ETSI MEC:
Edge Computing Standard

Agenda

- Introduction
  - Requirements: Edge Native Platform/Applications
  - Why Standardization
  - What is ETSI MEC
  - ETSI MEC Focus Areas & Roadmap
- MEC Architecture
- Reference Points
- Model (Application & Edge)
- API Details
- Sample Application Instantiation Flow
- Client to Edge Application Connection

Edge Native Applications+Platform Requirement: Network Capability

- **Network Capabilities: Edge Applications to consider Network condition/ Topology**

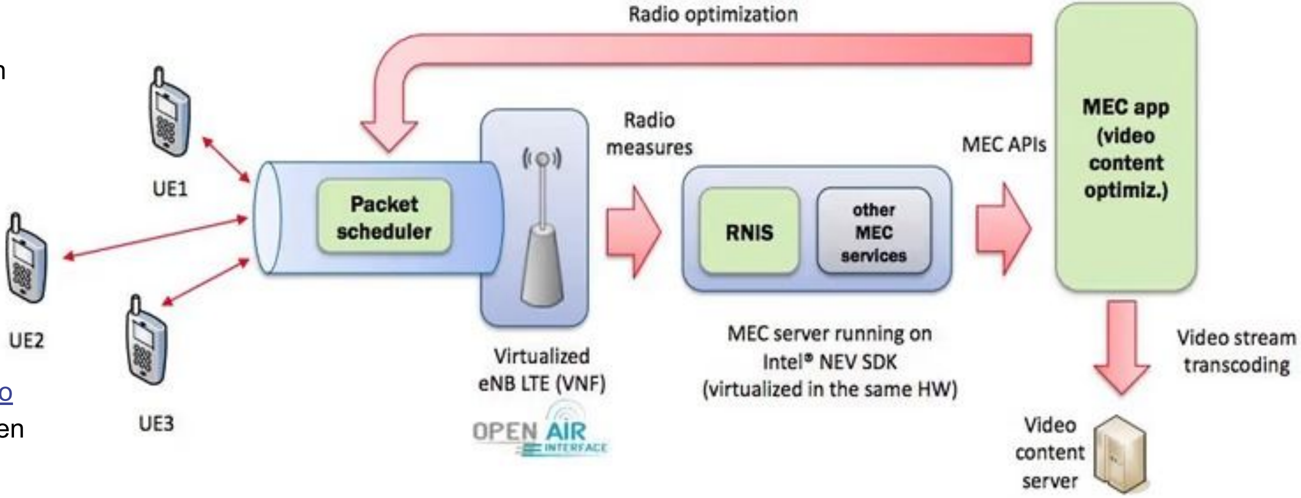
Traditionally (Cloud based system/ Internet connectivity): Applications and Networks should be completely agnostic to each other.
Note: Till date, few traditional applications encountered issues with this approach since the performance (latency/throughput) requirements of applications have traditionally been much looser than the performance that networks can deliver.

With 5G/Industrial 4.0 (Edge Apps): For **expected application performance** (latency/throughput/reliability), both aspects (the application design and the network) are important and remaining completely agnostic becomes difficult
Example: Adjusting the application behavior for example: video stream compression ratio in response to throughput throttling (network condition) and thereby making application performance more predictable.

Problem: Mobile video streaming/conferencing suffer from sluggish video buffering times.

Reason: Video buffering is caused by the Transmission Control Protocol (TCP) not adapting fast enough to varying radio conditions.

Solution: Edge technology dodges those video streaming issues by communicating to the video server the best bit rate for the given radio conditions. This reduces the buffering time of the device's video stream.



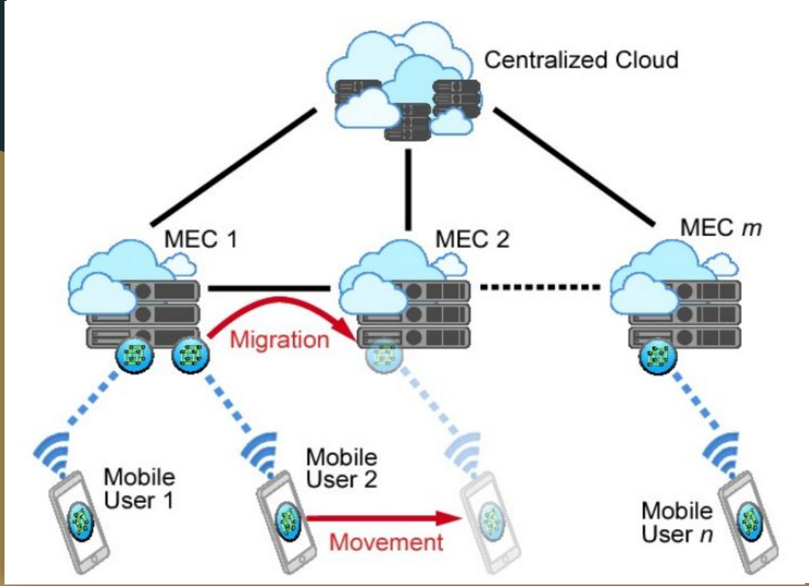
Edge Native Applications+Platform Requirement: Mobility

- **Application Mobility: Edge Applications to consider Application mobility due to User mobility**

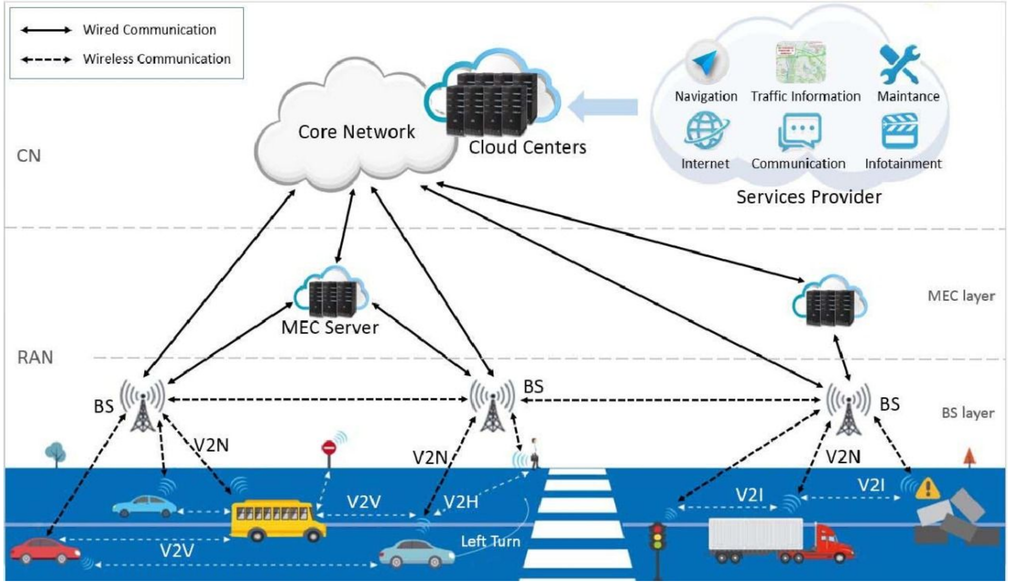
Traditionally (Cloud based system/ Internet connectivity): Applications are on central cloud, no need for mobility

With 5G/V2X (Edge Apps): Application mobility is required to ensure required application performance (latency/throughput/ reliability) without affecting service continuity.
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/021/02.01.01_60/gs_MEC021v020101p.pdf

Mobile Application Use Case



V2X Use Case



Edge Native Applications+Platform Requirement: Interface for Device

Application Deployment Triggering Points

1. User initiated via CFS (customer facing service) Portal (Traditionally)
2. Device application driven: When first client application within that edge comes UP (Extra requirement for Edge)

Additionally Client Application would need to know about change in application IP address (in case for mobility, relocation) - Extra requirement for Edge

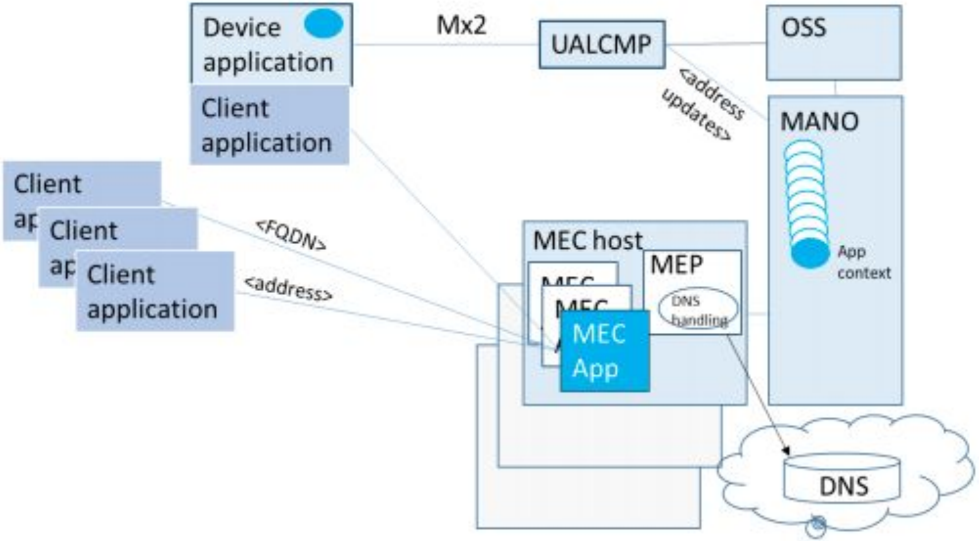


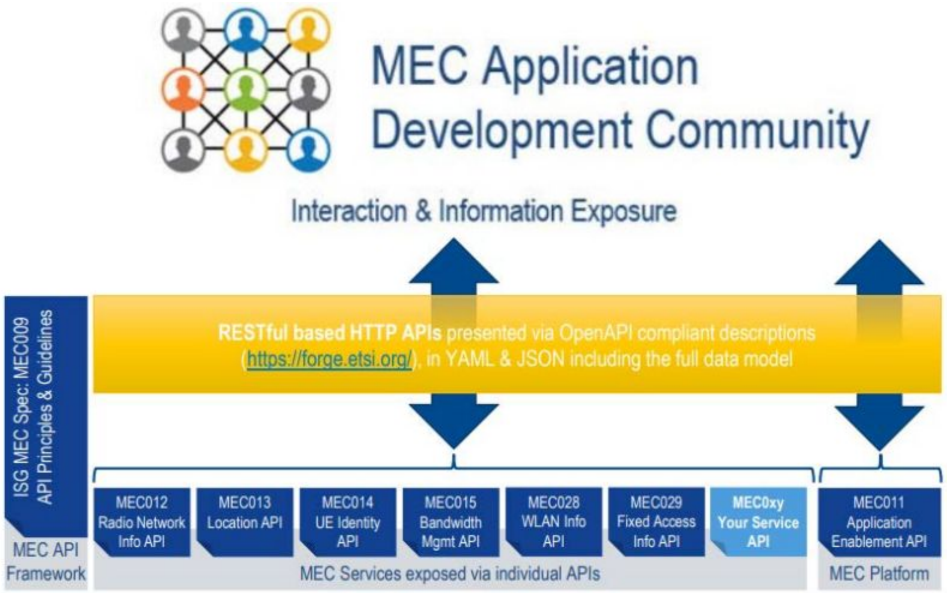
Figure 6: Communications between terminals and the MEC system

LAMBDA as an Alternative ??

Edge Native Platform Requirement: Application/Service enablement

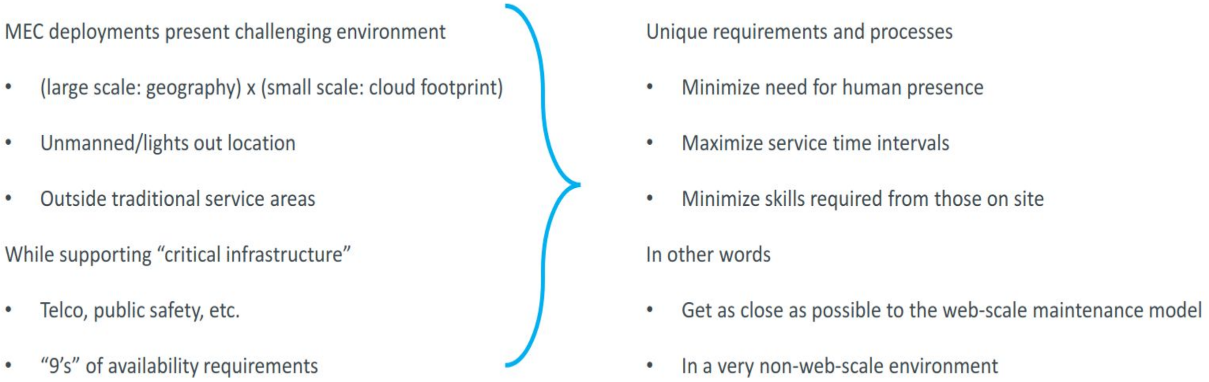
That’s all you need as a MEC App developer

- Discover network, users, capabilities and local services (Location, bandwidth, RNI)
- Manage traffic, DNS, mobility, V2X, etc.
- Register your own service and discover third party services available locally
- Dependent Service Up/Down Notifications


MEC Application Development Community
Interaction & Information Exposure
ISG MEC Spec: MEC009 API Principles & Guidelines
RESTful based HTTP APIs presented via OpenAPI compliant descriptions (https://forge.etsi.org/), in YAML & JSON including the full data model
MEC012 Radio Network Info API
MEC013 Location API
MEC014 UE Identity API
MEC015 Bandwidth Mgmt API
MEC028 WLAN Info API
MEC029 Fixed Access Info API
MEC0xy Your Service API
MEC011 Application Enablement API
MEC API Framework
MEC Services exposed via individual APIs
MEC Platform

Note: These requirements are common with any Cloud Platforms too.

Edge Native Platform Requirement: Infra/App Management



MEC deployments present challenging environment

- (large scale: geography) x (small scale: cloud footprint)
- Unmanned/lights out location
- Outside traditional service areas

While supporting “critical infrastructure”

- Telco, public safety, etc.
- “9’s” of availability requirements

Unique requirements and processes

- Minimize need for human presence
- Maximize service time intervals
- Minimize skills required from those on site

In other words

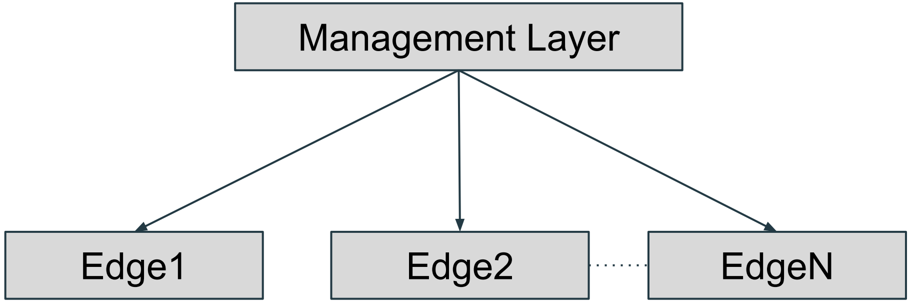
- Get as close as possible to the web-scale maintenance model
- In a very non-web-scale environment

Centralized Management
With Edge Selection logic

Autonomous Edges
Edges with not so stable connectivity with center.

Management Tasks

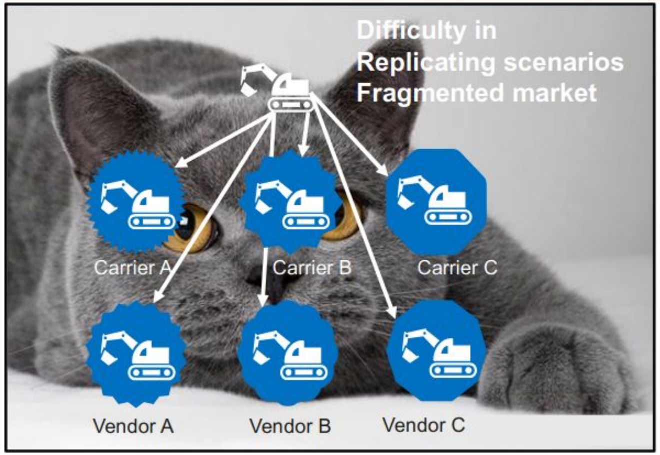
- Application Packaging
- Package Onboarding
- Application Instantiation to desired edge.
- Traffic Rule & DNS Rule Configuration
- Application LifeCycle Management
  - Ensure KPIs
  - Relocation if needed



Note: These requirements are common with any Cloud Platforms too. Differentiating factor could be scale (no of edges), edge selection logic, autonomous edges etc.

Why Standardization to achieve these Requirements

Small-scale Application Ecosystems of Vendors and Carriers



Difficulty in
Replicating scenarios
Fragmented market

Carrier A
Carrier B
Carrier C

Vendor A
Vendor B
Vendor C

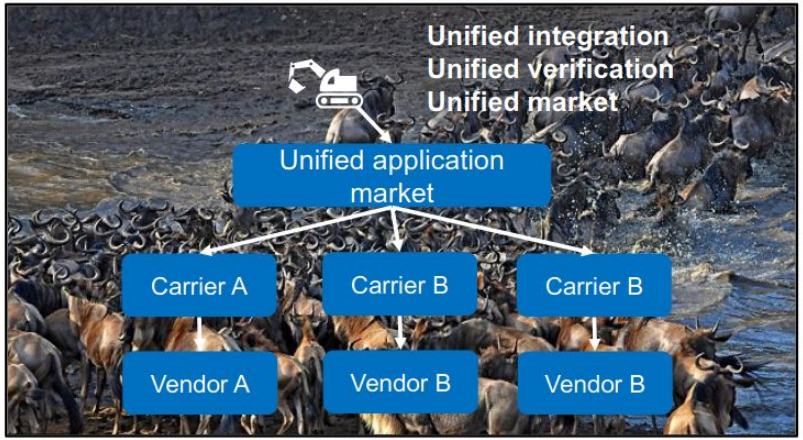
Manually integrated
growth mode

Focus on Few
application types

Hundreds
of applications

- There is no **unified platform** for innovative achievements in the industry. Everything is reorganized from scratch, every year.
- No **closed loop of business** is formed, and projects are almost impossible to replicate.
- There is no drive for innovation, and the **technology growth is slow.**

Large-scale Ecosystems of Edge Computing Industry



Unified integration
Unified verification
Unified market

Unified application market

Carrier A
Carrier B
Carrier B

Vendor A
Vendor B
Vendor B

Self-reproduction
growth mode

Diversified
application types

Millions
of applications

- **New Ecosystem:** Building Industry Application Ecosystems with Unified Market and Shared Applications
- **The open and unified market**, unified integration, and unified verification enable convenient circulation of an application across vendors and carriers, reduce application replication costs, and build a large-scale, self-growth, and diversified new ecosystem

What: ETSI MEC

The Multi-access Edge Computing (MEC) initiative is an Industry Specification Group **(ISG) within ETSI** since **2014**.
Aim to define **Edge Computing standard Specification** with focus on **Telco Edge**
Initiative Include 110 members from mobile operators, application developers, Over the Top (OTT) players, Independent Software Vendors (ISVs), telecom equipment vendors, IT platform vendors, system integrators, and technology providers

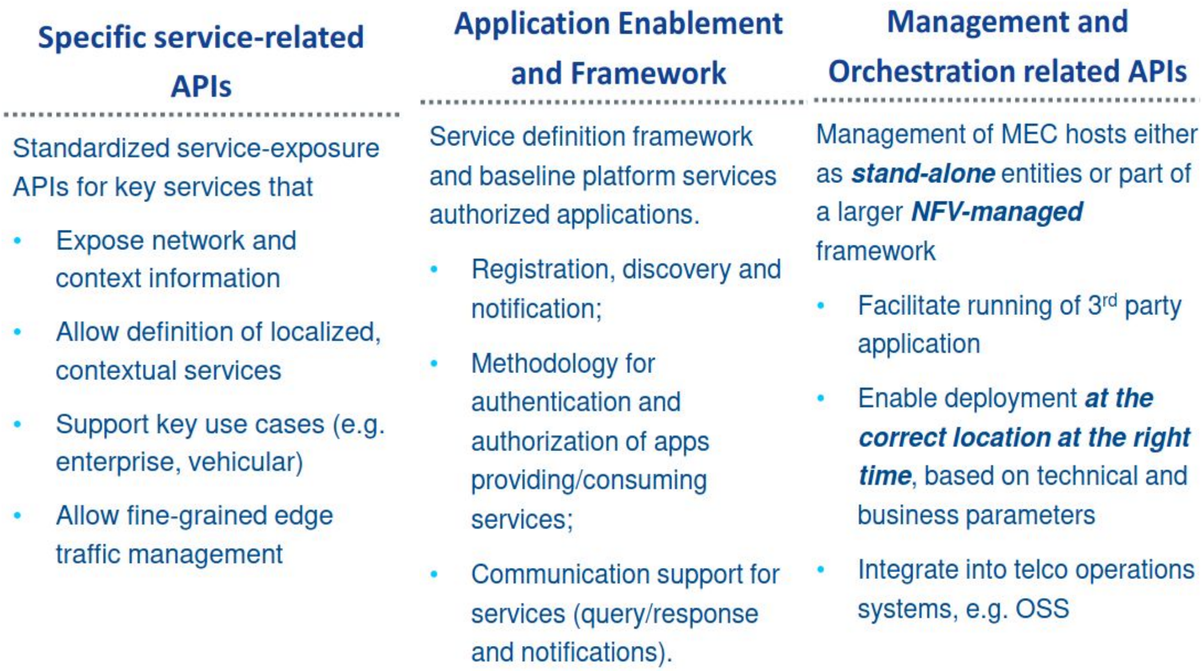
110 members - Operators – Technology Vendors – IT players – Application developers

Hewlett Packard Enterprise
vodafone
NOKIA
HUAWEI
intel
SAMSUNG
ZTE
NEC
INTERDIGITAL
NTT
中国移动 China Mobile
MAVENIR
Qualcomm
vmware
Saguna
docomo
RADCOM
SK telecom
TNO
Telefonica
UBiqube
BOSCH Invented for life
ETRI
ITALTEL
redhat
AT&T
verizon
ROGERS
LG Life's Good
MITSUBISHI ELECTRIC
VO GO
China unicom中国联通
TIM
Rakuten
中国电信 CHINA TELECOM
KDDI
OPENET
amdocs
DOLBY
Türk Telekom
IBM
radware
afnor
cnrs
ubiwhere
orange
ATHONET
PT
Quortus
ADVA
netrounds
iPgallery
SONY
flex
NEXTRA PARTNERS
KEYSIGHT TECHNOLOGIES
UNIVERSITY OF SURREY
ITRI
ADLINK
FUTUREWEI
Silicom
CERAGON
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
東京大学 THE UNIVERSITY OF TOKYO
INTRACOM TELECOM
ECI THE ELASTIC NETWORK
CTTC
POLITECNICO DI TORINO
DELL
CAICT
EURECOM
Altiostar
Acelleran
SCILD INNOVATIONS
nife
NETAS
kpn
NEXTWORKS
cnit
JUNIPER
TELECOM FORESIGHT CONSULTING
Tech Mahindra
ACS
ASTRI
VIAVI
affirmed
CPLANE NETWORKS
Qwilt
ARTESYN
uc3m
AsiaInfo 亚信科技
STC
SES
LINKS
telenity
Akamai
GSMA

ETSI MEC: Focus Areas

1. Requirement/ Usecases
2. Architecture
3. Model
   a. Application
   b. Infra
4. API & Reference Points
   a. Network Capabilities
   b. Application enablement
   c. Management
5. Mobility
6. Analysis on multiple industry segment & communication technology

APIs in Focus



Specific service-related APIs

Standardized service-exposure APIs for key services that

- Expose network and context information
- Allow definition of localized, contextual services
- Support key use cases (e.g. enterprise, vehicular)
- Allow fine-grained edge traffic management

Application Enablement and Framework

Service definition framework and baseline platform services authorized applications.

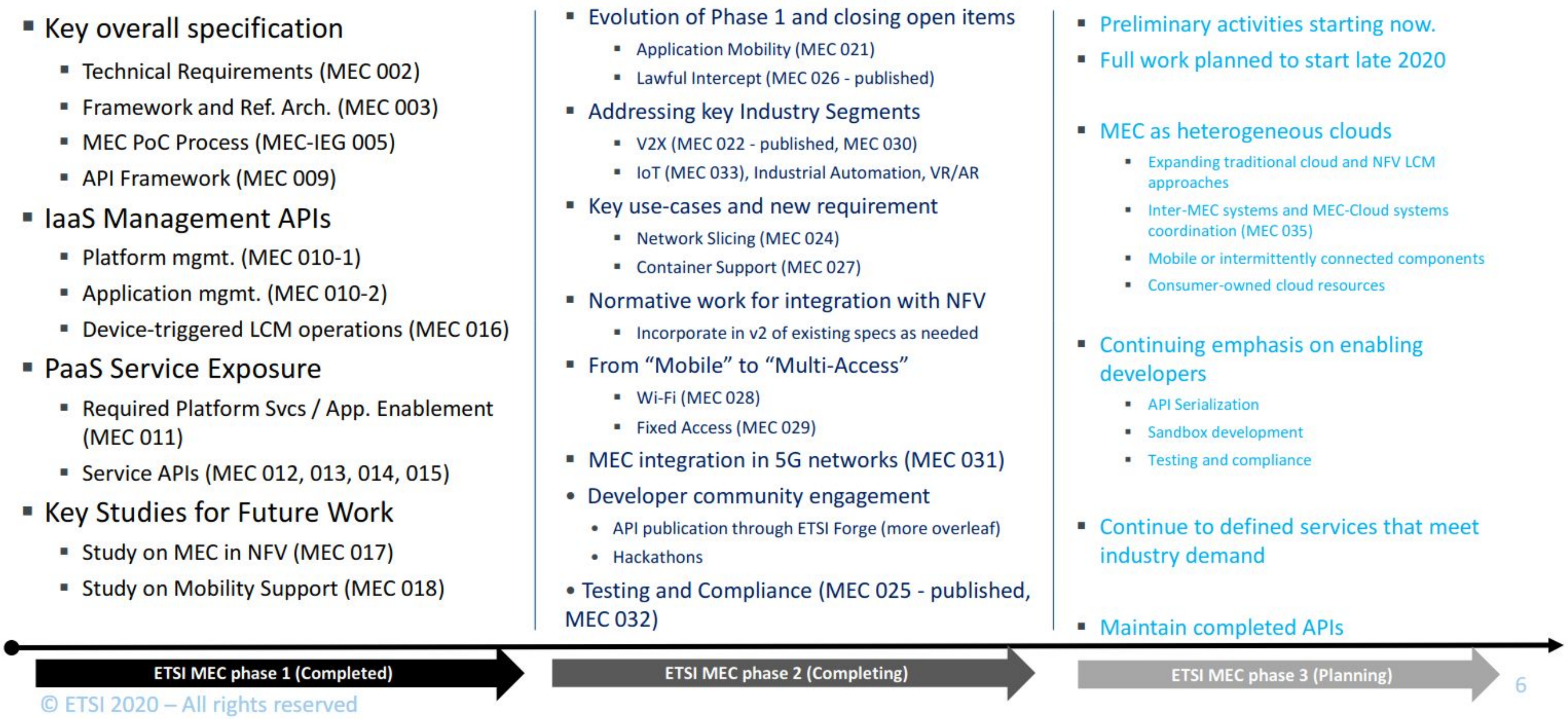
- Registration, discovery and notification;
- Methodology for authentication and authorization of apps providing/consuming services;
- Communication support for services (query/response and notifications).

Management and Orchestration related APIs

Management of MEC hosts either as ***stand-alone*** entities or part of a larger ***NFV-managed*** framework

- Facilitate running of 3rd party application
- Enable deployment ***at the correct location at the right time***, based on technical and business parameters
- Integrate into telco operations systems, e.g. OSS

ETSI MEC: Where it Stand



- Key overall specification
  - Technical Requirements (MEC 002)
  - Framework and Ref. Arch. (MEC 003)
  - MEC PoC Process (MEC-IEG 005)
  - API Framework (MEC 009)
- IaaS Management APIs
  - Platform mgmt. (MEC 010-1)
  - Application mgmt. (MEC 010-2)
  - Device-triggered LCM operations (MEC 016)
- PaaS Service Exposure
  - Required Platform Svcs / App. Enablement (MEC 011)
  - Service APIs (MEC 012, 013, 014, 015)
- Key Studies for Future Work
  - Study on MEC in NFV (MEC 017)
  - Study on Mobility Support (MEC 018)

ETSI MEC phase 1 (Completed)

- Evolution of Phase 1 and closing open items
  - Application Mobility (MEC 021)
  - Lawful Intercept (MEC 026 - published)
- Addressing key Industry Segments
  - V2X (MEC 022 - published, MEC 030)
  - IoT (MEC 033), Industrial Automation, VR/AR
- Key use-cases and new requirement
  - Network Slicing (MEC 024)
  - Container Support (MEC 027)
- Normative work for integration with NFV
  - Incorporate in v2 of existing specs as needed
- From “Mobile” to “Multi-Access”
  - Wi-Fi (MEC 028)
  - Fixed Access (MEC 029)
- MEC integration in 5G networks (MEC 031)
- Developer community engagement
  - API publication through ETSI Forge (more overleaf)
  - Hackathons
- Testing and Compliance (MEC 025 - published, MEC 032)

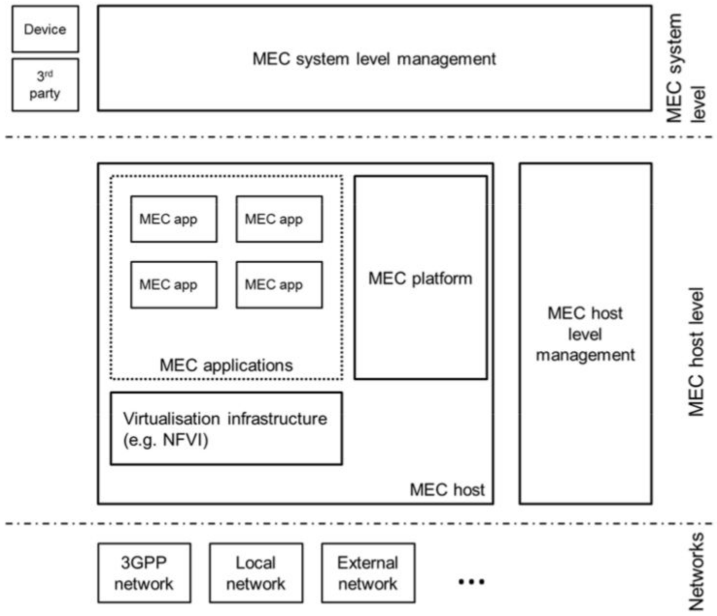
ETSI MEC phase 2 (Completing)

- Preliminary activities starting now.
- Full work planned to start late 2020
- MEC as heterogeneous clouds
  - Expanding traditional cloud and NFV LCM approaches
  - Inter-MEC systems and MEC-Cloud systems coordination (MEC 035)
  - Mobile or intermittently connected components
  - Consumer-owned cloud resources
- Continuing emphasis on enabling developers
  - API Serialization
  - Sandbox development
  - Testing and compliance
- Continue to defined services that meet industry demand
- Maintain completed APIs

ETSI MEC phase 3 (Planning)

© ETSI 2020 – All rights reserved

ETSI MEC: Architecture



The **MEC management** comprises the **MEC system level management** and the **MEC host level management.**

The **MEC system level management** includes the **Multi-access edge orchestrator** as its core component, which has an overview of the complete MEC system.

The **MEC host level management** comprises the **MEC platform manager** and the **virtualization infrastructure manager,** and handles the management of the MEC specific functionality of a particular MEC host and the applications running on it.

The **MEC host** is an entity that contains a MEC platform and a virtualization infrastructure which provides compute, storage, and network resources, for the purpose of running MEC applications.

The **MEC platform** is the collection of essential functionality required to run MEC applications on a particular virtualization infrastructure and enable them to provide and consume MEC services. The MEC platform can also provide services.

MEC applications are instantiated on the virtualization infrastructure of the MEC host based on configuration or requests validated by the MEC management.

Architecture

OSS

- Receives requests via the CFS portal and from device applications for instantiation or termination of applications
- Granted requests are forwarded to the multi-access edge orchestrator for further processing.
- Receives requests from device applications for relocating applications between external clouds and the MEC system.

User APP LCM Proxy

- allows device applications to request on-boarding, instantiation, termination of user applications
- relocation of user applications in and out of the MEC system.
- Informs the device applications about the state of the user applications.
- Authorizes requests from device applications in the device and interacts with the OSS and the multi-access edge orchestrator for further processing of these requests.

VIM

- Allocating, managing and releasing virtualized resources
- preparing the virtualization infrastructure to run a software image.
- collecting and reporting performance and fault information about the virtualized resources;
- when supported, performing application relocation.

MEC Orchestrator

- Maintenance view: Deployed host, available resources/services and topology
- App package management, App repository
- Host selection for app deployment
- Initiate App instantiation or instance termination
- App location migration

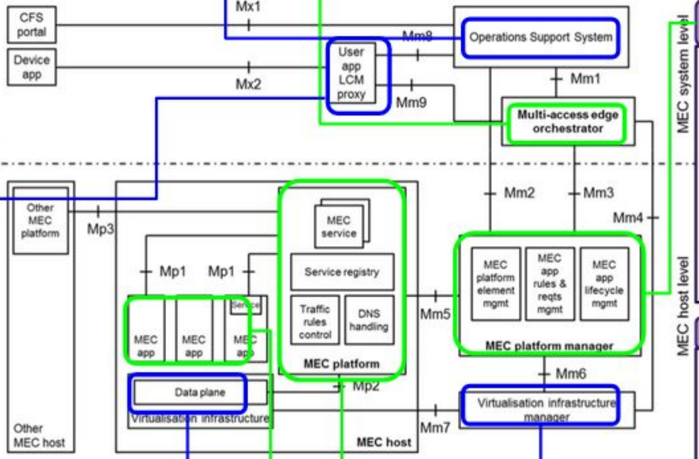


Figure 6-1: Multi-access edge system reference architecture

Data Plane

- Provides Intermediate UPF
- App flow scheduling and load balancing
- Traffic redirection

MEC App

- Runs on containers/VM of MEC hosts
- Interworks with the MEPM via Mp1 interface for lifecycle, flow rule management and service governance
- Provides services for other APPs
- Access MEC services

MEC Platform Manager

- App instance lifecycle management, including status report to the MEO
- MEP NE management, including NE and service deployment
- Flow and DNS rule management, including rule setting receipe and distribution to each MEP
- Virtual resource status, fault report and performance statistics collection .

MEC Platform

- Provides services governance for apps(Release,discovery, subscription and consumption)
- Receives flow rule settings and sends them to the data plane
- Receive DNS policies and uses them in DNS services (service/proxy)
- Monitors app integration and O&M status
- Offer basic app services, including API GW capability exposure.

Reference Points

Mp1 Reference Point

- ❑ Service Registration and discovery
- ❑ Service availability query, subscription and notification
- ❑ Flow Rule Query, activation, deactivation and update
- ❑ DNS rule activation and deactivation
- ❑ Platform capability retrieval and timestamp.

Mm1 Reference Point

- ❑ App Package mgmt
  - ❑ On boarding
  - ❑ Search, delete, enable, disable
- ❑ App Life cycle mgmt
  - ❑ App Instantiation
  - ❑ Termination
  - ❑ Status change

Mm2 Reference Point

- ❑ Mgmt data model definition
- ❑ Data model, data retrieval from MEPM
- ❑ Data model change notice to the OSS
- ❑ Traffic distribution rule addition, activation or deactivation
- ❑ DNS rule addition, activation, or deactivation
- ❑ Fault management.

Mm3 Reference Point

- ❑ App Package mgmt
  - ❑ App package data requesting
  - ❑ App package change notice
  - ❑ App pkg on-boarding notice
  - ❑ App pkg retrieval
- ❑ App Life cycle mgmt (MEO->MEPM)
  - ❑ App instantiation and termination
  - ❑ App instance status change
  - ❑ Status request (MEO->MEPM)
  - ❑ App lifecycle change notice

Mx1 Reference Point

- ❑ Used by the third-parties to request the MEC system to run applications in the MEC system.

Mx2 Reference Point

- ❑ Used by a device application to request the MEC system to run an application in the MEC system
- ❑ Used by a device application to request the MEC system move an application in or out of the MEC system

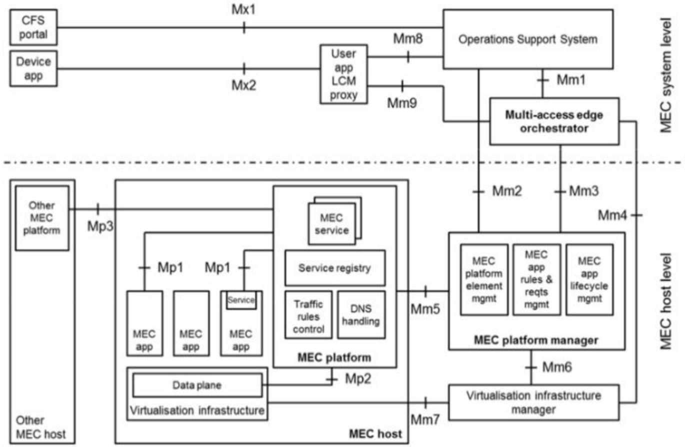


Figure 6-1: Multi-access edge system reference architecture

There are three groups of reference points defined between the system entities:

- reference points regarding the MEC platform functionality (Mp);
- management reference points (Mm); and
- reference points connecting to external entities (Mx).

Mp2 Reference Point

- ❑ instruct the data plane on how to route traffic among applications, networks, services, etc.

Mp3 Reference Point

- ❑ control communication between MEC platforms.

Mm9 Reference Point

- ❑ Manage MEC applications requested by device application.

Mm4 Reference Point

- ❑ manage virtualized resources of the MEC host, including keeping track of available resource capacity
- ❑ manage application images.

Mm5 Reference Point

- ❑ Perform platform configuration
- ❑ Configuration of the application rules and requirements
- ❑ Performs application lifecycle support procedures

Mm6 Reference Point

- ❑ Manage virtualized resource to realize the application lifecycle management.

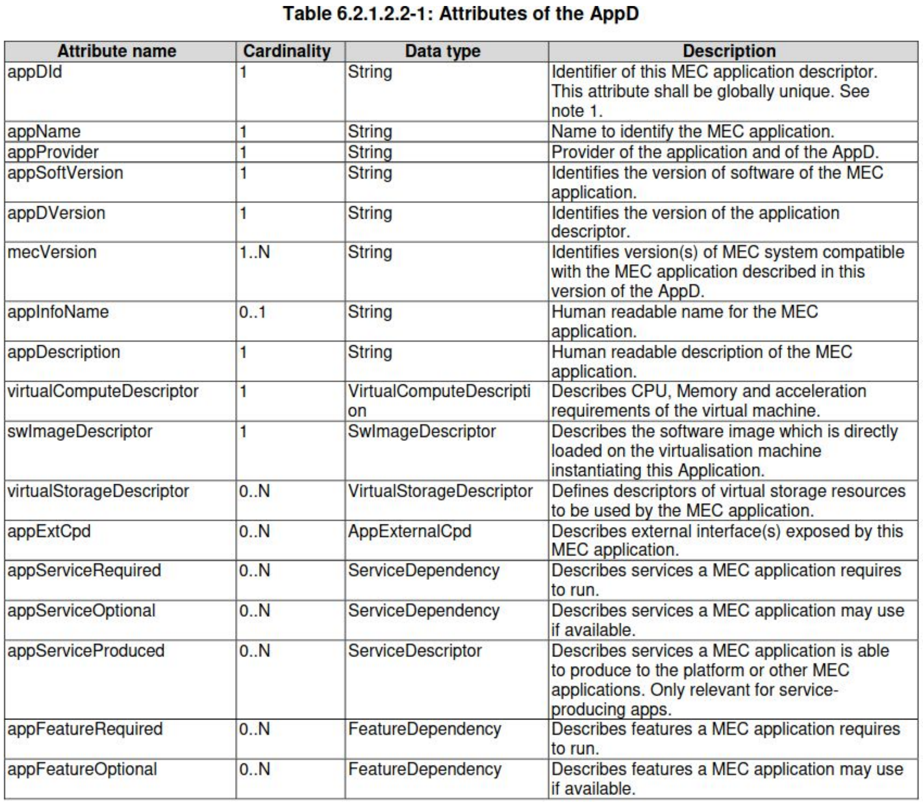
Mm7 Reference Point

- ❑ Manage the virtualization infrastructure inside MEC host.

Mm8 Reference Point

- ❑ Handle device applications requests for running applications in the MEC system.

Model - AppD (Part of Package)



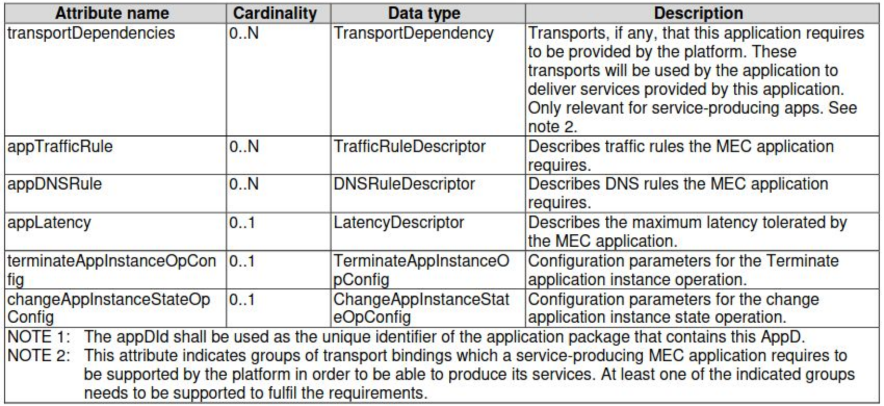


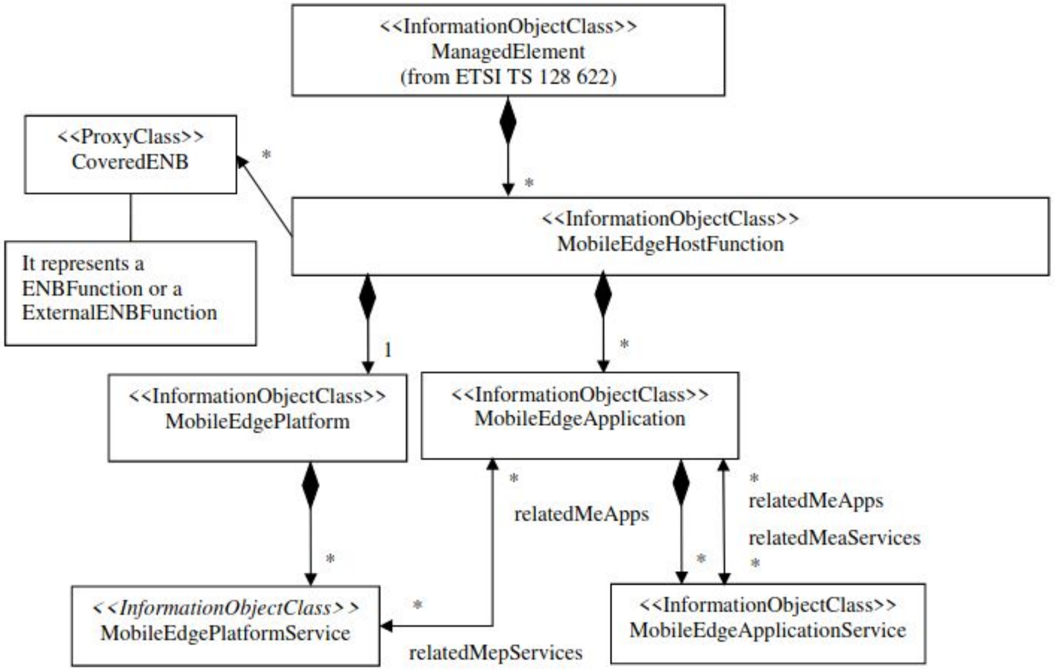
Table 6.2.1.2.2-1: Attributes of the AppD

Attribute name	Cardinality	Data type	Description
appDId	1	String	Identifier of this MEC application descriptor. This attribute shall be globally unique. See note 1.
appName	1	String	Name to identify the MEC application.
appProvider	1	String	Provider of the application and of the AppD.
appSoftVersion	1	String	Identifies the version of software of the MEC application.
appDVersion	1	String	Identifies the version of the application descriptor.
mecVersion	1..N	String	Identifies version(s) of MEC system compatible with the MEC application described in this version of the AppD.
appInfoName	0..1	String	Human readable name for the MEC application.
appDescription	1	String	Human readable description of the MEC application.
virtualComputeDescriptor	1	VirtualComputeDescription	Describes CPU, Memory and acceleration requirements of the virtual machine.
swImageDescriptor	1	SwImageDescriptor	Describes the software image which is directly loaded on the virtualisation machine instantiating this Application.
virtualStorageDescriptor	0..N	VirtualStorageDescriptor	Defines descriptors of virtual storage resources to be used by the MEC application.
appExtCpd	0..N	AppExternalCpd	Describes external interface(s) exposed by this MEC application.
appServiceRequired	0..N	ServiceDependency	Describes services a MEC application requires to run.
appServiceOptional	0..N	ServiceDependency	Describes services a MEC application may use if available.
appServiceProduced	0..N	ServiceDescriptor	Describes services a MEC application is able to produce to the platform or other MEC applications. Only relevant for service-producing apps.
appFeatureRequired	0..N	FeatureDependency	Describes features a MEC application requires to run.
appFeatureOptional	0..N	FeatureDependency	Describes features a MEC application may use if available.
transportDependencies	0..N	TransportDependency	Transports, if any, that this application requires to be provided by the platform. These transports will be used by the application to deliver services provided by this application. Only relevant for service-producing apps. See note 2.
appTrafficRule	0..N	TrafficRuleDescriptor	Describes traffic rules the MEC application requires.
appDNSRule	0..N	DNSRuleDescriptor	Describes DNS rules the MEC application requires.
appLatency	0..1	LatencyDescriptor	Describes the maximum latency tolerated by the MEC application.
terminateAppInstanceOpConfig	0..1	TerminateAppInstanceOpConfig	Configuration parameters for the Terminate application instance operation.
changeAppInstanceStateOpConfig	0..1	ChangeAppInstanceStateOpConfig	Configuration parameters for the change application instance state operation.

NOTE 1: The appDId shall be used as the unique identifier of the application package that contains this AppD.

NOTE 2: This attribute indicates groups of transport bindings which a service-producing MEC application requires to be supported by the platform in order to be able to produce its services. At least one of the indicated groups needs to be supported to fulfil the requirements.

Model - MEC Hosts


<<InformationObjectClass>>
ManagedElement
(from ETSI TS 128 622)
<<ProxyClass>>
CoveredENB
It represents a ENBFunction or a ExternalENBFunction
*
*
<<InformationObjectClass>>
MobileEdgeHostFunction
1
*
<<InformationObjectClass>>
MobileEdgePlatform
<<InformationObjectClass>>
MobileEdgeApplication
*
relatedMeApps
*
relatedMeApps
relatedMeaServices
*
*
*
<<InformationObjectClass>>
MobileEdgePlatformService
*
relatedMepServices
<<InformationObjectClass>>
MobileEdgeApplicationService

NOTE 1: A service producing `MobileEdgeApplication` instance contains at least one `MobileEdgeApplicationService` instance. A non-service producing `MobileEdgeApplication` instance does not contain any `MobileEdgeApplicationService` instances.

NOTE 2: A service producing `MobileEdgeApplication` instance containing a `MobileEdgeApplicationService` instance cannot have a relationship with that `MobileEdgeApplicationService` instance.

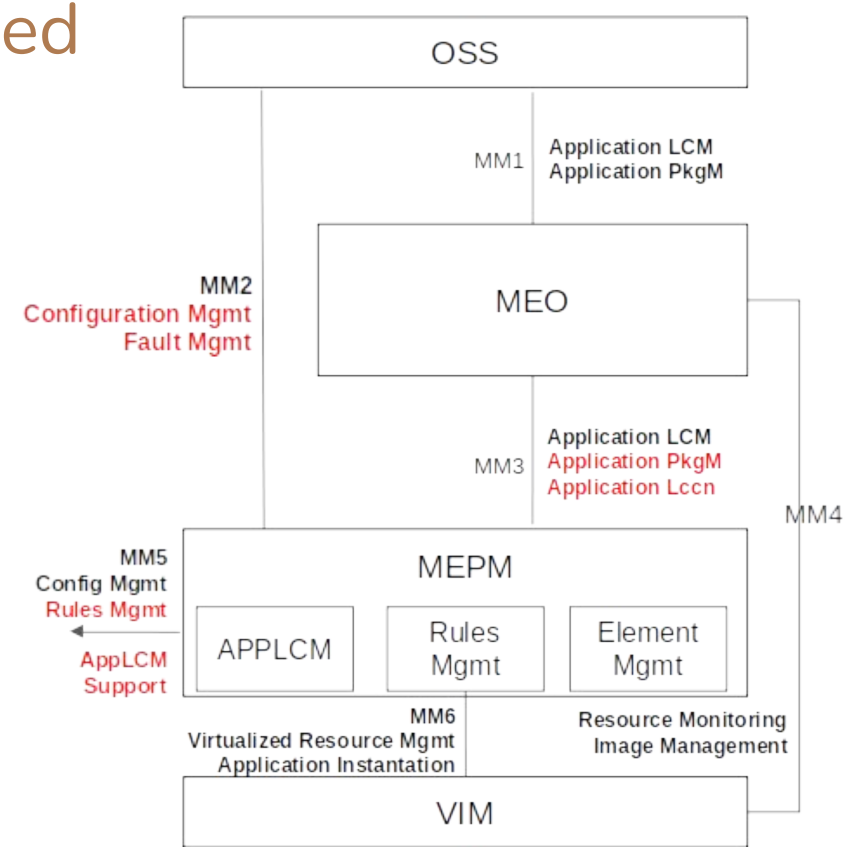
Figure 7.2.1.2.1-1: Information Model of the Mobile Edge Host - Part 1

Management APIs Simplified

MM1, MM2 & MM3 - Defined by ETSI MEC

MM4, MM6 - VIM specific

MM5 - Currently not defined by ETSI MEC.



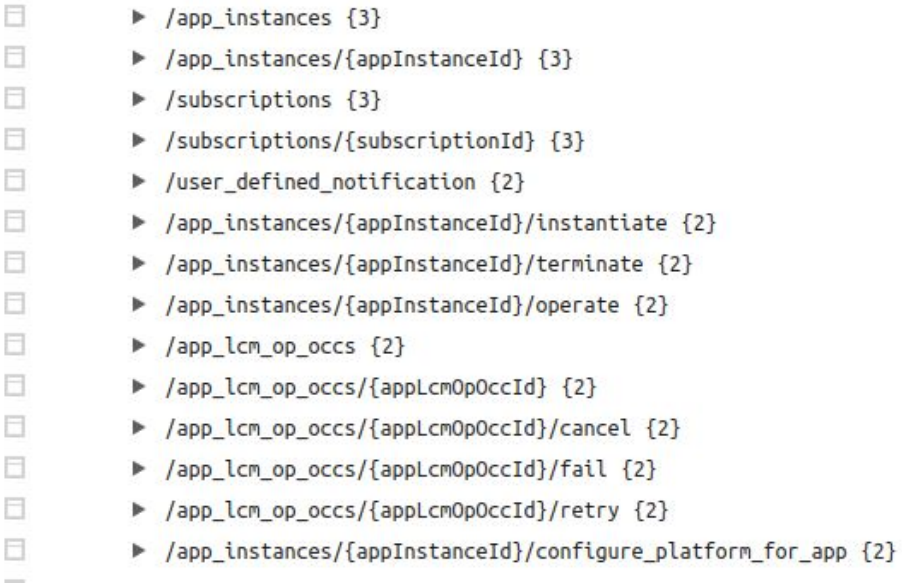
API Details - MM1 & MM3

The operations of application instance lifecycle management are:

- Instantiate.
- Operate.
- Terminate.

The MEO can also subscribe to notifications sent by the MEPM when an application instance LCM operation occurrence changes its state.

APPLCM



```
□  ▶ /app_instances {3}
□  ▶ /app_instances/{appInstanceId} {3}
□  ▶ /subscriptions {3}
□  ▶ /subscriptions/{subscriptionId} {3}
□  ▶ /user_defined_notification {2}
□  ▶ /app_instances/{appInstanceId}/instantiate {2}
□  ▶ /app_instances/{appInstanceId}/terminate {2}
□  ▶ /app_instances/{appInstanceId}/operate {2}
□  ▶ /app_lcm_op_occs {2}
□  ▶ /app_lcm_op_occs/{appLcmOpOccId} {2}
□  ▶ /app_lcm_op_occs/{appLcmOpOccId}/cancel {2}
□  ▶ /app_lcm_op_occs/{appLcmOpOccId}/fail {2}
□  ▶ /app_lcm_op_occs/{appLcmOpOccId}/retry {2}
□  ▶ /app_instances/{appInstanceId}/configure_platform_for_app {2}
```

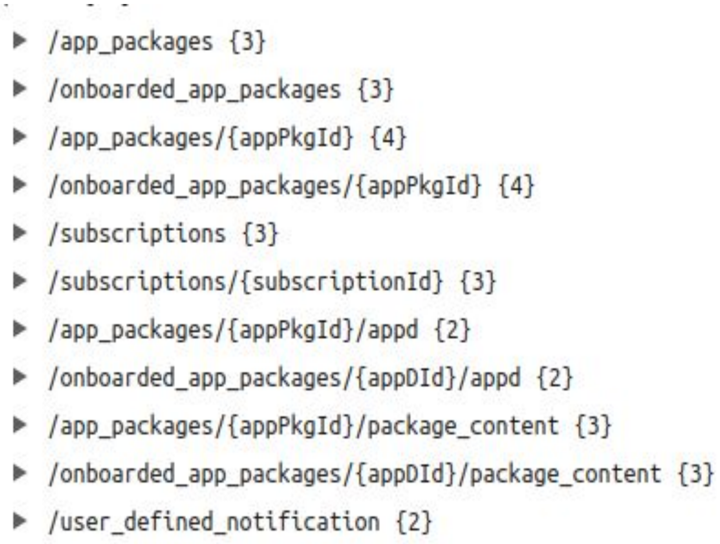
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/01002/02.01.01_60/gs_mec01002v020101p.pdf

API Details - MM1 & MM3

Message flows for application package management are used to make application package available to the MEC system, delete the application package from the MEC system, or query information of one or more application packages. The series of message flows include:

- On-board application package.
- Query application package information.
- Disable application package.
- Enable application package.
- Delete application package.
- Fetch application package.

APPPKG Mgmt



▶ /app_packages {3}

▶ /onboarded_app_packages {3}

▶ /app_packages/{appPkgId} {4}

▶ /onboarded_app_packages/{appPkgId} {4}

▶ /subscriptions {3}

▶ /subscriptions/{subscriptionId} {3}

▶ /app_packages/{appPkgId}/appd {2}

▶ /onboarded_app_packages/{appDId}/appd {2}

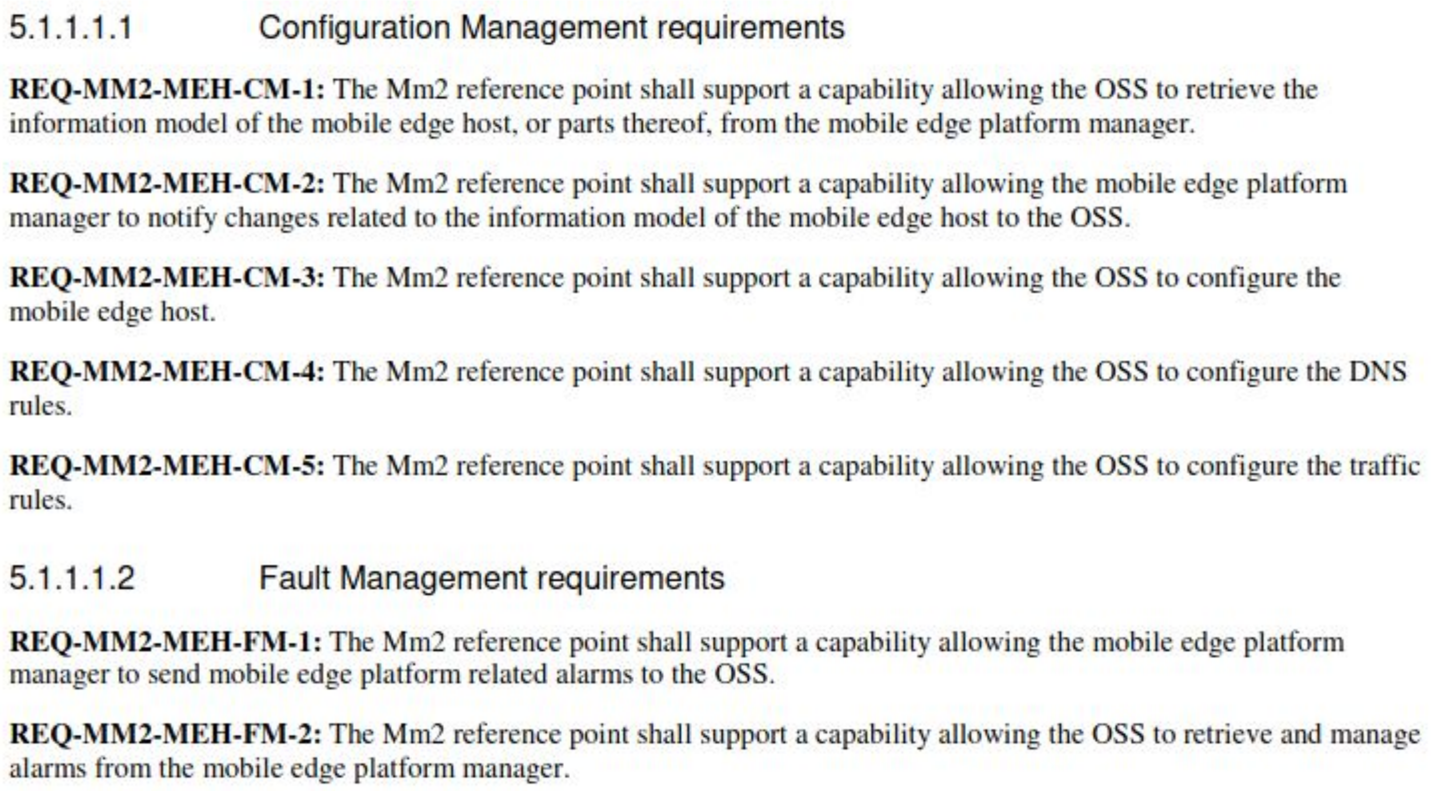
▶ /app_packages/{appPkgId}/package_content {3}

▶ /onboarded_app_packages/{appDId}/package_content {3}

▶ /user_defined_notification {2}

API Details - MM2

1. Infra management - configuration & fault monitoring
2. Dynamic Traffic & DNS rule configuration



5.1.1.1.1 Configuration Management requirements

REQ-MM2-MEH-CM-1: The Mm2 reference point shall support a capability allowing the OSS to retrieve the information model of the mobile edge host, or parts thereof, from the mobile edge platform manager.

REQ-MM2-MEH-CM-2: The Mm2 reference point shall support a capability allowing the mobile edge platform manager to notify changes related to the information model of the mobile edge host to the OSS.

REQ-MM2-MEH-CM-3: The Mm2 reference point shall support a capability allowing the OSS to configure the mobile edge host.

REQ-MM2-MEH-CM-4: The Mm2 reference point shall support a capability allowing the OSS to configure the DNS rules.

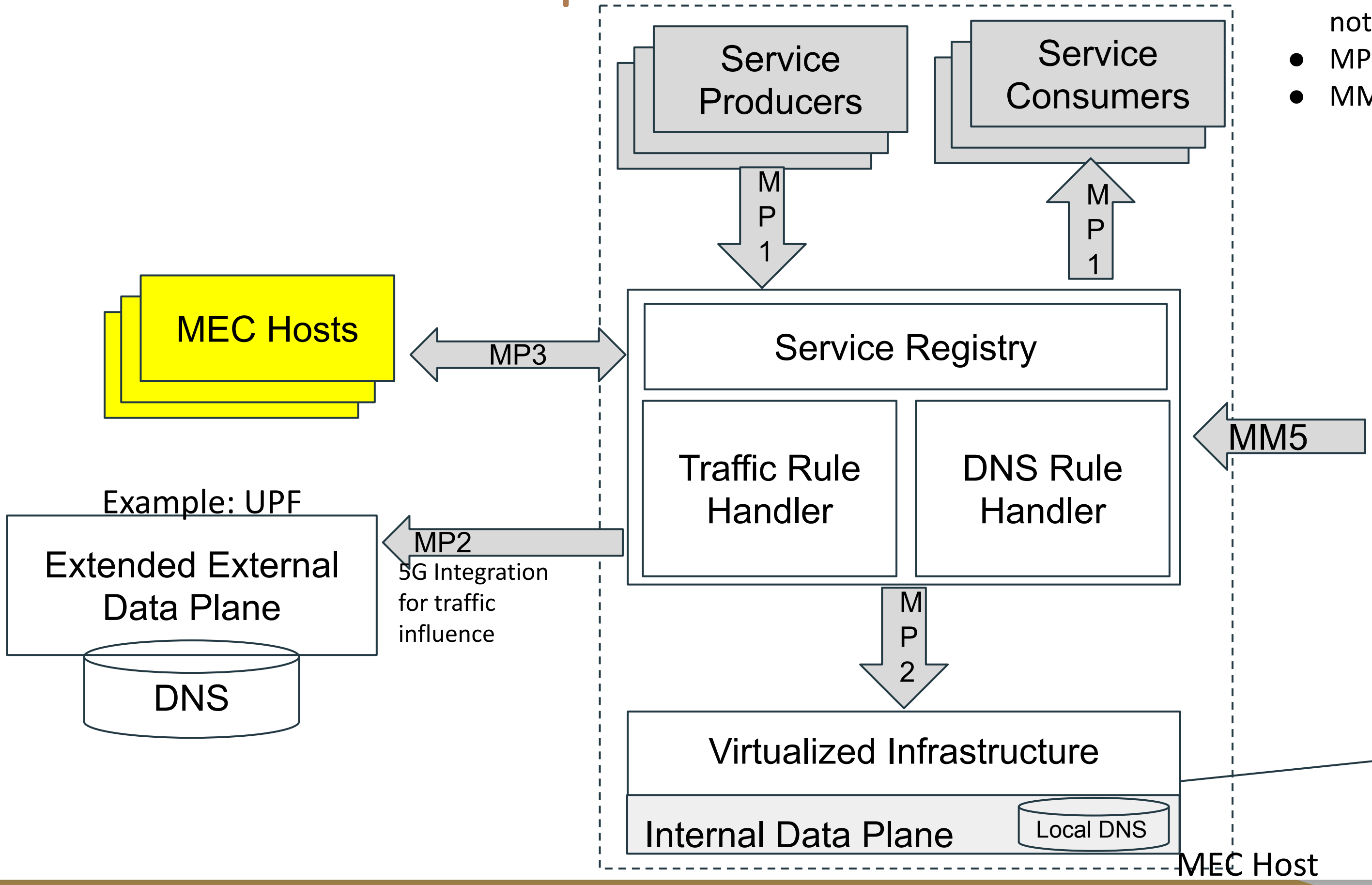
REQ-MM2-MEH-CM-5: The Mm2 reference point shall support a capability allowing the OSS to configure the traffic rules.

5.1.1.1.2 Fault Management requirements

REQ-MM2-MEH-FM-1: The Mm2 reference point shall support a capability allowing the mobile edge platform manager to send mobile edge platform related alarms to the OSS.

REQ-MM2-MEH-FM-2: The Mm2 reference point shall support a capability allowing the OSS to retrieve and manage alarms from the mobile edge platform manager.

Platform APIs Simplified



- MP2 is Data Plane specific and not defined by ETSI
- MP3 is WIP
- MM5 is again not defined yet.

https://forge.etsi.org/swagger/ui/?url=https://forge.etsi.org/gitlab/mec/gs011-app-enablement-api/raw/v2.2.1/MecAppSupportApi.yaml

API Details - MP1 - App Support

MEC application assistance:

- MEC application start-up procedure;
- MEC application graceful termination/stop;

Traffic routing:

- traffic rules update, activation and deactivation;

DNS rules:

- DNS rules activation and deactivation;

Timing:

- providing access to time of day information;

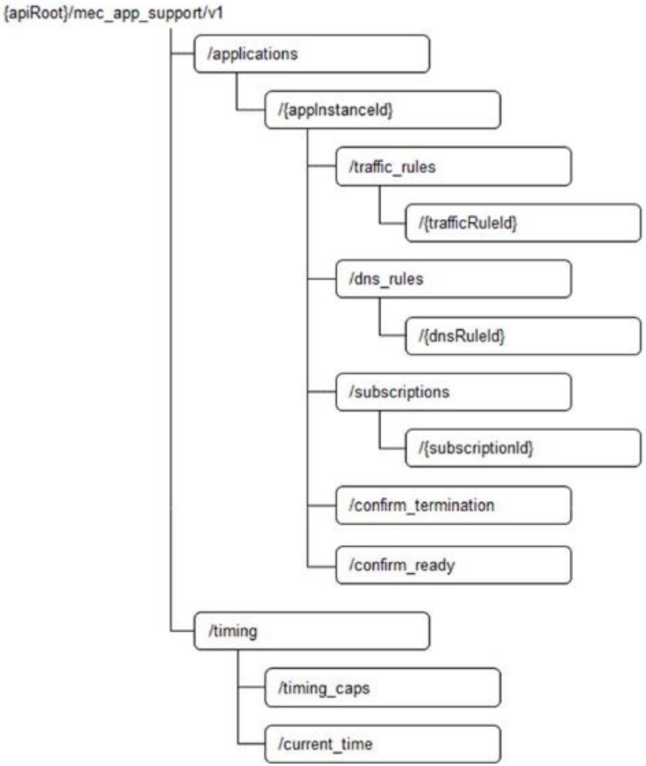


Figure 7.2.2-1: Resource URI structure of the MEC application support API

https://www.etsi.org/deliver/etsi_gs/MEC/001_099/011/02.01.01_60/gs_mec011v020101p.pdf

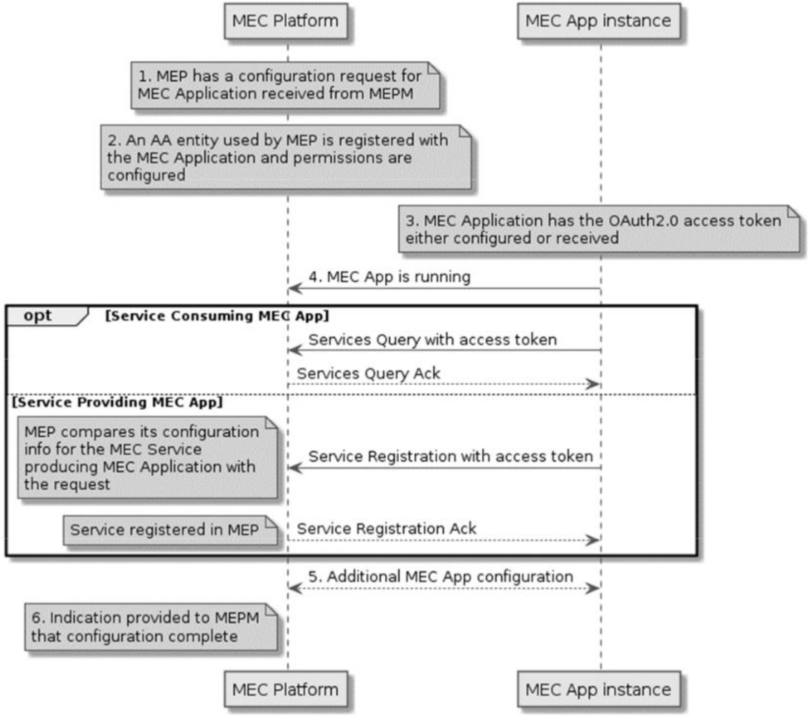

MEC Platform
MEC App instance
1. MEP has a configuration request for MEC Application received from MEPM
2. An AA entity used by MEP is registered with the MEC Application and permissions are configured
3. MEC Application has the OAuth2.0 access token either configured or received
4. MEC App is running
opt
[Service Consuming MEC App]
Services Query with access token
Services Query Ack
[Service Providing MEC App]
MEP compares its configuration info for the MEC Service producing MEC Application with the request
Service Registration with access token
Service registered in MEP
Service Registration Ack
5. Additional MEC App configuration
6. Indication provided to MEPM that configuration complete
MEC Platform
MEC App instance

Figure 5.2.2-1: Flow of MEC application start up

After Application is UP, it provide acknowledgement to MEP via this message

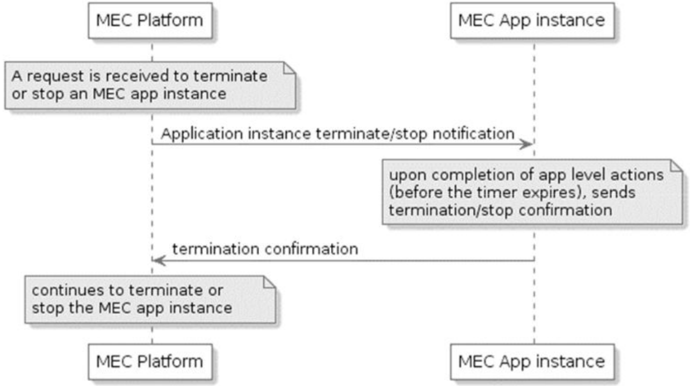

MEC Platform
MEC App instance
A request is received to terminate or stop an MEC app instance
Application instance terminate/stop notification
upon completion of app level actions (before the timer expires), sends termination/stop confirmation
termination confirmation
continues to terminate or stop the MEC app instance
MEC Platform
MEC App instance

Figure 5.2.3-1: Example flow of MEC application instance graceful termination/stop

Triggered via MEPM to terminate APP, MEP try to gracefully terminate it and remove things like DNS/Traffic rule too

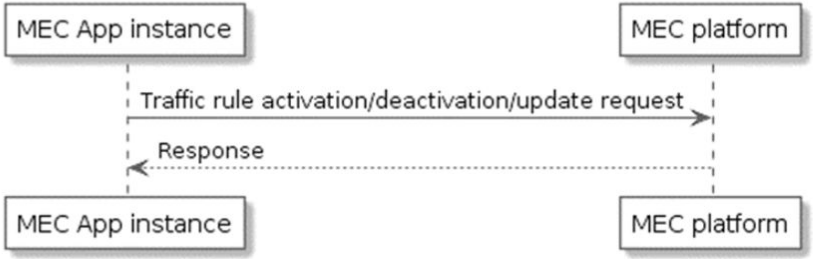


Figure 5.2.7-1: Flow of traffic rule activation/deactivation/update

APP inform MEP and MEP via MP2 activate traffic/DNS rule

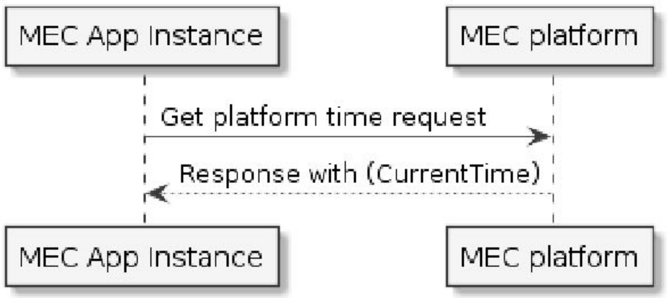


Figure 5.2.10.2-1: Flow of MEC application requesting platform time

- MEC applications may require TOD information for notifications, logs and special events time notions, packets receipt and transmit timestamping and other needs depending on application purpose.
- Low accuracy TOD information may be provided to the application by use of simple procedure of current time retrieval from the platform. Higher TOD accuracy may be achieved by use of special protocols that allows timing transfer over packet networks, such as NTP

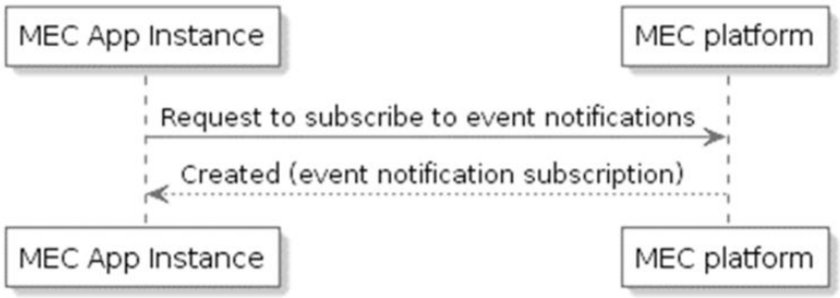

MEC App Instance
MEC platform
Request to subscribe to event notifications
Created (event notification subscription)
MEC App Instance
MEC platform

Figure 5.2.6.2-1: Flow of Subscribing to event notifications

- Application instance terminate/stop notification: MEP will try graceful termination iff APP has subscribed for it

API Details - MP1

MEC service assistance:

- authentication and authorization of producing and consuming MEC services;
- a means for service producing MEC applications to register/deregister towards the MEC platform the MEC services they provide, and to update the MEC platform about changes of the MEC service availability;
- a means to notify the changes of the MEC service availability to the relevant MEC application;
- discovery of available MEC services;

Transport information:

- This method is typically used by a service-producing application to discover transports provided by the MEC platform in the "transport information query" procedure

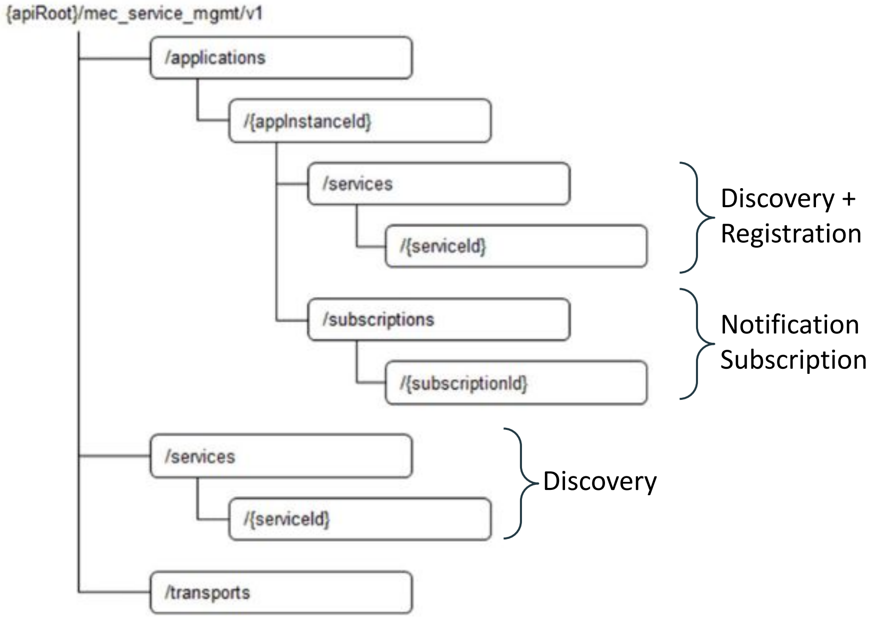


Figure 8.2.2-1: Resource URI structure of the MEC service management API

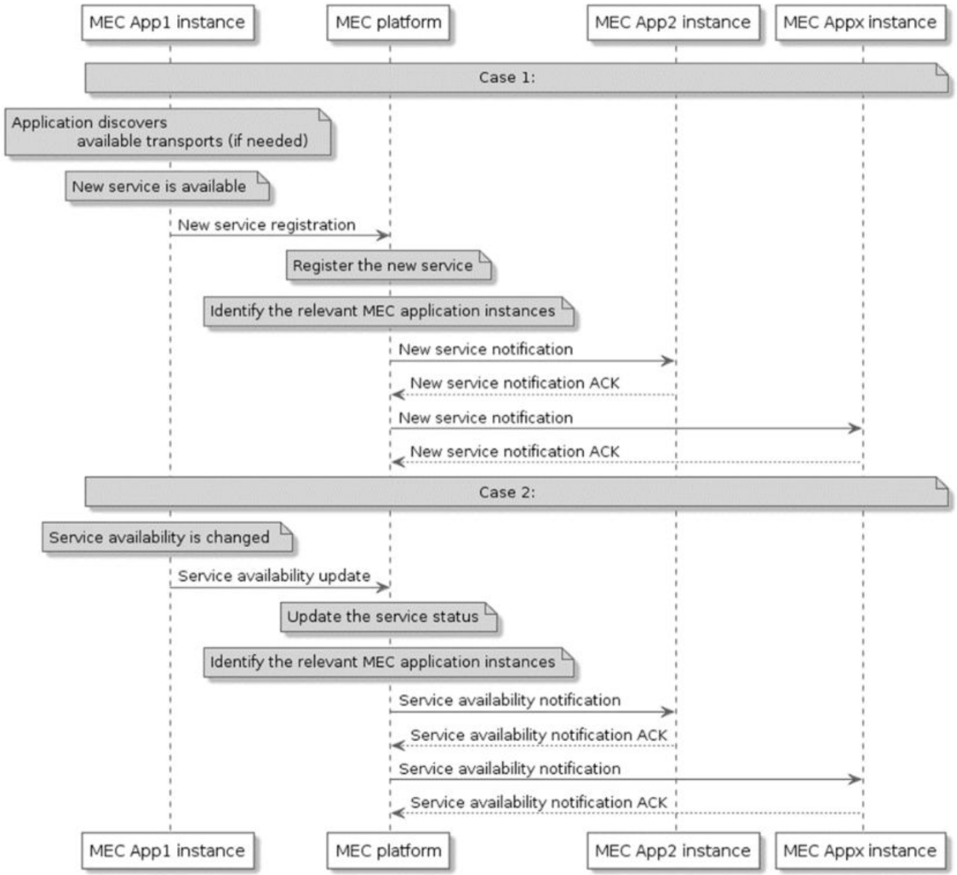

MEC App1 instance
MEC platform
MEC App2 instance
MEC Appx instance
Case 1:
Application discovers
available transports (if needed)
New service is available
New service registration
Register the new service
Identify the relevant MEC application instances
New service notification
New service notification ACK
New service notification
New service notification ACK
Case 2:
Service availability is changed
Service availability update
Update the service status
Identify the relevant MEC application instances
Service availability notification
Service availability notification ACK
Service availability notification
Service availability notification ACK
MEC App1 instance
MEC platform
MEC App2 instance
MEC Appx instance

Figure 5.2.4-1: Flow of new service registration and service availability update

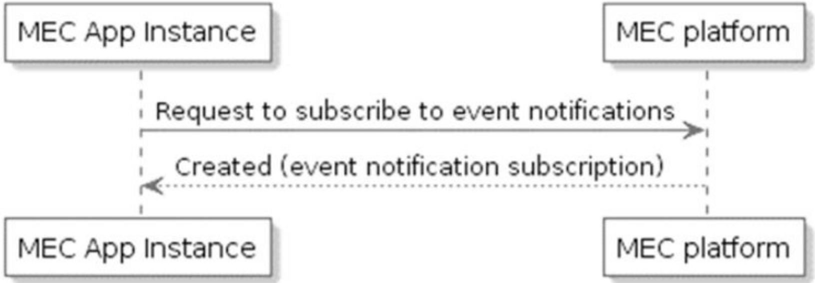

MEC App Instance
MEC platform
Request to subscribe to event notifications
Created (event notification subscription)
MEC App Instance
MEC platform

Figure 5.2.6.2-1: Flow of Subscribing to event notifications

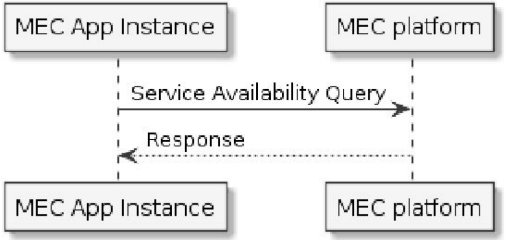


Figure 5.2.5-1: Flow of MEC application requesting service availability information

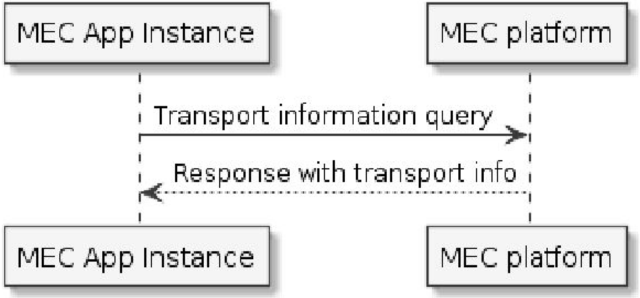


Figure 5.2.9-1: Flow of MEC application requesting transport information

APP to obtain transport information (REST, GRPC etc) provided by platform. But same come as part of service discovery too??

Application Instantiation Flow

https://www.etsi.org/deliver/etsi_gs/MEC/001_099/01002/02.01.01_60/gs_mec01002v020101p.pdf

Instantiation Triggering Points

1. Customer (CFS) Portal: Manually Pre-Deployed
2. Client Application Triggered: OnDemand

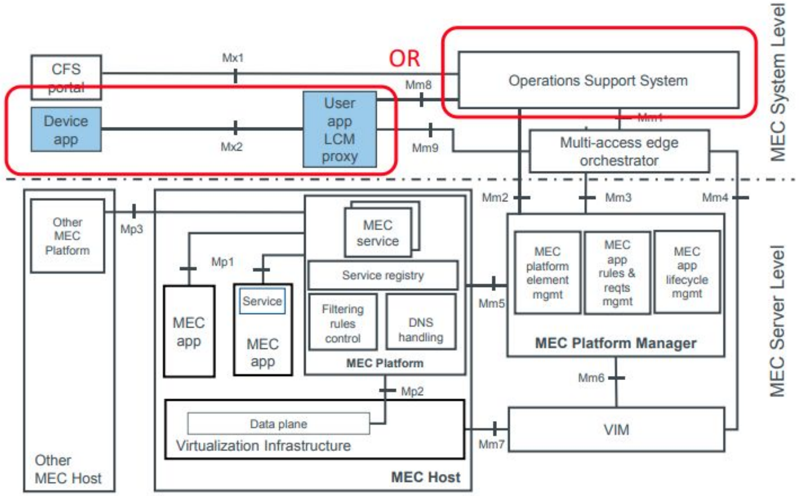
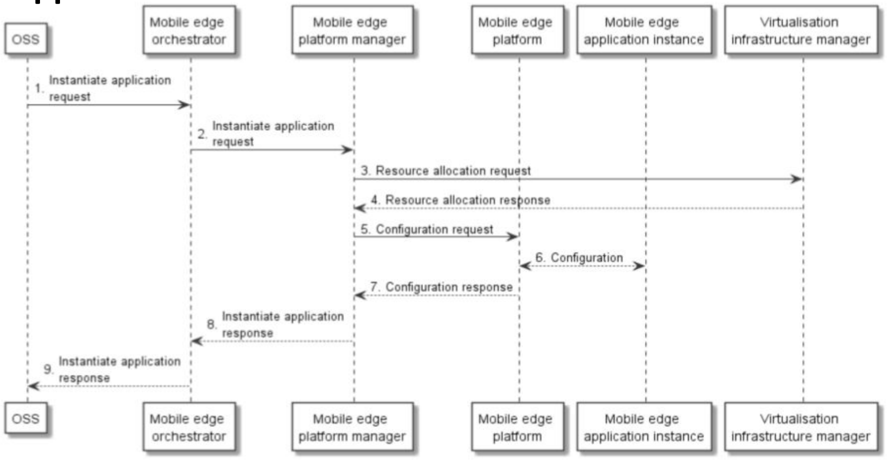


Figure 5: MEC app instantiation options

Application/Package OnBoarding

The MEO assigns an application package ID and provides the MEC Platform Manager (MEPM) with the location of the application image if it is not yet on-boarded in the MEC system. The MEPM prepares the Virtualized Infrastructure Managers (VIMs), selected by the MEO for application instantiation, by providing the necessary infrastructure configuration information and sending the application images, which are then stored by the VIM.

Application Instantiation



OSS
Mobile edge orchestrator
Mobile edge platform manager
Mobile edge platform
Mobile edge application instance
Virtualisation infrastructure manager
1. Instantiate application request
2. Instantiate application request
3. Resource allocation request
4. Resource allocation response
5. Configuration request
6. Configuration
7. Configuration response
8. Instantiate application response
9. Instantiate application response
OSS
Mobile edge orchestrator
Mobile edge platform manager
Mobile edge platform
Mobile edge application instance
Virtualisation infrastructure manager

Figure 5.3-1: Application instantiation flow

Application Instantiation Flow

PreRequisite

1. Infra Setup
2. Edge & MEPM registration with MEO

Package Onboarding

1. User trigger Package distribution from OSS/Portal
2. MEO read package & pull image to its local image repository (optional, not as per ETSI MEC). Assign & return unique id for package.
3. MEO should upload image to VIM (not happening in EdgeGallery)
4. **MEO Distribute package to desired MEPM**

Note: As per ETSI MEC last two steps are done during application instantiation step 3.

Application Instantiation Flow

Application Instantiation

1. Application instantiation triggered by OSS/Portal with Package ID and with Edge selection information.
2. MEO finds the edge via Host selection logic / user defined edge.
3. MEO sends instantiation request to selected MEPM APPLCM.
4. APPLCM does following:
   a. Allots a AK, SK & AppInstanceID for an application and adds the same to deployment package
   b. It also configures same to MEP sub module (MEP-Auth)
   c. Sends request to VIM (kubernetes/openstack) for application instantiation.
   d. VIM pulls image and deploy the same (for ETSI MEC, image should have been pushed to VIM prehand)
   e. APPLCM wait for application up event (APP informed MEP via "ConfirmReady") Not implemented in EG
   f. After application is UP, Producer will register, consumer will subscribe (covered in MEP flow)
5. MEO sends Traffic/DNS rule configuration to APPRuleMgr (As per ETSI MEC, MEPM should open package and find traffic/dns rule to configure, but in edgegallery MEO open package)
6. AppRuleMgr Invokes MEP to configure application rules (Async)
   a. MEP flow Details (covered in MEP flow)
   b. MEP once validate and accept will confirm to AppRuleMgr
   c. AppRule Mgr continuously query status, once success or timeout will inform MEO.

Application Instantiation Flow

MEP Flow after Application is UP

1. Application is UP and sends confirm ready to MEP. (EG this step is optional)
2. If application is producer application, will register its service with MEP
3. If application is consumer application will discover required service details (EndPoint, transport etc) from MEP.
4. Additionally Consumer application will subscribe of service notification.

MEP will update all subscriber in case any change in related produced service

Note: For simplicity all Authentication part is omitted and will be covered when we will talk about MEP.

Application Instantiation Flow

MEP Flow on receiving Rule configuration

1. MEP receives request to configure traffic and dns rule from AppRuleMgr.
2. MEP returns response (async) on accepting request
3. MEP wait for Application UP even “Confirm Ready” (EG is unknown)
4. Traffic Rule handling
   a. Invokes Data Plane via MEP Adapter to configure traffic rule
5. DNS rule handling
   a. Configure DNS rule in its DNS server
   b. Configure DNS redirection rule in DNS provided by Data plane via MEP Adapter
6. Activation of Traffic and DNS rule triggered after receiving activation request from Application. (EG is unknown)
7. Update state in DB, AppRuleMgr was continuously polling and will get to know that rule is active.

To be checked for EG

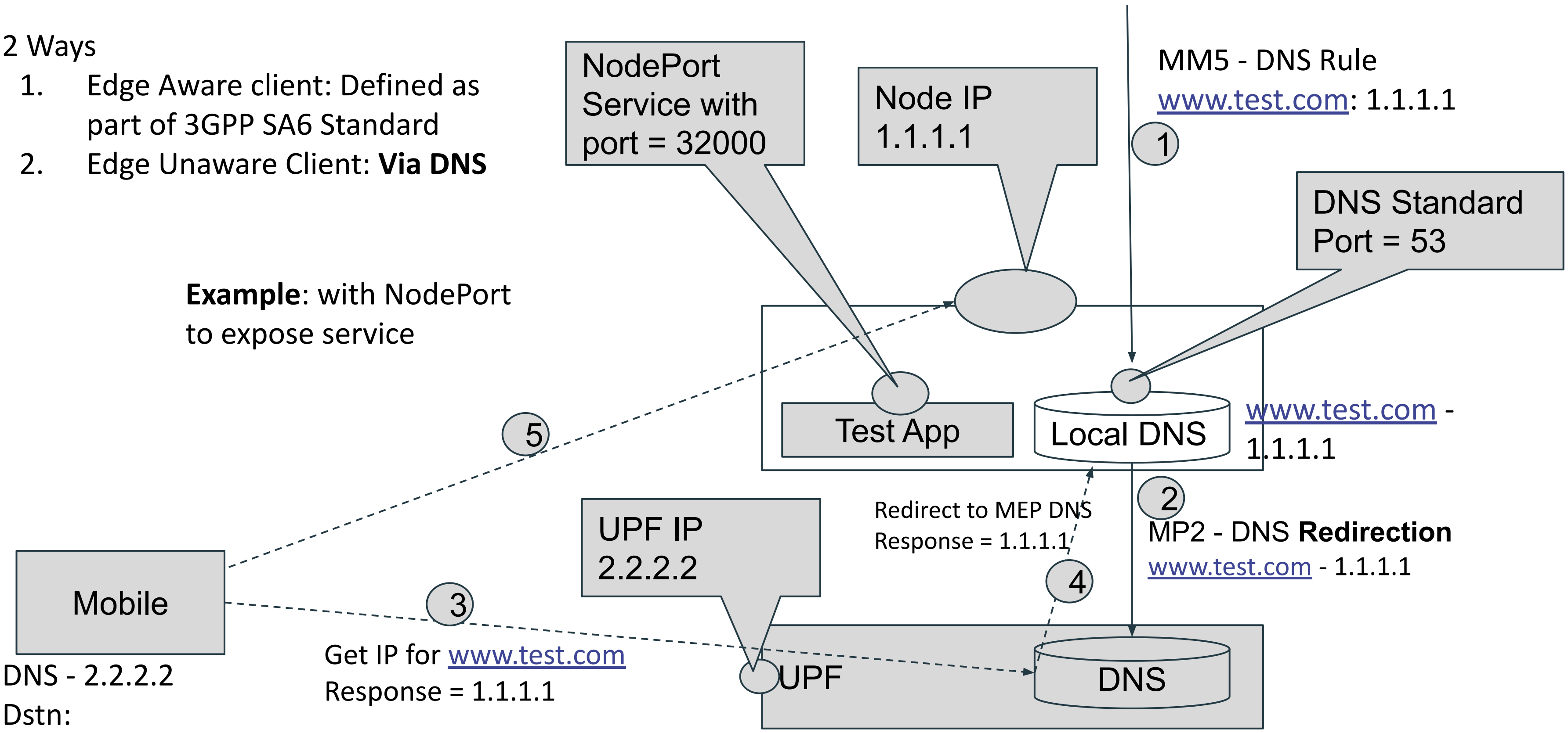
1. Whether MEP waits for confirm ready before configuring these rules or whether wait for dns/traffic activation request from application before configuring rules?? Whether configure and activate is handled separately in MEP??

Client to Edge Connection: Data Path

2 Ways

1. Edge Aware client: Defined as part of 3GPP SA6 Standard
2. Edge Unaware Client: **Via DNS**

Example: with NodePort to expose service



DNS - 2.2.2.2

Dstn: www.test.com:32000

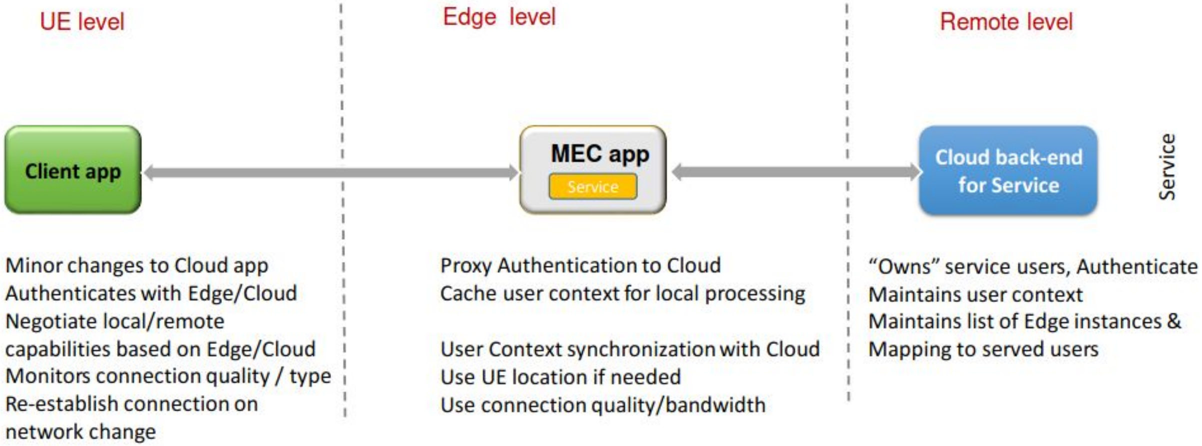
Thanks

1. **E2E Detailed Instantiation Flow**
2. **Traffic Flow**
3. Deployment (MEC in 5G, MEC in Enterprise) with Integration with 5G
4. Relation with other Edge Standards (3GPP SA6 & CAPIF & GSMA OPG)

3 Tier Architecture for Usecase Realization

Traditionally: Client-Server Architecture
With Edge: Extra Edge layer in middle and end-to-end service can be split into three applications or components: terminal device component(s), edge component(s) and remote component(s)

Terminal device: Perform some preliminary processing. Such preliminary processing requires **near zero latency** and it requires the terminal device to **support some computing capabilities**.
Edge cloud: To **offload the computing** away from the terminal device while still leveraging **very low latency** and predictable performance. Also extracting some information using RNI API or Location API.
Remote components: implement operations to be carried out in the remote data centre, e.g. to benefit from large storage and database access.



UE level:
Minor changes to Cloud app
Authenticates with Edge/Cloud
Negotiate local/remote capabilities based on Edge/Cloud
Monitors connection quality / type
Re-establish connection on network change

Edge level:
Proxy Authentication to Cloud
Cache user context for local processing

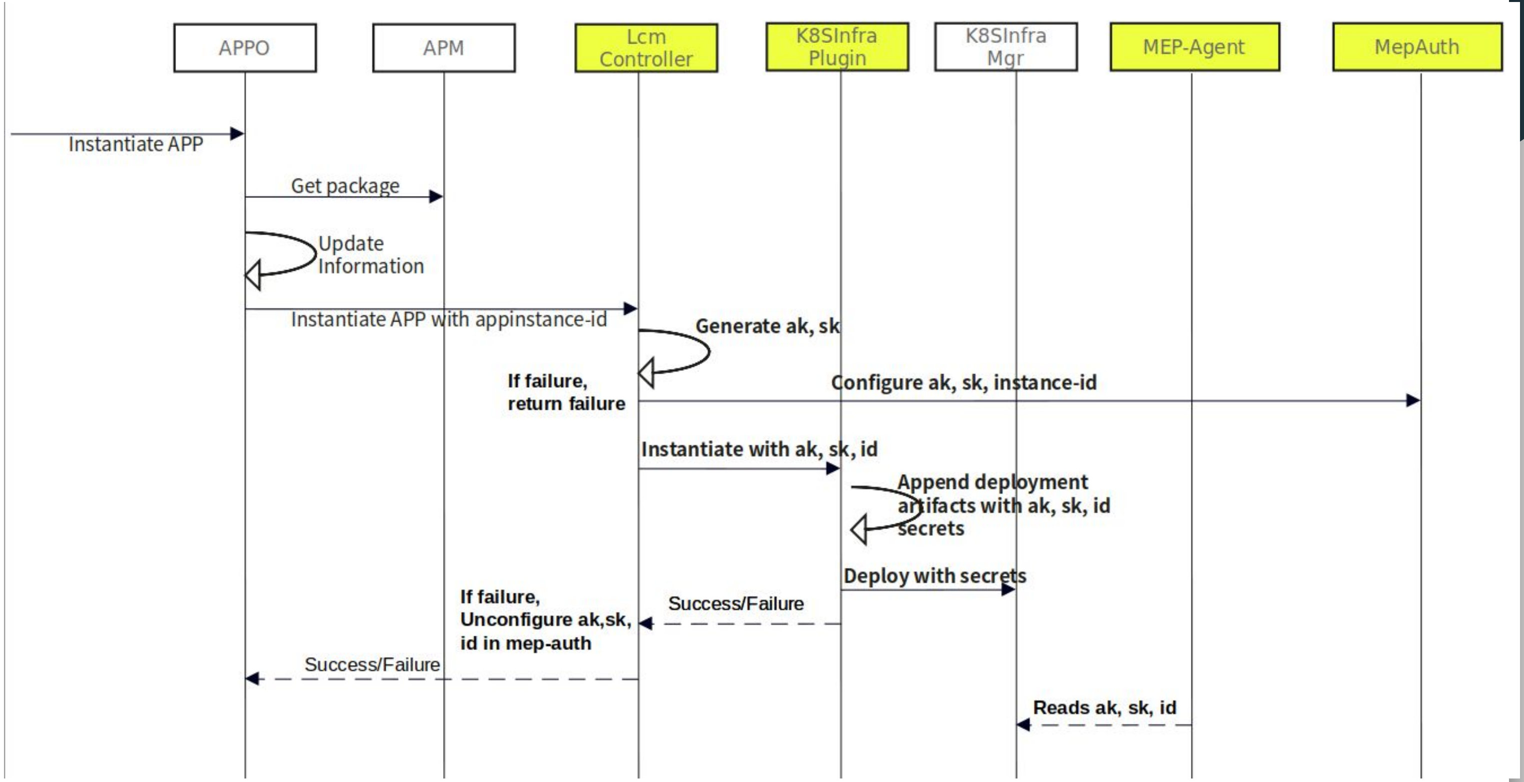
User Context synchronization with Cloud
Use UE location if needed
Use connection quality/bandwidth

Remote level:
"Owns" service users, Authenticate
Maintains user context
Maintains list of Edge instances & Mapping to served users

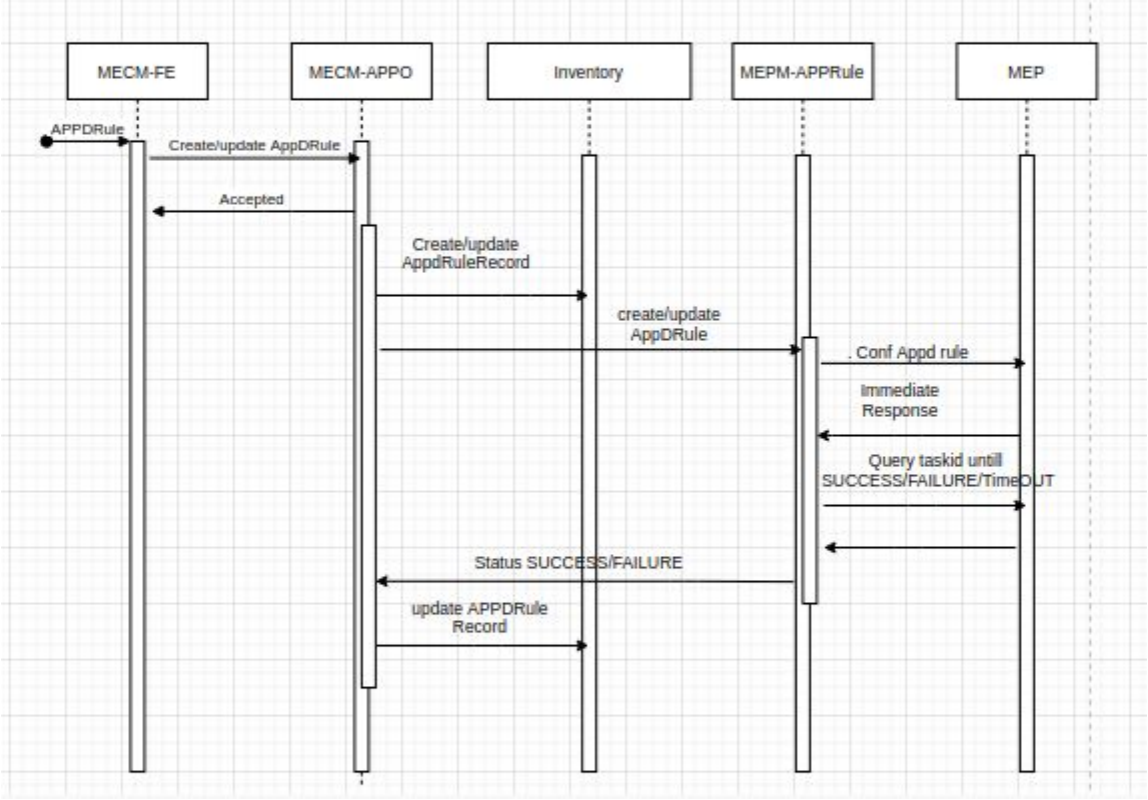
Figure 3: Example of splitting an application into "terminal", "edge" and "remote" components

Note: Depending on use case, service could leverage 1 or more layer

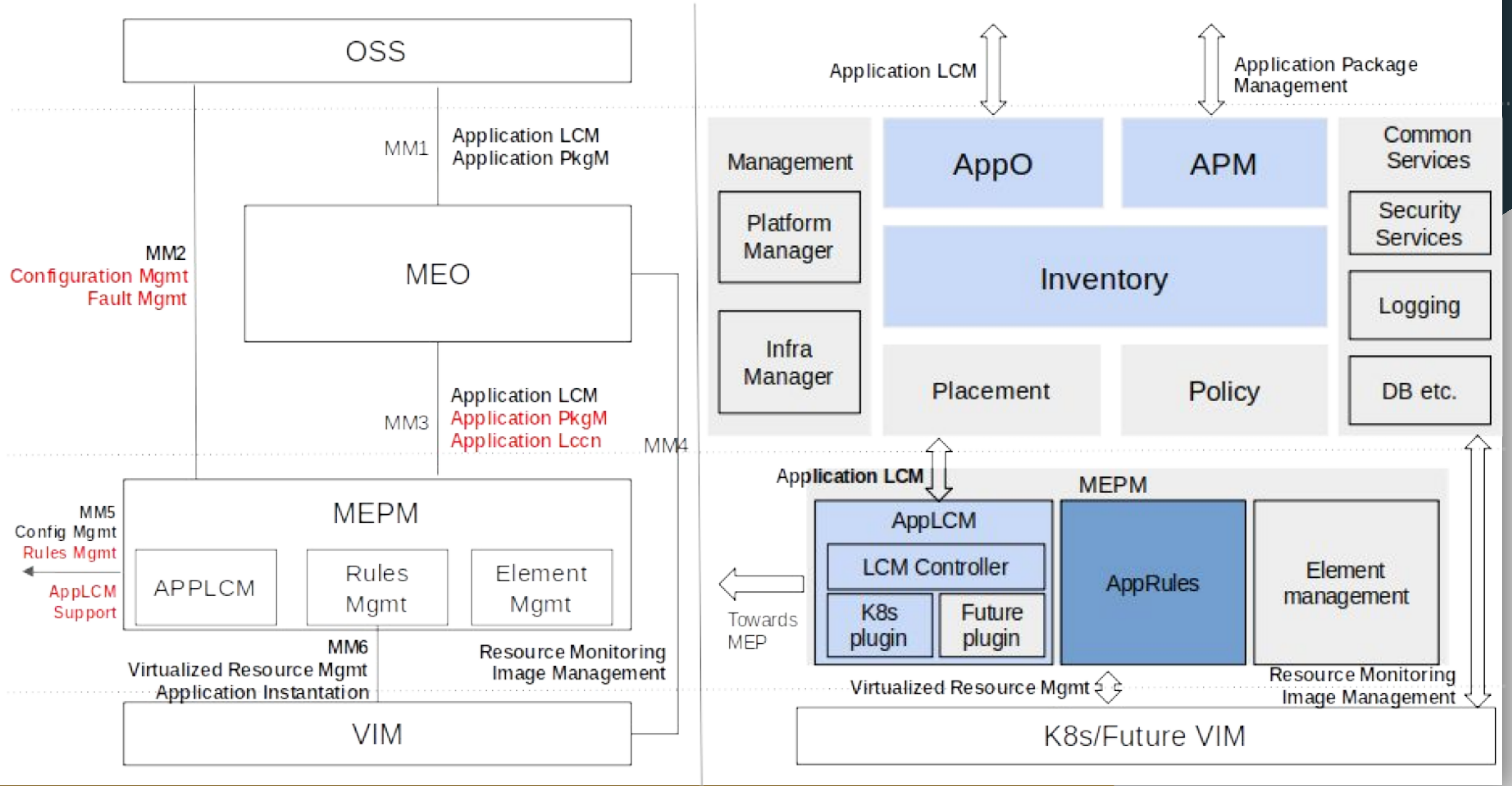
AK/SK automation flow


APPO
APM
Lcm Controller
K8SInfra Plugin
K8SInfra Mgr
MEP-Agent
MepAuth
Instantiate APP
Get package
Update Information
Instantiate APP with appinstance-id
Generate ak, sk
If failure, return failure
Configure ak, sk, instance-id
Instantiate with ak, sk, id
Append deployment artifacts with ak, sk, id secrets
Deploy with secrets
Success/Failure
If failure, Unconfigure ak,sk, id in mep-auth
Success/Failure
Reads ak, sk, id

APP rule configuration (MM2)

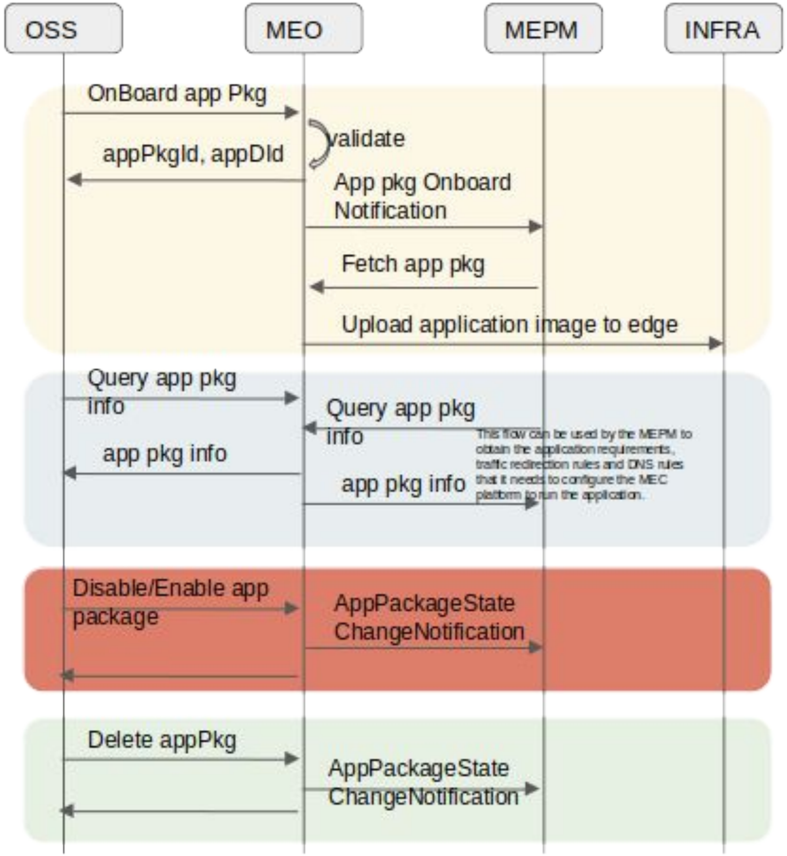


EdgeGallery API Alignment with ETSI MEC Summary (Management Part)



EdgeGallery Flow Alignment with ETSI MEC

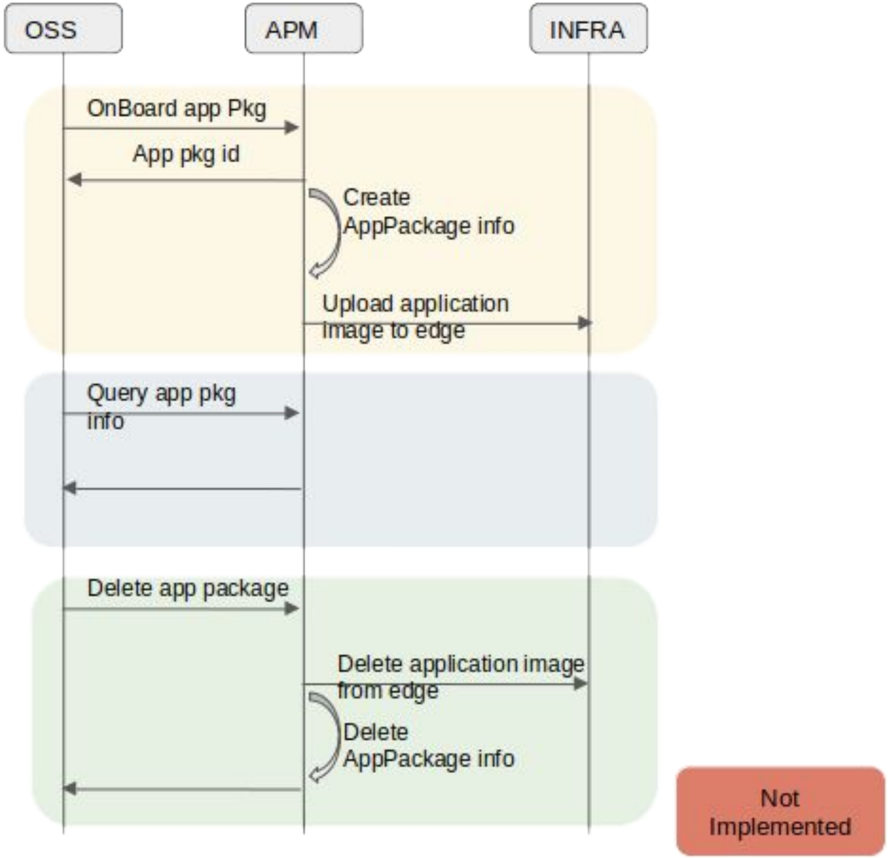
Application package management


OSS
MEO
MEPM
INFRA
OnBoard app Pkg
validate
appPkgId, appDId
App pkg Onboard Notification
Fetch app pkg
Upload application image to edge
Query app pkg info
Query app pkg info
app pkg info
app pkg info
Disable/Enable app package
AppPackageState ChangeNotification
Delete appPkg
AppPackageState ChangeNotification

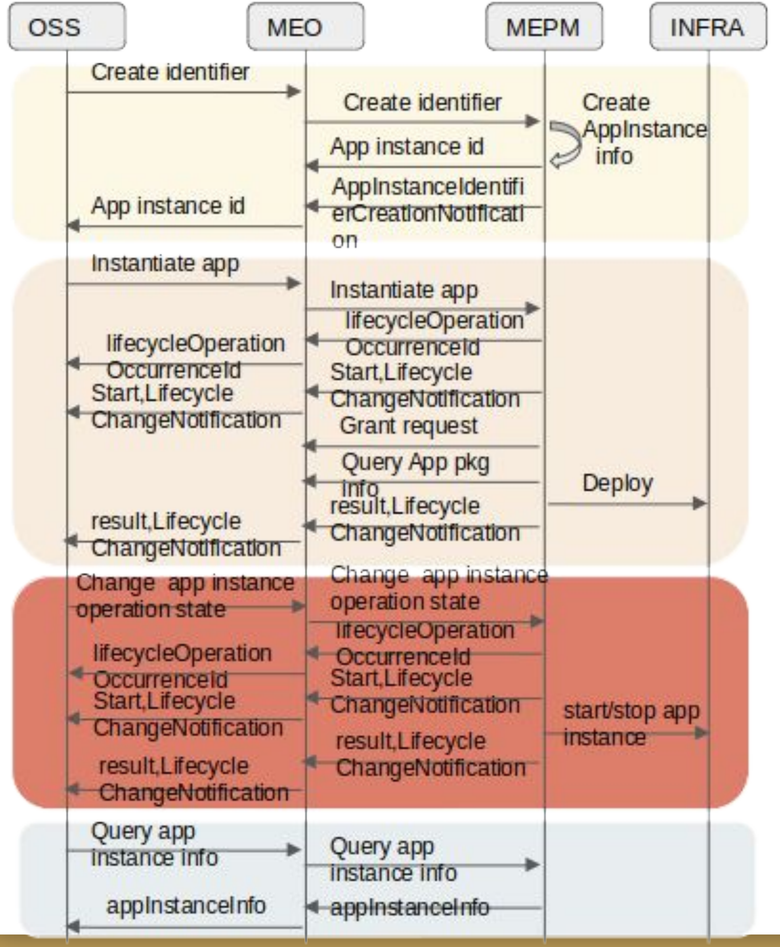
This flow can be used by the MEPM to obtain the application requirements, traffic redirection rules and DNS rules that it needs to configure the MEC platform to run the application.

EdgeGallery Existing Flow

Application package management



EdgeGallery LCM Flow Alignment with ETSI MEC



OSS

MEO

MEPM

INFRA

Create identifier

Create identifier

Create AppInstance info

App instance id

AppInstanceIdentifierCreationNotification

App instance id

Instantiate app

Instantiate app

lifecycleOperation OccurrenceId

lifecycleOperation OccurrenceId

Start,Lifecycle ChangeNotification

Start,Lifecycle ChangeNotification

Grant request

Query App pkg info

Deploy

result,Lifecycle ChangeNotification

result,Lifecycle ChangeNotification

Change app instance operation state

Change app instance operation state

lifecycleOperation OccurrenceId

lifecycleOperation OccurrenceId

Start,Lifecycle ChangeNotification

Start,Lifecycle ChangeNotification

start/stop app instance

result,Lifecycle ChangeNotification

result,Lifecycle ChangeNotification

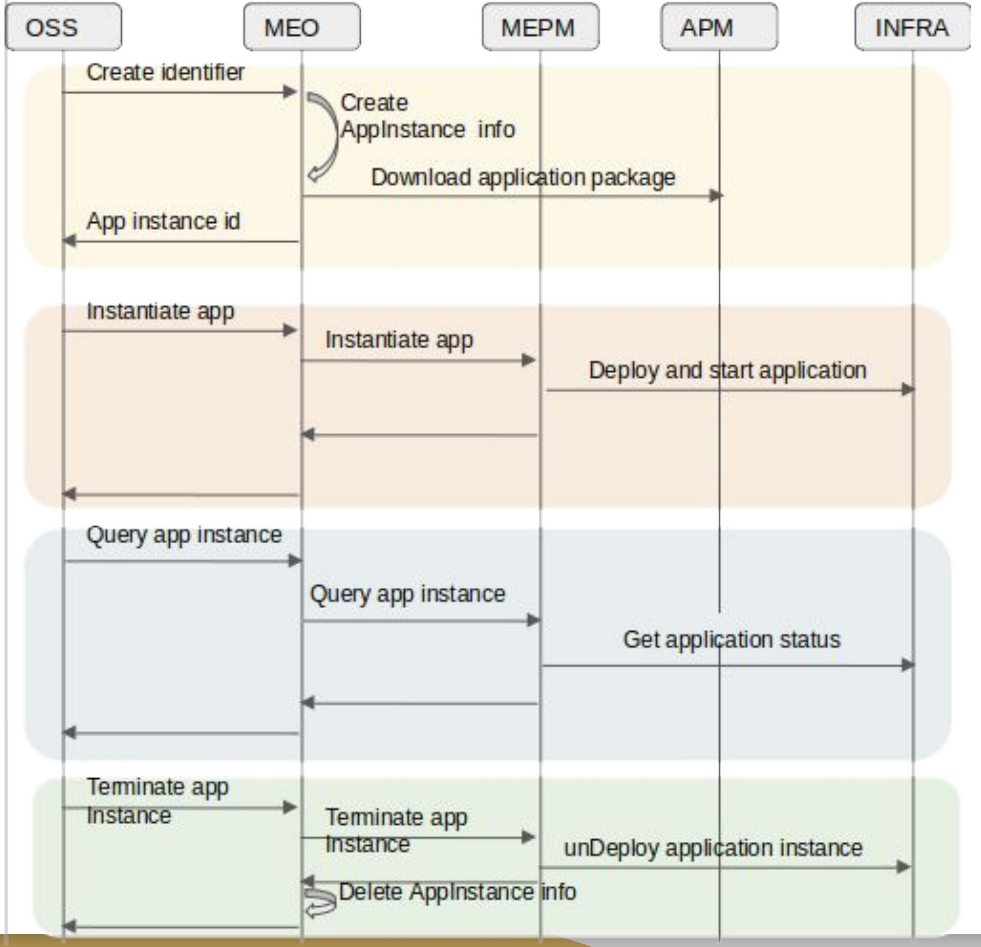
Query app instance info

Query app instance info

appInstanceInfo

appInstanceInfo

EdgeGallery Existing LCM Flow



OSS

MEO

MEPM

APM

INFRA

Create identifier

Create AppInstance info

Download application package

App instance id

Instantiate app

Instantiate app

Deploy and start application

Query app instance

Query app instance

Get application status

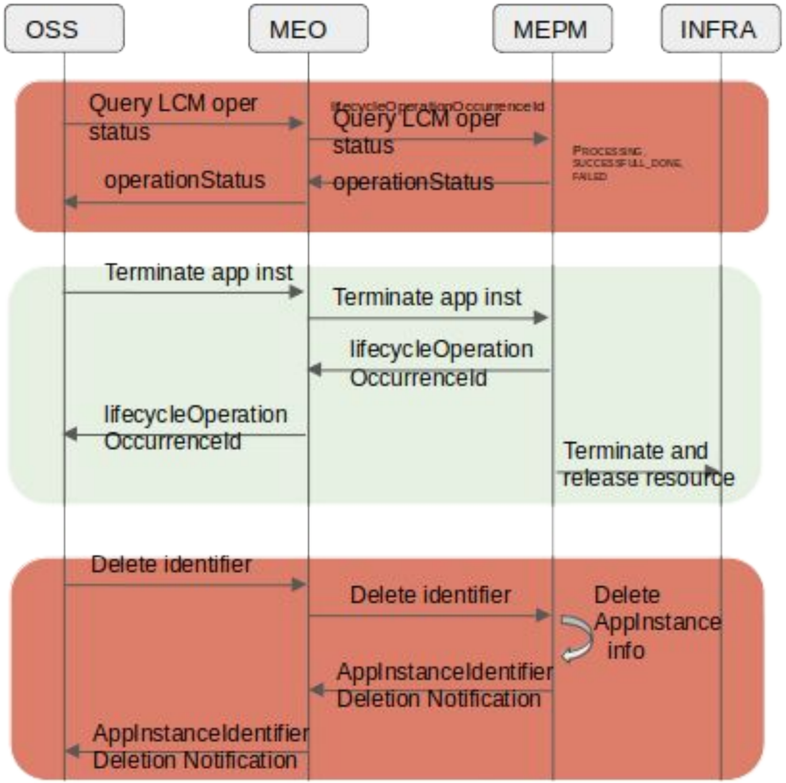
Terminate app Instance

Terminate app Instance

unDeploy application instance

Delete AppInstance info

EdgeGallery LCM Flow Alignment with ETSI MEC cont..



OSS

MEO

MEPM

INFRA

Query LCM oper status

lifecycleOperationOccurrenceId

Query LCM oper status

PROCESSING,
SUCCESSFULL_DONE,
FAILED

operationStatus

operationStatus

Terminate app inst

Terminate app inst

lifecycleOperation OccurrenceId

lifecycleOperation OccurrenceId

Terminate and release resource

Delete identifier

Delete identifier

Delete AppInstance info

AppInstanceIdentifier Deletion Notification

AppInstanceIdentifier Deletion Notification

🟥 Not Implemented