the search results. They should be contained in the URL.
Request Payload	(Empty)

Table 3.2.3.3-2 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 should be in JSON format.

3.2.4 Resource Reconfiguration

Resource reconfiguration means the modification of the properties or attributes of the target resource. These properties and attributes are generated during the resource creation operation automatically, 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 the resource replacement, the resources can be overwritten by the HTTP PUT method with a resource representation. That is, all the old resource content will be omitted after resource replacement. For the 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 should be performed to ensure the version of the resource, in order to avoid race conditions.

As 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.

3.2.4.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The target resource has been created in the Edge system.

The post-condition of the procedure:

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

3.2.4.2 Procedures - Replacement

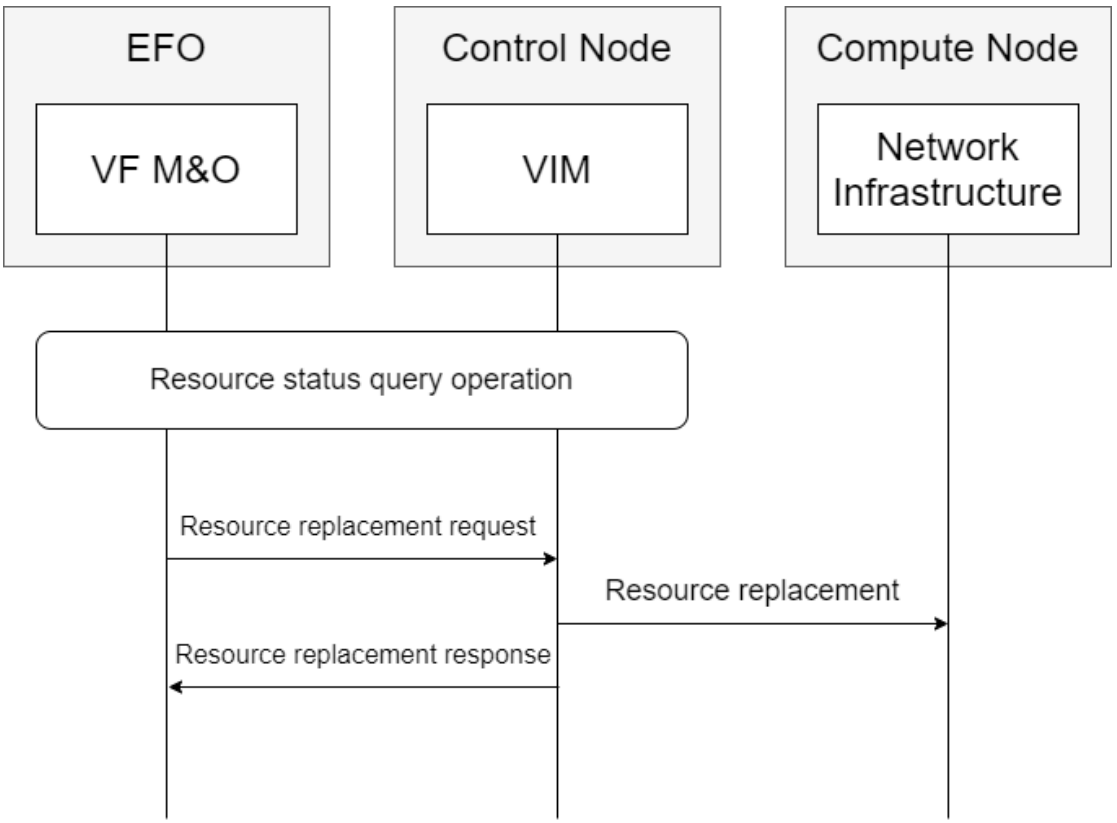

Figure 3.2.4.2-1 The workflow of the resource replacement

1. 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 Section 3.2.2.2. If the specific resource is found, the information of the requested resource should be returned. If not, the process is terminated as there is no resource to replace.
2. The VF M&O sends the resource replacement request (Table 3.2.4.4-1) which contains the version number obtained in Step 1 to the VIM.
3. The VIM checks the version number of the resource. If the version number is valid, the VIM will replace the target resource. If not, an error message will be returned. If needed, the related resources will be reconfigured as well.
4. The VIM returns the success or error message to the VF M&O (Step 3.2.4.4-3, 3.2.4.4-4, or Table 3.2.6-1).

3.2.4.3 Procedures - Update

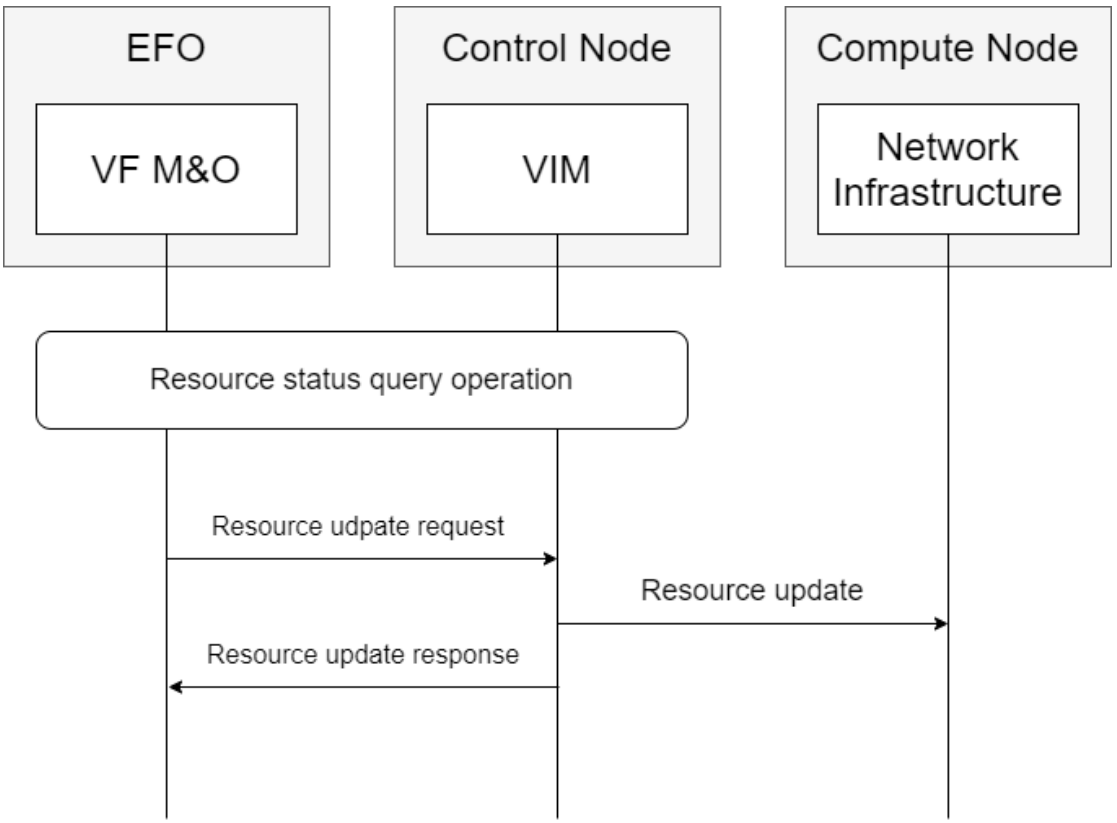

Figure 3.2.4.3-1 The workflow of the resource update

The request can be sent from the API client or the VF M&O. If the request is sent from the VF M&O, the procedures skip Step 2-5 and then end in Step 8.

1. 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 Section 3.2.2.2. If the specific resource is found, the information of the requested resource should be returned. If not, the process is terminated as there is no resource to replace.
2. The VF M&O sends the resource update request (Table 3.2.4.4-1) which contains the version number obtained in Step 1 to the VIM.
3. The VIM checks the version number of the resource. If the version number is valid, the VIM will update the target resource. If not, an error message will be returned. If needed, the related resources will be reconfigured as well.
4. The VIM returns the success or error message to the VF M&O (Step 3.2.4.4-3, 3.2.4.4-4, or Table 3.2.6-1).

3.2.4.4 Requests and Responses

On success, either “200 OK” or “204 No Content” should be returned (Table 3.2.4.4-3 & 3.2.4.4-4); if a client attempts to modify the resource with an expired version number, “412 Precondition Failed” should be returned. On other failures, the specific status error code should be returned (Table 3.2.6-1).

Table 3.2.4.4-1 Resource replacement request

Element	Description
Request Type	HTTP PUT
Request Payload	The new data and information of the target resource. The 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 3.2.4.4-2 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 the race conditions.

Table 3.2.4.4-3 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 3.2.4.4-4 Resource reconfiguration successful response with empty payload

Element	Description
HTTP Status Code	204 No Content
HTTP Header	(Empty)
Response Payload	(Empty)

3.2.5 Resource Deletion

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 the other operations, the resource discovery operation is suitable for both physical and virtual resources. The system can delete the target resource based on their needs and the 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 virtual resources removes the resource actually; however, it is possible to re-create them as long as the physical resource is still in the Edge system.

3.2.5.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The target resource has been created in the Edge system.

The post-condition of the procedure:

(1) The target resource will be removed from the Edge system. The related resources are reconfigured.

3.2.5.2 Procedures

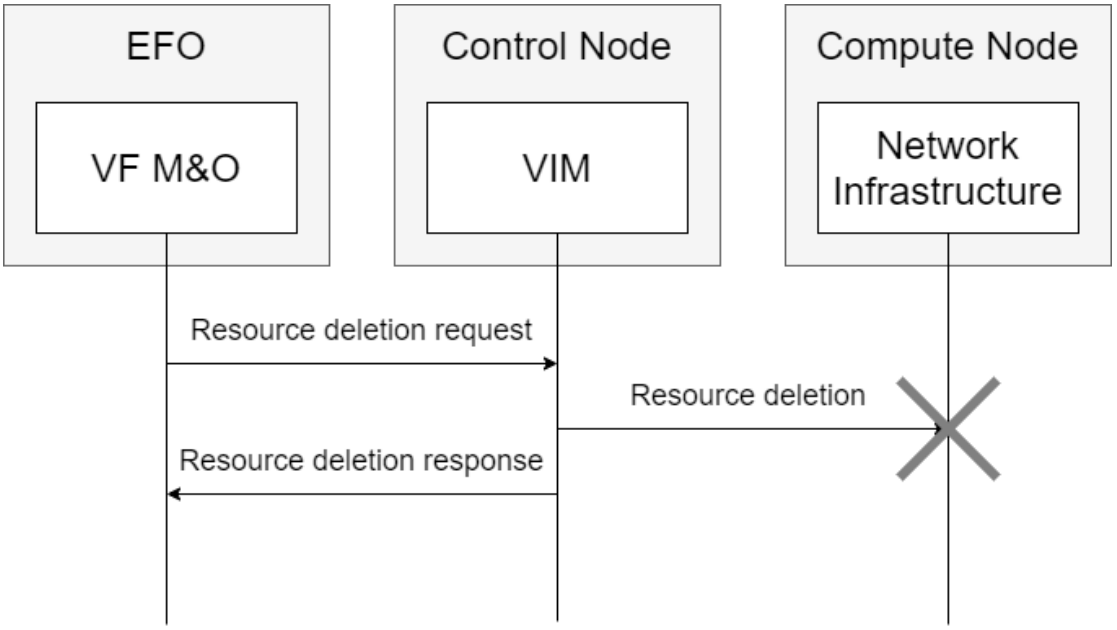

Figure 3.2.5.1-1 The workflow of the resource deletion

1. The VF M&O sends a resource deletion request to the VIM (Table 3.2.5.3-1).
2. The VIM deletes the target resource as the Network Infrastructure, and clean 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.
3. The VIM returns the deletion success or error message to the VF M&O (Table 3.2.5.3-2, 3.2.5.3-3, or Table 3.2.6-1).

3.2.5.3 Requests and Responses

On success, if the payload is empty, “204 No Content” should be returned, otherwise “200 OK” should be returned (Table 3.2.5.3-2 & 3.2.5.3-3). On failure, the specific error code should be returned (Table 3.2.6-1).

Table 3.2.5.3-1 Resource deletion request

Element	Description
Request Type	HTTP DELETE
Request Payload	(Empty)

Table 3.2.5.3-2 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 3.2.5.3-3 Resource deletion successful response with empty payload

Element	Description
HTTP Status Code	204 No Content
HTTP Header	(Empty)
Response Payload	(Empty)

3.2.6 Responses for Request Failure

In previous subsections, we have listed the requests and responses in success cases. In this section, we discuss about the failure cases and list the corresponding status code for such responses. If not specified, these responses are common for all operations we have mentioned in Section 3.

Table 3.2.6-1 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 to perform it.
404 Not Found	This status code is used when the requested URL is invalid in the system, including the cases that 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 the edit conflict.
412 Precondition Failed	This status code is used if a client attempts to modify the resource with an expired version number.

4 Edge Application Management and Orchestration

In order to provide the complete lifecycle management of Edge Applications, the system has to have knowledge about the application run-time information, and thus it is necessary to specify the procedures and the message flows between all components involved. This section describes the managing procedures and message flows of applications at Edge. Compared to Section 3, this section more focus on how to run and coordinate Edge Applications in the Edge system.

4.1 Overview of Edge Application Management

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 to 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).

Fig 4.1-1 the diagram of lifecycle procedures of Edge Applications.

As shown in Fig 4.1-1, a typical application has the following lifecycle management procedures:

(1) Application on-boarding: In the application on-boarding procedure, the Edge management needs to obtain the necessary software files and description files that are necessary to start running applications.
(2) Application instantiation: Based on the application blueprint file, which 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.
(3) 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.
(4) Context deletion: Edge Service Providers are able to remove the certain application context (such as user data) which they don’t need anymore.
(5) 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.
(6) 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 even the Edge Platform Manager.

It is common for a standard Edge Application to undergo various lifecycle procedures, including on-boarding, 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, which corresponds to context creation and context deletion procedures. An instance is a concrete occurrence of an Edge Application which 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 hasn't been terminated.

In the process, Edge Service Providers offer their applications and services onto the Edge system by uploading the specific manifest files, and 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 the different groups of people.

The manifest file is necessary for the 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 and 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 4.1-3 and the related subsections.

In addition, it is common for the entities and components to send requests and responses to make sure that the progress goes properly. Table 4.1-1 shows the default format of a request/response. The format of those with specific needs is described in each section respectively.

Table 4.1-1 Default format of a request/response

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

Table 4.1-2 The components mentioned in the 4.2 section

Types	Components	4.2.1	4.2.2	4.2.3	4.2.4	4.2.5	4.2.6
Users	Service Provider	O					

	Operator	O	O			O	O
	End User App			O	O		
EFO	External API		O			O	O
	End User Proxy			O	O		
	Edge App Designer	O					
	VF M&O	O	O	O	O	O	O
	Edge Inventory Manager	O	O	O		O	O
	LCM Enabler	O					
	Rule Framework	O					
	EFO Controllers	O					
Control nodes	Edge Platform Manager	O	O	O	O	O	O
	VIM	O	O			O	O
Compute nodes	Edge Platform		O	O	O	O	O
	Edge App		O	O	O	O	O
	VNI & Data Plane	O	O			O	

Table 4.1-3 The Files used in the Lifecycle Procedures

Files	Descriptions	Sections
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.	4.2.3.1
Application package	Application packages are ones of the files that are 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.	4.2.1.1
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.	4.2.1.1
Blueprint file	A blueprint file is a file that includes the orchestration rules and the resource lifecycle management designs, etc. This file is used to configure the components in the EFO.	4.2.1.1
Inventory	Inventory are the network information, topology, and metadata that are about the Edge system environment and are stored in the	4.2.4.1

	Edge Inventory Manager component.	
Manifest file	Manifest files are the files used in the application onboarding procedures. A manifest file contains an application package and a blueprint file.	4.2.1.1
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.	4.2.1.1

4.2 Manageability and Orchestration Procedures

The details of the service and message design flow of each procedure are presented in this section. 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 subsection in detail. Each subsection 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.

4.2.1 Application On-boarding

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

Operators and Service providers mentioned in this section are assumed either by the same or different people/groups/organizations. That is, if they are assumed by the same people, they don’t need to perform the query request (Step 2 in section 4.2.1.2) before the distribution request (Step 3 in section 4.2.1.2). 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.

4.2.1.1 Conditions and Involved Information

The pre-condition of the procedure:

(2) The Edge Service Providers should prepare the manifest files.
(3) 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:

(2) 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 Section 4.2.1.4.
(3) The application packages are saved to the VNI via the VIM.

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

(1) Manifest file: a manifest file that includes an application package and a blueprint file.

(2) 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.

(3) VF image: a virtual function. An application package is composed of one or more virtual functions called VF images.

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

(5) Artifacts: all the files that are generated by the Edge App Designer component after the manifest file is passed to the Edge App Designer. They are the descriptor files for the components in the EFO and will be distributed to the other components in the EFO.

4.2.1.2 Procedures

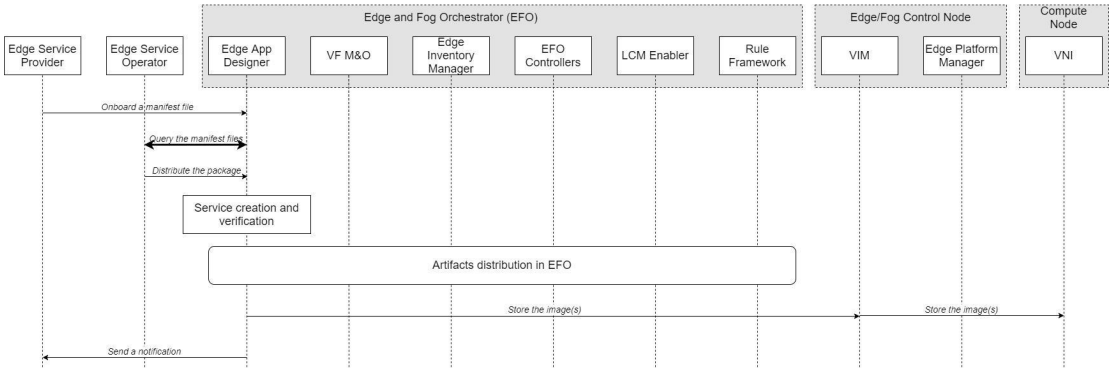

Figure 4.2.1.2-1 the workflow of application onboarding procedures

The detailed description of the flow is documented below:

1. The Edge Service Provider on-boards a manifest file that includes an application package and a blueprint file via the Edge App Designer component of the EFO (Table 4.2.1.3-1). The Edge App designer checks if the Edge Service Provider is authorized to perform the uploading operation. A response should be made if successful (Table 4.2.1.3-2).
2. (Optional) the Edge Service Operator sends a query request (Table 4.2.1.3-3 and 4.2.1.3-4) 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.
3. The Edge Service Operator sends a distribution request (Table 4.2.1.3-5) to trigger the distribution of the packages and blueprint files contained in the selected manifest file.
4. 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.

5. 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.
6. All the artifacts are distributed to their responsible components in the EFO from the Edge App Designer component. The example can be found in Section 4.2.1.4.
7. The application package is saved to the VNI which is responsible for storage via the corresponding VIM.
8. The Edge App Designer sends a notification to the Edge Service Operator for the success of the distribution (Table 4.2.1.3-6).

4.2.1.3 Requests and Responses

The following tables describe the elements contained in the request.

Table 4.2.1.3-1 Application manifest file uploading request

Element	Description
Service Provider ID	The Edge Service Provider identifier.
Manifest file	A manifest file that includes an application package and a blueprint file. It is the payload of the request.

Table 4.2.1.3-2 Application manifest file uploading response

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

Table 4.2.1.3-3 Uploaded manifest files query request

Element	Description
Operator ID	The Edge Service Operator identifier. The ID should be authorized by the EFO.
Query filter	The filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations.

Table 4.2.1.3-4 Uploaded manifest files query response

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

Table 4.2.1.3-5 Artifact distribution request

Element	Description

Operator ID	The Edge Service Operator identifier. The ID should be authorized by the EFO.
Manifest file ID	The identifier of the manifest file which the Edge Service Operator supposes to distribute.

Table 4.2.1.3-6 Distribution success notification

Element	Description
Operator ID	The Edge Service Operator identifier.
Application ID	The packages are distributed successfully, and the application is given an identifier for the Edge Service Operators to instantiate it in the next section.

4.2.1.4 Artifact distribution

In this section, we describe an example of the process of artifact distribution. The artifact distribution is mentioned in Step 6 of Section 4.2.1.2, whose purpose is to classify and send the relevant files to the corresponding component in the EFO.

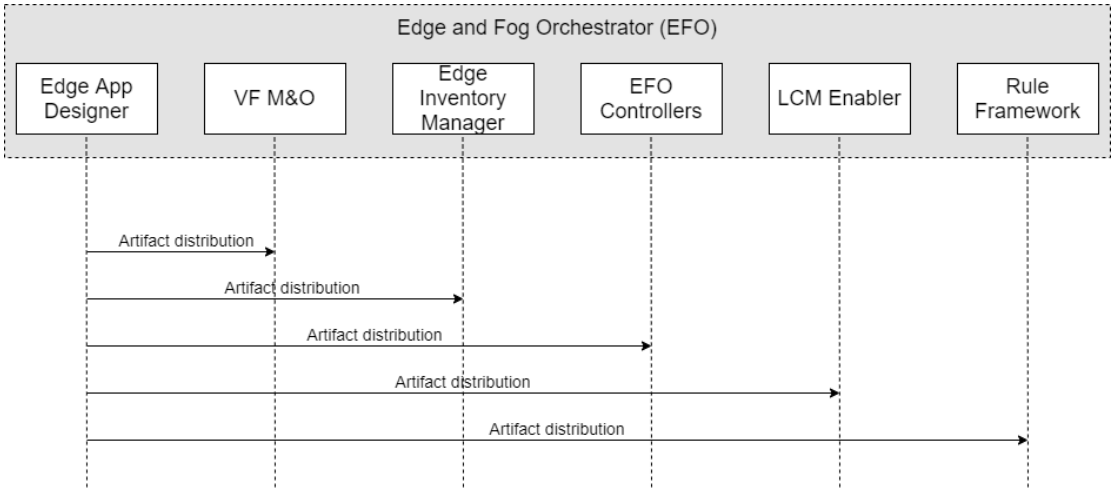

Figure 4.2.1.4-1 A schematic diagram of the workflow of artifact distribution.

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 run-time. 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.

4.2.2 Application Instantiation

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.

4.2.2.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The artifacts have been distributed to each component in the EFO.
(2) The application package has been saved to the VNI.

The post-condition of the procedure:

(1) The Edge Application has been instantiated following the steps described below.
(2) The inventory is created and updated based on the information of the created Edge Application instance.
(3) The Edge Platform has been configured to allocate its computing and networking resources properly.

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

(1) Workload: the workload covers all the pre-work needed for the application to launch properly.
(2) The inventory: the network information, topology, and metadata that are about the Edge system environment and are stored in the Edge Inventory Manager component.

4.2.2.2 Procedures

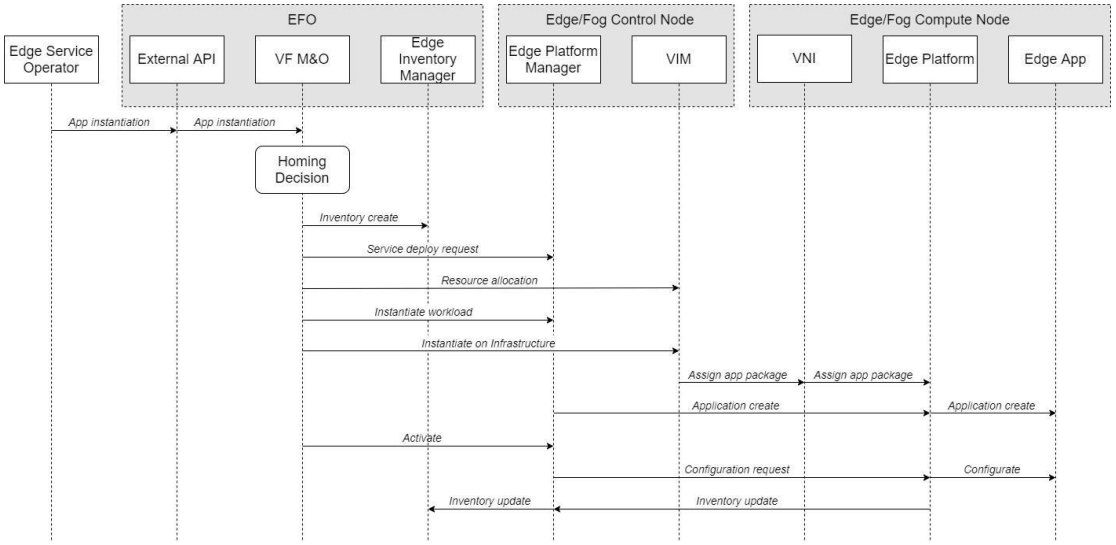

Figure 4.2.2.2-1 the workflow of application instantiation procedures

The workflow of the application instantiation procedures is documented below:

1. The Edge Service Operator sends an application instantiation request (Table 4.2.2.3-1) to the VF M&O component through External API. A response should be made (Table 4.1-1).
2. The VF M&O makes the homing decision of the instance to be instantiated. The homing decision is to decide which Edge Platform the Edge Application will be instantiated on.
3. The VF M&O creates the empty inventory in the Edge Inventory Manager (Table 4.2.2.3-2). The empty inventory will be fulfilled with real values after the Edge

Application is created.

4. The VF M&O sends a service deployment request to the target Edge Platform Manager (Table 4.2.2.3-4). A response should be made if successful (Table 4.1-1).
5. The VF M&O requests the VIM to allocate resources, including computing resource and networking resource (Table 4.2.2.3-5). A response should be made if successful (Table 4.1-1).
6. The VF M&O requests the Edge Platform Manager entity to instantiate the workload (Table 4.2.2.3-6). A response should be made if successful (Table 4.1-1).
7. The VIM deals with the instantiation work on the infrastructure (Table 4.2.2.3-7). In the process, the VIM loads the needed application package from the VNI and deploy it onto the Edge Platform (Table 4.2.2.3-8 & 4.2.2.3-9). A response should be made if successful (Table 4.1-1).
8. The Edge Platform Manager informs the Edge Platform assigned by the EFO to create the Edge Application (Table 4.2.2.3-10). A response should be made if successful (Table 4.1-1).
9. The VF M&O requests the Edge Platform Manager to activate the instance (Table 4.2.2.3-11). The Edge Platform Manager informs the corresponding Edge Platform to deal with the configuration work on the instance (Table 4.2.2.3-12). A response should be made if successful (Table 4.1-1).
10. The Edge Platform returns the information to the Edge Platform Manager to update the inventory in Edge Inventory Manager (Table 4.2.2.3-3).

4.2.2.3 Requests and Responses

Table 4.2.2.3-1 Application instantiation request

Element	Description
Operator ID	The Edge Service Operator identifier. The ID should be authorized by the EFO.
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-2 Inventory update request from an Edge Platform to an Edge Platform Manager

Element	Description
Edge Platform ID	The Edge Platform identifier.
Application ID	The identifer of the application that has been instantiated successfully.
New inventory	The empty inventory which will be updated after instantiation finished.

Table 4.2.2.3-3 Inventory update request from an Edge Platform Manager to the Edge Inventory Manager in an EFO

Element	Description
Edge Platform Manager ID	The Edge Platform Manager identifier.

Updated inventory	The updated inventory. It should fulfill the empty inventory that has been created with parameters and values of the real environment.

Table 4.2.2.3-4 Service deployment request

Element	Description
Edge Platform Manager ID	The Edge Platform Manager identifier.
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-5 Resource allocation request

Element	Description
Edge Platform Manager ID	The Edge Platform Manager identifier.
VIM ID	The VIM identifier.
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-6 Workload instantiation request

Element	Description
Edge Platform Manager ID	The Edge Platform Manager identifier.
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-7 Infrastructure instantiation request

Element	Description
VIM ID	The VIM identifier.
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-8 App package assignment request from VIM to VNI

Element	Description
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.
Edge Platform ID	The Edge Platform identifier that the VIM tells the VNI to assign the application package to.

Table 4.2.2.3-9 App package assignment request from VNI to Edge Platform

Element	Description
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.
Application package	The application package for the target application to be instantiated.

Table 4.2.2.3-10 Edge App creation request

Element	Description
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-11 Edge App activation request

Element	Description
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.

Table 4.2.2.3-12 Edge App configuration request

Element	Description
Application ID	The identifer of the application that the Edge Service Operator supposes to instantiate.
Configuration details	The configuration of the Edge App. The details of the configuration vary depending on the system and the Edge App itself.

4.2.3 Application Context Creation

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 the Edge Service Operators to make the decision, such as streaming resolution for each user in the case of 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 one; if not, it is needed to instantiate a new space to store the context. 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 Section 4.2.3.2, and which for Edge Service Operators are described in Section 4.2.3.3.

4.2.3.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The Edge App has been instantiated.
(2) The inventory has been created and updated in the Edge Inventory Manager.

The post-condition of the procedure:

(1) The application context has been added and stored into the Edge Application.

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

(1) 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 the decision.

(2) The inventory: the network information, topology, and metadata that are about the Edge system environment and are stored in the Edge Inventory Manager component.

4.2.3.2 Procedures (from End Users)

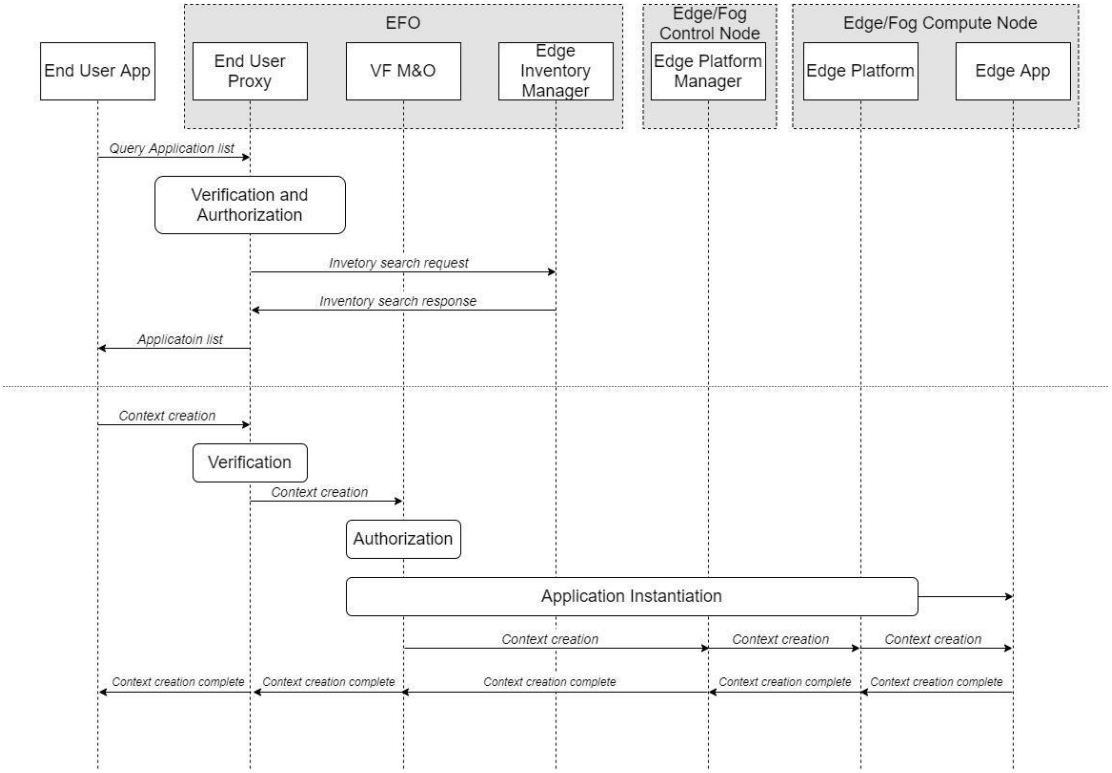

Figure 4.2.3.2-1 the workflow of application context creation procedures

The procedures of application context creation are as the following:

1. The End User App sends the application list query request (Table 4.2.3.4-1) to the End User Proxy (EUP).
2. 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.
3. 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 4.2.3.4-2 & 4.2.3.4-3).
4. The EUP sends the query results to the End User App (Table 4.2.3.4-4). The query results should include a list of all applications that satisfy the end user's requirements.
5. The End User App sends the application context creation request (Table 4.2.3.4-5) to the EUP, which contains the application context to be created.
6. 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.

7. 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 Section 4.2.2.2.
8. The EFO forwards the context creation request (Table 4.2.3.4-5) to the instance, and the instance registers the user information inside.
9. The context creation complete response is replied to the end user app (Table 4.2.3.4-6).

4.2.3.3 Procedures (from Edge Service Operators)

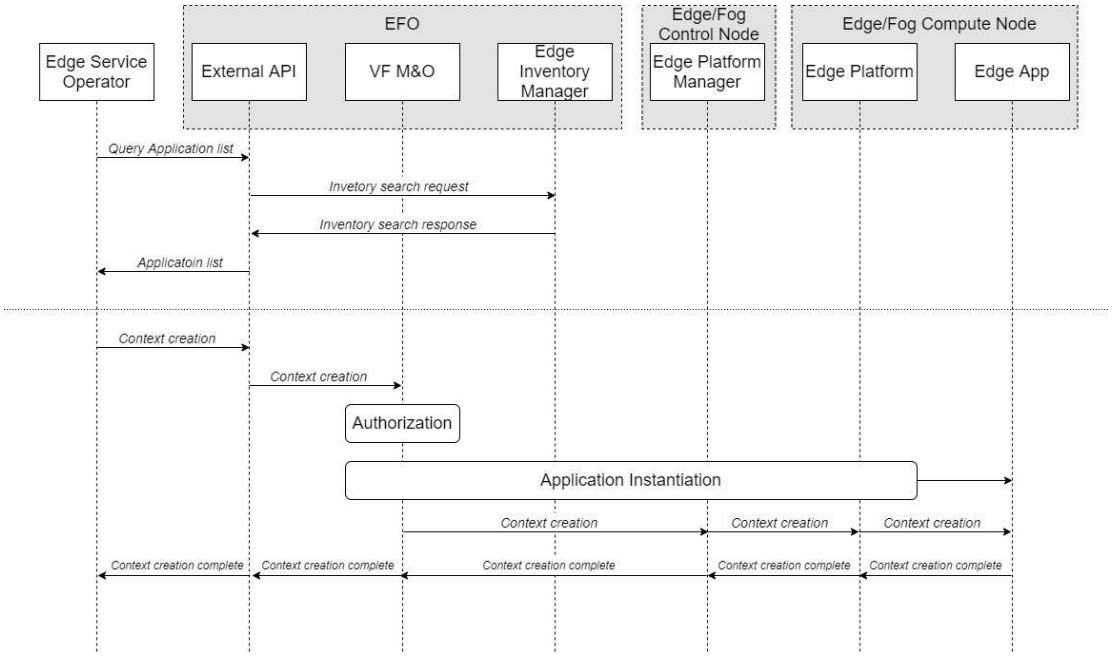

Figure 4.2.3.2-1 the workflow of application context creation procedures

The procedures of application context creation are as the following:

1. The Edge Service Operator sends the application list query request (Table 4.2.3.4-1) to the Edge Inventory Manager via the external API.
2. The request retrieves the list of Edge applications available to the end user app from the Edge Inventory Manager in the EFO (Table 4.2.3.4-2 & 4.2.3.4-3).
3. The query results are sent to the Edge Service Operator (Table 4.2.3.4-4). The query results should include a list of all applications that satisfy the end user's requirements.
4. The Edge Service Operator sends the application context creation request (Table 4.2.3.4-5) to the VF M&O via the external API, which contains the application context to be created.
5. 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 Section 4.2.2.2.
6. The EFO forwards the context creation request (Table 4.2.3.4-5) to the instance, and the instance registers the user information inside.

7. The context creation complete response is replied to the end user app (Table 4.2.3.4-6).

4.2.3.4 Requests and Responses

Table 4.2.3.4-1 Application list query request

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

Table 4.2.3.4-2 Inventory search request

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

Table 4.2.3.4-3 Inventory search response

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

Table 4.2.3.4-4 Application list query response

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

Table 4.2.3.4-5 Application context creation request

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

Table 4.2.3.4-6 Application context creation response

Element	Description

Operation status	A 200 OK code should be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code should be returned if failed.
Application ID	The identifier of the application that the End User App supposes to do context creation.

4.2.4 Application Context Deletion

The purpose of application context deletion is to delete the application context which is added or collected in the process of application context creation. The application context may be regarded as some information that helps the Edge Service Operators to make the decision, such as streaming resolution for each user in the case of video streaming application. As context creation, it is possible for both end users and Edge Service Operators to do application context deletion opration. 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 Section 4.2.4.2, and which for Edge Service Operators are described in Section 4.2.4.3.

4.2.4.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The Edge App has been instantiated.
(2) The inventory has been created and updated in the Edge Inventory Manager.
(3) The application context has been created in the target Edge Application.

The post-condition of the procedure:

(1) The application context has been removed from the system.

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

(1) Application context: Application context involves all the files that contain the custom information operators need, such as user information and statistical data.
(2) The inventory: the network information, topology, and metadata that are about the Edge system environment and are stored in the Edge Inventory Manager component.

4.2.4.2 Procedures (for End Users)

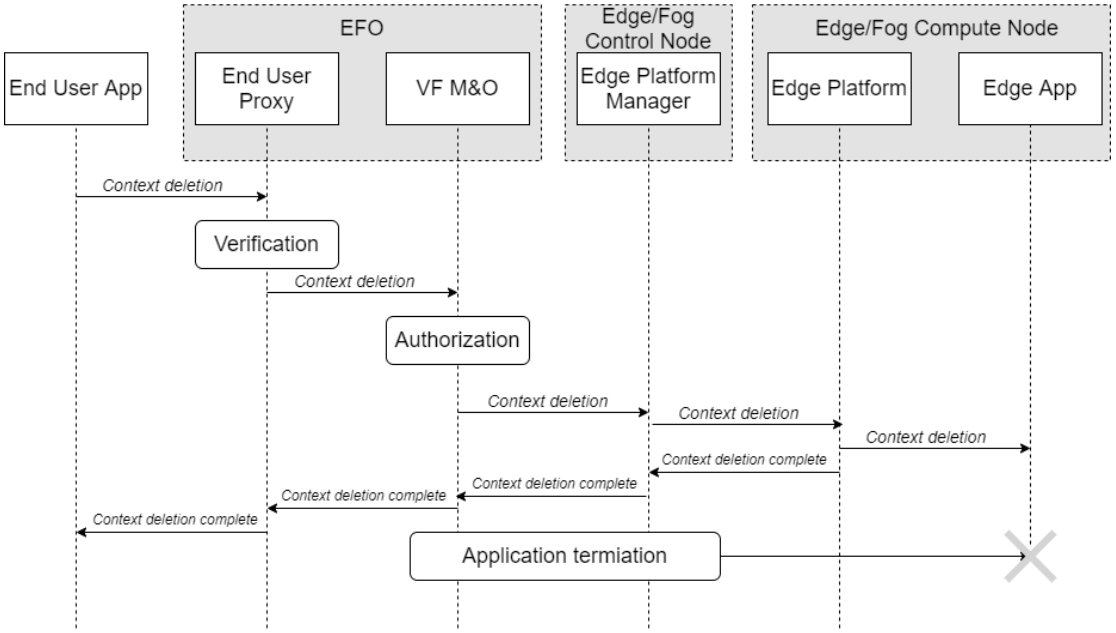

Figure 4.2.4.2-1 the workflow of application context deletion procedures

The procedures of application context deletion are as the following:

1. The end user app sends the application context deletion request (Table 4.2.4.4-1) to the End User Proxy (EUP), which contains the application context to be deleted.
2. 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.
3. The VF M&O grants the request and forwards the context deletion request to the instance, and the instance erases the related user information.
4. The context creation complete response is returned to the end user app (Table 4.2.4.4-2).
5. (Optional) If necessary, the EFO and the corresponding entities do the application termination as mentioned in Section 4.2.5.2.

4.2.4.3 Procedures (for Edge Service Operators)

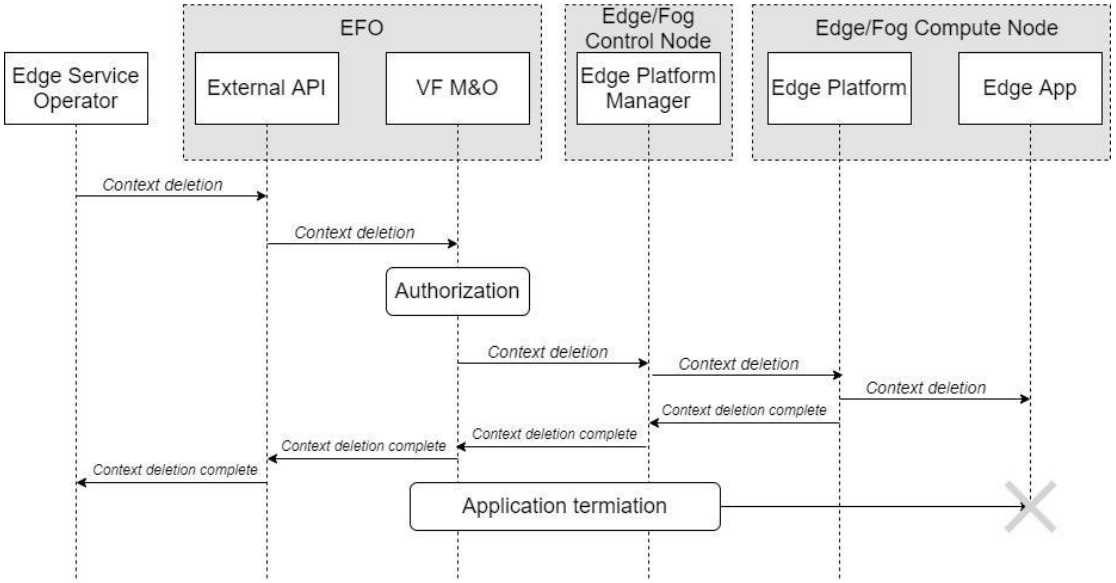

Figure 4.2.4.3-1 the workflow of application context deletion procedures

The procedures of application context deletion are as the following:

1. The end user app sends the application context deletion request (Table 4.2.4.4-1) to the VF M&O via the external API, which contains the application context to be deleted.
2. The VF M&O grants the request and forwards the context deletion request to the instance, and the instance erases the related user information.
3. The context creation complete response is returned to the Edge Service Operator (Table 4.2.4.4-2).
4. (Optional) If necessary, the EFO and the corresponding entities do the application termination as mentioned in Section 4.2.5.2.

4.2.4.4 Requests and Responses

Table 4.2.4.4-1 Application context deletion request

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

Table 4.2.4.4-2 Application context deletion response

Element	Description
Operation status	A 200 OK code should be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code should be returned if failed.
Application ID	The identifier of the application that the End User App supposes to do context deletion.

4.2.5 Application Termination

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.

4.2.5.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The Edge App has been instantiated.
(2) The inventory has been created and updated in the Edge Inventory Manager.

The post-condition of the procedure:

(1) The Edge App has been terminated.
(2) The related inventory is removed from the Edge Inventory Manager.
(3) The artifacts, which are distributed to each component in the process of on-boarding, are still saved in each component in the EFO.

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

(1) Workload: all the work that needs to be done before the Edge Application instance actually starts up.

4.2.5.2 Procedures

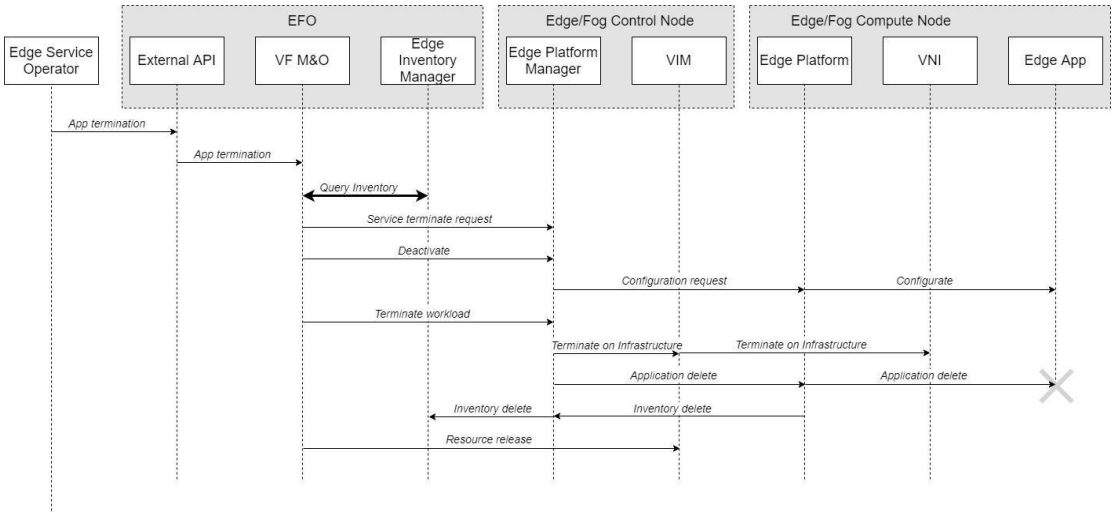

Figure 4.2.5.2-1 the workflow of application termination procedures

The detailed description of the application termination workflow is documented below:

1. The Edge Service Operator sends an application termination request (Table 4.2.5.3-1) to the VF M&O component in the EFO through the External API.
2. The VF M&O retrieves the location of the instance from Edge Inventory Manager (Table 4.2.5.3-2 & 4.2.5.3-3).
3. The VF M&O sends service terminate request (Table 4.2.5.3-1) to the target Edge Platform Manager. A response should be made if successful (Table 4.1-1).

4. The VF M&O requests the Edge Platform Manager to deactivate the instance (Table 4.2.5.3-4).
5. The Edge Platform Manager informs the corresponding Edge Platform to do the configuration work to the Edge Application (Table 4.2.5.3-5). The configuration may include the necessary settings which should be made before the Edge Application is actually shut down, such as terminate the internal VFs in order. A response should be made if successful (Table 4.1-1).
6. The VF M&O requests the Edge Platform Manager to terminate the workload (Table 4.2.5.3-4).
7. The Edge Platform Manager informs the VIM to deal with the termination work on the infrastructure (Table 4.2.5.3-6). The VIM terminates the related VNI according to the request. A response should be made if successful (Table 4.1-1). Note that the default response is also generated and sent to the VF M&O.
8. The Edge Platform Manager informs the Edge Platform to actually terminate and delete the Edge Application (Table 4.2.5.3.7). A response should be made if successful (Table 4.1-1).
9. The Edge Platform Manager returns the information to update the inventory in Edge Inventory Manager component (Table 4.2.5.3-8).
10. The VF M&O requests the VIM to release corresponding resources, including computing and networking resources (Table 4.2.5.3-9). A response should be made if successful (Table 4.1-1).

4.2.5.3 Requests and Responses

Table 4.2.5.3-1 Application termination request

Element	Description
Operator ID	The Edge Service Operator identifier. The ID should be authorized by the EFO.
Application ID	The identifier of the application that the Edge Service Operator supposes to terminate.

Table 4.2.5.3-2 Instance location retrieval request

Element	Description
Query ID	The query identifier.
Query filter	The filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations. In the case, there should be only one Edge Application that satisfies the conditions.

Table 4.2.5.3-3 Instance location retrieval response

Element	Description
Query identifier	The query identifier.
Operation status	A 200 OK code should be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code should be returned if failed.
Query result	The location information of the target Edge Application.

Table 4.2.5.3-4 Deactivation and workload termination request

Element	Description
Application ID	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 4.2.5.3-5 Edge App configuration request

Element	Description
Application ID	The identifer of the application that the Edge Service Operator supposes to terminate.
Configuration details	The configuration of the Edge App. The details of the configuration vary depending on the system and the Edge App itself.

Table 4.2.5.3-6 Infrastructure termination request

Element	Description
Application ID	The identifier of the application that the Edge Service Operator supposes to terminate. The VIM will terminate the use of infrastructure according to the request.

Table 4.2.5.3-7 Application termination and deletion request

Element	Description
Application ID	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 4.2.5.3-8 Inventory update request

Element	Description
Edge Platform Manager ID	The Edge Platform Manager identifier.
Updated inventory	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 Edge Platform may be idle now. All this information needs to be updated.

Table 4.2.5.3-9 Resource release request

Element	Description
Application ID	The identifier of the Edge Application that has been terminated.

4.2.6 Application Configuration/Re-Configuration

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.

4.2.6.1 Conditions and Involved Information

The pre-condition of the procedure:

(1) The Edge App has been instantiated.
(2) The inventory has been created and updated in the Edge Inventory Manager.

The post-condition of the procedure:

(1) The configurations of VIM, Edge Platform, and Edge App have been updated.
(2) The inventory in the Edge Inventory Manager has been updated based on the reconfiguration result.

4.2.6.2 Procedures

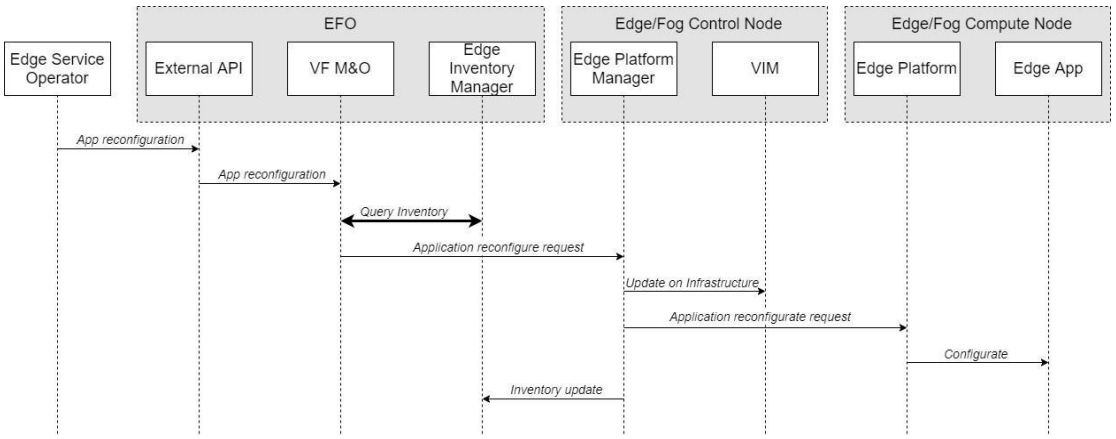

Figure 4.2.6.2-1 the workflow of application reconfiguration procedures

The detailed description of the application reconfiguration workflow is documented below:

5 The Edge Service Operator sends an application reconfiguration request (Table 4.2.6.3-1) to the VF M&O component in the Edge and Fog Orchestrator (EFO) through the External API.

5 The VF M&O retrieves the location of the instance from the Edge Inventory Manager (Table 4.2.6.3-2 & 4.2.6.3-3).

5 The VF M&O sends an application reconfiguration request to the target Edge Platform Manager (Table 4.2.6.3-4). A response should be made if successful (Table 4.1-1).

5 The Edge Platform Manager informs the VIM to update the configuration on the infrastructure (Table 4.2.6.3-4). A response should be made if successful (Table 4.1-1).

5 The Edge Platform Manager requests the corresponding Edge Platform to configure the target Edge application (Table 4.2.6.3-5). A response should be made if successful (Table 4.1-1).

5 (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 4.2.6.3-6).

4.2.6.3 Requests and Responses

Table 4.2.6.3-1 Application termination request

Element	Description
Operator ID	The Edge Service Operator identifier. The ID should be authorized by the EFO.
Application ID	The identifier of the application that the Edge Service Operator supposes to reconfigure.
New configuration	The new settings or the new configuration files of the application.

Table 4.2.6.3-2 Instance location retrieval request

Element	Description
Query ID	The query identifier.
Query filter	The filter conditions for the query. Operators can only query the manifest files that satisfy these conditions and limitations. In the case, there should be only one Edge Application that satisfies the conditions.

Table 4.2.6.3-3 Instance location retrieval response

Element	Description
Query identifier	The query identifier.
Operation status	A 200 OK code should be returned if successful. A 400 Bad Request/401 Unauthorized/403 Forbidden code should be returned if failed.
Query result	The location information of the target Edge Application.

Table 4.2.6.3-4 Application termination request

Element	Description
Application ID	The identifier of the application that the Edge Service Operator supposes to reconfigure.
New configuration	The new settings or the new configuration files of the application.

Table 4.2.6.3-5 Application termination request

Element	Description
Edge Platform Manager ID	The Edge Platform Manager identifier.
Application ID	The identifier of the application that the Edge Service Operator supposes to reconfigure.
New configuration	The new settings or the new configuration files of the application.

Table 4.2.6.3-6 Inventory update request

Element	Description
Edge Platform	The Edge Platform Manager identifier.

Manager ID	
New inventory	The inventory which is updated after the application reconfiguration.

5 Management Domains

As shown in Figure 5-1, 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 section defines two types of administrative domains related to the multiple Edge system scenario, known as the Connectivity/Interoperability Domains (CID) and the Service/Application Domains (SAD), to achieve these objectives.

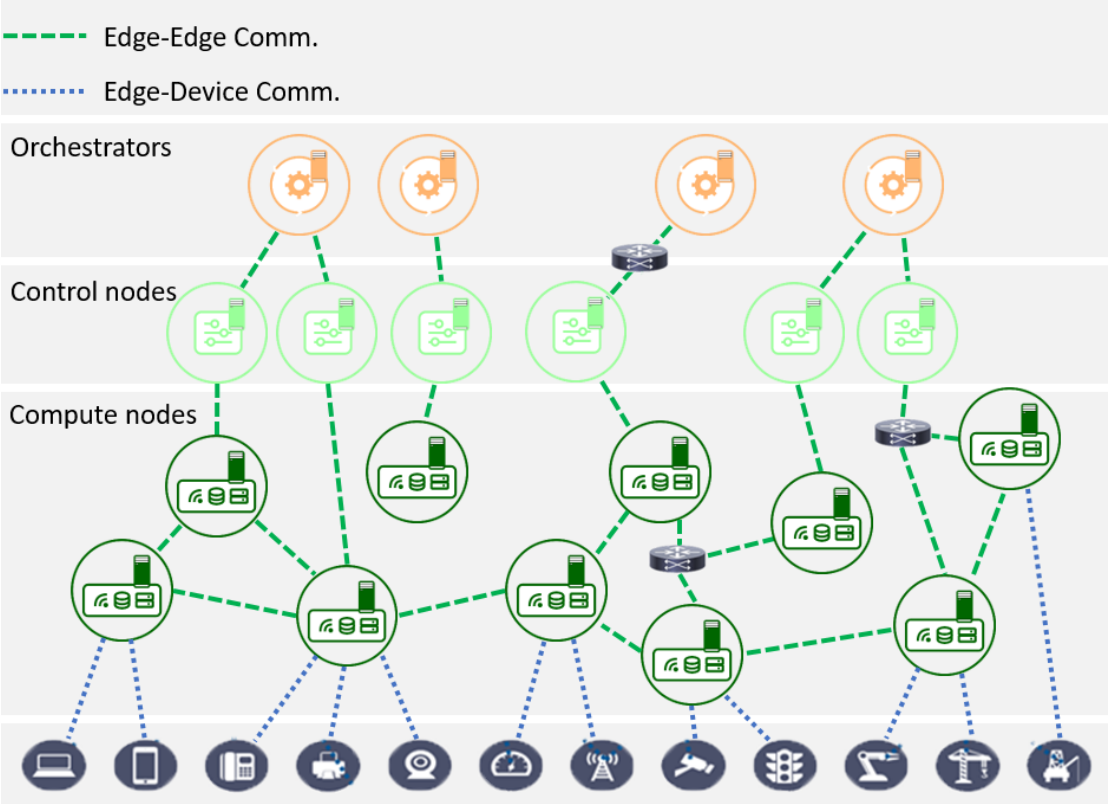

Figure 5-1 Multi-tier Edge Computing Infrastructure

5.1 Domain Components

5.1.1 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).

5.1.2 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 should be managed by an Edge control nodes 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, which 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 should expose the capability, resources, workload status, and telemetry of individual nodes and the entire cluster to the EFO.

5.1.3 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 should 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, should 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 should be the sole point of contact between the virtual Edge node and the EFO.

5.2 Connectivity and Interoperability Domains (CID)

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[2] and interoperate[3] with one another. That is, two EFOs in two different CIDs may still be able to interconnect or interoperate with each other but

[2] *Interconnectivity* is the capability of networked entities to exchange information with one another.

[3] 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.

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.

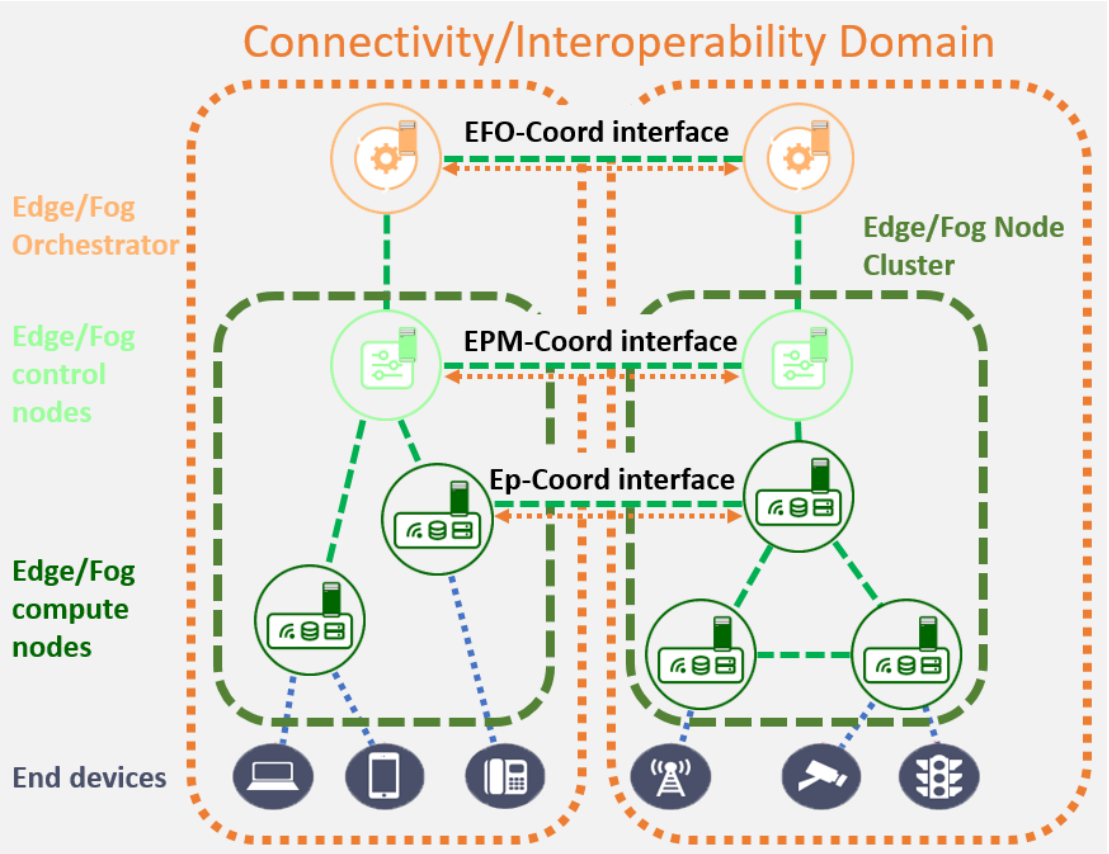

Figure 5.2-1 Structure of Connectivity/Interoperability Domains (CID)

5.2.1 Organization

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 subsections explain the concept of these terms and components.

5.2.1.1 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.

5.2.1.2 Border Edge Nodes

Two or more CIDs may interconnect and interoperate with one another directly through a bridging Edge node, known as a Border Edge Node (or just Border Edge Node, BEN), or indirectly through their shared Edge service provider(s). A Border Edge Nodes is a physical or logical Edge node that serves as an intermediary between multiple CIDs. A BEN must be a member of each CID or Edge node cluster that it serves to bridge.

A recommended approach to implement a BEN is to establish a private network among multiple physical or virtual Edge nodes, each of which is a member of a CID that the BEN serves to bridge. Since these Edge nodes form a cluster, they can implement their own protocols and enforce their own policies that specify the extent of interconnectivity and interoperability supported by the BEN.

5.2.1.3 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, and two Edge control nodes can share the run-time information of the underlying Edge Platforms via the “EPM-Coord” interface.

5.2.2 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 of them. Each interface should implement the set of cryptographic algorithms employed by the CID and enforce the same security policies.

5.3 Service and Application Domains (SAD)

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 in order 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 CID. The Edge Applications and the involved resources are considered as a SAD.

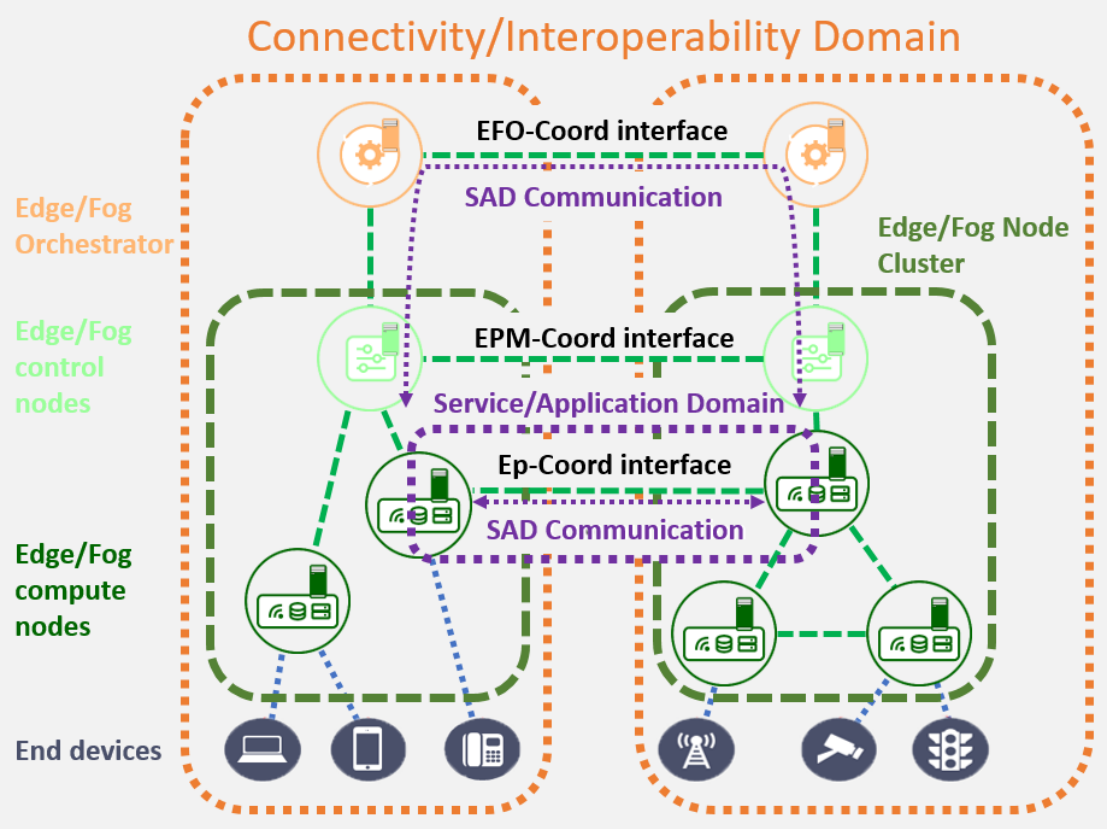

Figure 5.3-1 Structure of Service/Application Domain (SAD)

5.3.1 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, which 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 progress, two Edge Application instance 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.

5.3.2 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.