ETSI White Paper No. #59

Enabling Multi-access Edge Computing in Internet-of-Things: how to deploy ETSI MEC and oneM2M

1st edition – June 2023

Authors:
Dario Sabella, Roland Hechwartner, Enrico Scarrone, Samar Shailendra, JaeSeung Song, Bob Flynn, Arif Ishaq, Laurent Velez, Robert Gazda, Lee Jieun

ETSI
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org

About the authors

Dario Sabella
Intel, ETSI ISG MEC Chair

Roland Hechwartner
Deutsche Telekom, oneM2M Technical Plenary Chair

Enrico Scarrone
TIM, ETSI TC SmartM2M Chair, oneM2M Steering Committee Chair

JaeSeung Song
Sejong University, oneM2M Technical Plenary Vice Chair

Lee Jieun
Sejong University, oneM2M delegate

Bob Flynn
Exacta Global Smart Solutions, oneM2M TDE WG Vice Chair

Arif Ishaq
Athonet, ETSI ISG MEC delegate

Laurent Velez
ETSI CTI, MEC and oneM2M

Robert Gazda
InterDigital, ETSI MEC delegate

Samar Shailendra
Intel, 3GPP SA6 Prime and TSDSI SGN Vice Chair

Contents

Executive Summary

There are a vast number of Internet of Things (IoT) use cases where the computing capacity would be best deployed between the edge of the network and the IoT end-devices. Multi-access Edge Computing (MEC) is envisioned to provide geographical presence and distribution of compute sites at cell towers, creating value-added connectivity and easy deployment of services for sensitive delay IoT applications. At the same time the need for interoperability of machine-to-machine (M2M) communications protocols in deployed IoT systems motivated the development of an IoT standard platform known as oneM2M. Using oneM2M IoT systems developers do not need to master integrated communication protocols to design and deploy IoT applications. The integration of these two architectures enhances the benefits of each of them, making them more attractive for edge deployment of IoT systems. The combination of oneM2M and ETSI ISG MEC frameworks also enables the IoT ecosystem to benefit some of the key 5G technologies that allow deployment of IoT applications tailored according to the end users' requirements.

However, to realize the enablement of edge computing for IoT services, the oneM2M system can be enhanced to allow the interworking, as well as maximize the advantages that can be made from the exposure of MEC service. This White Paper first provides a set of relevant use cases for edge IoT environments, and then a comprehensive description of the ETSI ISG MEC framework, ETSI MEC's API, oneM2M standard, and oneM2M common services, where authors argue that network function virtualization applied to oneM2M common service layer would provide the ability to place initial workloads into the edge. In addition, a description of the possible interworking model between oneM2M and ETSI ISG MEC architectures is provided, as a suitable clarification for the readers that may want to navigate into those technologies, both for system implementation purposes (infrastructure owners) and also for application design purposes (application developers). The interworking among these standard architectures can in fact enable IoT edge services (e.g., mobility in smart city, real-time analytical functions) to exploit oneM2M common service layer instantiated on top of MEC host and enable the MEC applications based IoT system to use IoT devices' data generated through the oneM2M standard and supporting MEC system architecture and APIs.

In summary, the paper aims at providing suitable clarifications that may facilitate interoperability and better adoption of edge computing and IoT technologies through coherent standard solutions.

1 Introduction

Internet-of-Things (IoT) and edge computing are two key technology trends that continue to attract a growing interest from industry. Bringing compute capabilities to the so called “Edge” are likely to influence future communication systems and enable new services for consumer and business customers.

Edge computing, an evolution of cloud computing, brings application and data hosting from centralized cloud data centers to the network edge, closer to the consumers and the applications that use the data. Edge computing is acknowledged as one of the key pillars for meeting the demanding Key Performance Indicators (KPIs) of 5G, especially as far as low latency and bandwidth efficiency are concerned. However, not only is edge computing in telecommunications networks a technical enabler for the demanding KPIs, also plays an essential role in the transformation of the telecommunications business, where it telecommunications networks are turning into versatile service platforms for industry and other specific customer segments.

IoT technology embeds a network function that can communicate with various devices around us, such as sensors, lights, switches, TVs, and more. It connects these devices to the Internet and enables them to communicate with each other without human interaction. This technology allows people to receive more advanced and convenient services by enabling communication between devices. Early IoT services were mainly limited to user-oriented services that monitored and controlled IoT devices installed in smart homes or buildings. However, as the IoT technology has become more common and advanced, it is being used in many fields such as smart cities, smart factories, smart agriculture, and smart homes in our daily lives. Furthermore, IoT technology is becoming an infrastructure technology that collects and manages essential data for core technologies such as artificial intelligence, cloud computing, blockchain, and edge computing that will lead to the fourth industrial revolution. In recent years, IoT technology has been used to provide more accurate data not only for services such as digital twin-based smart factories that require real-time operation, but also for services such as autonomous vehicles that require high-speed communication and large-capacity data processing in close proximity to users.

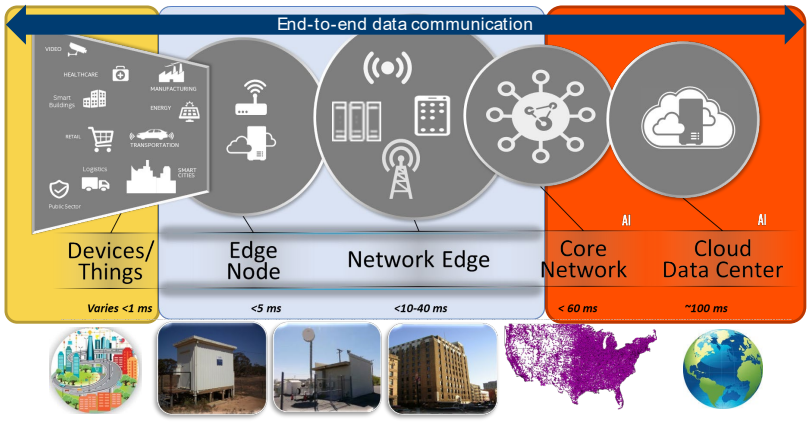

Figure 1: IoT in edge computing environments

This White Paper aims to bring together Internet-of-Things and edge computing standardization efforts from ETSI MEC and oneM2M groups (both described in the present clause), in order to provide a

comprehensive view for the reader on what is available in the standards. We describe how the two architectures complement each other and how to deploy oneM2M in a MEC environment. Finally, we identify some possible gaps and future work. The rest of the White Paper is organized as follows: Clause 2 describes some relevant use cases for edge IoT scenarios; Clause 3 is focused on ETSI MEC and oneM2M architectures, and how they can be deployed together; Clause 4 provides some deployment considerations on edge IoT environments, and finally Clause 5 concludes the White Paper.

1.1 Overview of ETSI ISG MEC

ETSI ISG MEC (European Telecommunications Standards Institute Industry Specification Group for Multi-access Edge Computing) is the home of technical standards for edge computing. The group has published a set of specifications (Phase 1) focusing on MEC architecture (ETSI GS MEC 003 [1]), management and orchestration (MANO) of MEC applications (see ETSI GS MEC 010-2 [2], application enablement and related service Application Programming Interfaces (APIs) (see ETSI GS MEC 011 [3]) and the User Equipment (UE) application API (See ETSI GS MEC 016 [4]). The MANO and application enablement functions define service environments in edge data centers, while the service APIs provide exposure of underlying network information and capabilities to applications. One of the key value-adding features of the MEC specification is this ability for applications to gain contextual information and real-time awareness of their local environment using these standardized APIs. This local services environment is a flexible and extendable framework, as new services can be introduced by following the API guidelines in ETSI GS MEC 009 [5], when creating new service APIs. And finally, the UE application API lets the client application in the UE interact with the MEC system for application lifecycle management [2].

Phase 2 of MEC (period 2018-2020) was focused on the evolution of Phase 1 and closure of the related open items, like Application Mobility (See ETSI GS MEC 021 [6]) and support for Lawful Intercept (see ETSI GS MEC 026 [7]). In particular, MEC expanded its scope from “Mobile” to “Multi-Access” edge computing, thus covering not only cellular networks (MEC integration in 5G networks, (see ETSI GR MEC 031 [8]) but also Wi-Fi (see ETSI GS MEC 028 [9]) and Fixed Access (see ETSI GS MEC 029 [10]). The group also conducted a study for the MEC deployment in NFV environments (see ETSI GR MEC 017 [11]) that led to architectural updates (see ETSI GS MEC 003 [1]). It also clarified the support for alternative virtualization technologies (e.g., on Container Support (see ETSI GR MEC 027 [12]) and started to address the needs of vertical markets like automotive, VR/AR and industrial automation. In particular, on the automotive domain, the group published a first study (ETSI GR MEC 022 [13]), that led to the specification ETSI GS MEC 030 [14] introducing the MEC V2X information services API. Network Slicing was studied in ETSI GR MEC 024 [15], while also – published). Moreover, for each MEC Service API published, the group provided a related OpenAPI representation in ETSI Forge website. As key effort to engage the edge ecosystem, the group supported MEC PoCs, MEC Deployment Trials and the organization of MEC Hackathons, to stimulate the adoption of MEC technologies also toward app developer communities. An additional effort on Testing and Compliance (see ETSI GR MEC-DEC 025 [16] and multipart specifications ETSI GS MEC-DEC 032 parts 1 to 3 [17]) permitted also to enable interoperability events like ETSI Plugtests, where multiple MEC Platforms and MEC Applications could verify their interworking in multi-vendor environments.

Then, in addition to the completion of the outstanding Phase 2 work, MEC Phase 3 (period 2021-2023) started to address the needs of MEC as heterogeneous clouds, thus expanding traditional cloud and NFV LCM approaches to Inter-MEC systems and MEC-Cloud systems coordination, i.e., standardizing the MEC

Federation (GR MEC 035 [18] and GS MEC 040 [19]). The group also started the study of mobile/intermittently connected and resource constrained devices (MEC 036), maintained and enhanced existing APIs (e.g., GS MEC 013 [20]), published the MEC IoT API (GS MEC 033 [21]) and a study on MEC deployments in Park enterprises (GR MEC 038 [22]).

The group in Phase 3 continued to define services that meet industry demand (e.g., Abstracted Radio Network Info for Industries, (MEC 043), and increased the emphasis on enabling developers, the group published the Application Package Format and Descriptor Specification (GS MEC 037 [23]) and continued its effort on API Serialization in ETSI Forge, by adding special efforts for the development of a MEC Sandbox (try-mec.etsi.org). Other key topics for MEC Phase 3 are MEC Security (MEC 041) and MEC Application Slices (MEC 044), paving the way for future work.

Finally, the transition from MEC Phase 3 to MEC Phase 4 (period 2024-2026) can lead to:

- more consolidated work on MEC Federation, including exposure of resources managed by multiple operators, e.g., addressing multi-domain and multi-tenancy slicing and MEC support for app slicing
- MEC architectural/service updates needed to support cloud native communication systems and edge native design for app developers (also with container support)
- introduction of proper normative work to improve security and privacy in MEC systems
- Further promotion of MEC as an attractive development environment for the industry by creating “developer-friendly environments” (e.g., portals, SDK) that enable convergence of key industry ecosystem, e.g., app developers and operators
- Further outreach efforts, e.g., Hackathons/trials in collab with open-source communities, industry groups (e.g., 5GAA, etc...).

1.2 Overview of oneM2M

oneM2M is the global standards initiative that develops technical specifications for a common Internet of Things (IoT) and Machine to Machine (M2M) Service Layer. oneM2M is creating a horizontal platform for the exchange and sharing of data among applications [24]. It also defines a distributed software layer - similar to an operating system - that facilitates unification of devices by providing a framework for interworking with different technologies. Since it started in August 2012, oneM2M was developed to be an interoperability enabler for the entire IoT and M2M ecosystem by defining a variety of common service functions and interworking technologies (e.g., oneM2M-3GPP Interworking [25], oneM2M-Modbus Interworking [26]). Edge computing are expected to be employed to mitigate the burdens on data centers and core networks and improve communication latency by acquiring, processing, and storing data at the edge of network. oneM2M has defined edge computing use cases and requirements in Stage 1 work in oneM2M Release 4 and is now developing oneM2M’s edge architecture. In some of these use cases (like the one where a vehicle driving along a road, passing several roadside units which are edge nodes in a larger system architecture), the oneM2M system will synchronize data and context information between the nodes to support uninterrupted and continuous intelligent transport services [27].

oneM2M comes from the collaboration of several major ICT Standards Development Organizations (SDOs) around the world, such as ARIB (Japan), ATIS (North America), CCSA (China), ETSI (Europe), TIA (North

America), TSDSI (India), TTA (S. Korea) and TTC (Japan)[1]. These SDOs, referred to as "Partners Type 1", share the objective of developing a set of standards for a common service layer that applies across many different industry segments. The Partners Type 1 have acted strategically to achieve a much-needed convergence in the IoT standards landscape. Instead of developing IoT standards individually and for their local markets, they agreed, in 2012, to collaborate through the oneM2M partnership project. To promote oneM2M, they facilitate the development and publication of oneM2M specifications as their own standards[2]. This ensures a global and institutional reach for oneM2M. In 2018, the ITU formally adopted the oneM2M specifications, including a full international recommendation for the oneM2M architecture[3]. Currently, there are over 200 active member companies in oneM2M. Several fora and industry alliances working on IoT-related topics also joined oneM2M. They play an important role in shaping oneM2M specifications and ensuring a coordinated approach. These are referred to as Partners Type 2, and the category presently includes Global Platform, which creates and publishes an international standard for enabling digital services and devices to be trusted and managed securely throughout their lifecycle.

Architecturally, oneM2M's common service layer [28] is a horizontal abstraction layer between IoT applications (i.e., business logic) and the communications networks that provide connectivity to end-point devices and sensors (i.e., actuation and data capture). The benefit of this abstraction layer is that users of the oneM2M specifications do not need to master the large number of integrated stack technologies to design, deploy and manage IoT applications. The oneM2M common service layer offers a set of commonly needed service functions for IoT, e.g., device management, registration, and security. It horizontally joins the middle layers of several separate, heterogenous, vertical IoT solutions. The sharing of common capabilities at this middle layer ensures re-usability and delivers economies of scale.

Another aspect of oneM2M's horizontal architecture is that it lays the foundations for cross vertical interoperability. Individual IoT solutions using oneM2M can share data and resources through common service layer functions such as resource discovery and semantic interoperability. As a result, application developers, solution providers and data suppliers can share data between applications that span multiple verticals and reduce their dependence on single-vendor solutions. oneM2M follows a use-case-driven approach to IoT standardization with real-world scenarios that serve as a basis to derive requirements for the common service layer. By deriving requirements in this manner, oneM2M can address the needs of multiple vertical domains including the needs that the developers did not anticipate at the time of their implementation, thus building an element of future proofing in their standardization process. To enhance the stability of the oneM2M specifications and shorten time to market, oneM2M hosts multiple interoperability and hackathon events every year. Engineers can test their products against each other, in accordance with oneM2M-defined test specifications, and participants can learn more about oneM2M. These events are a significant asset in validating oneM2M's technical specifications and their

[1] ARIB - Association of Radio Industries and Businesses - www.arib.or.jp/english/arib/about_arib.html
ATIS - Alliance for Telecommunications Industry Solutions - www.atis.org
CCSA - China Communications Standards Association - www.ccsa.org.cn/english/
ETSI - European Telecommunications Standards Institute - www.etsi.org/
TIA - Telecommunications Industry Association - www.tiaonline.org/
TSDSI - Telecommunications Standards Development Society, India - www.tsdsi.in/
TTA - Telecommunications Technology Association - www.tta.or.kr/English/
TTC - Telecommunication Technology Committee - https://www.ttc.or.jp/e/index.html

[2] oneM2M latest specification drafts http://onem2m.org/technical/published-drafts

[3] ITU-T Publication Y.4500.1 : oneM2M Functional Architecture - https://www.itu.int/rec/T-REC-Y.4500.1/en

implementation by multiple solution providers. In addition, the Global Certification Forum (GCF) manages a oneM2M global certification program[4]. oneM2M certification through GCF ensures oneM2M solution providers maintain the proper functionality and compliance with the oneM2M specifications and that oneM2M solutions can interoperate with one another. The full suite of oneM2M technical specifications, including different releases, is available for download at www.oneM2M.org.

2 Use cases for edge IoT scenarios

The converged technology of IoT and edge computing is emerging as a new paradigm. Edge computing can effectively and quickly process IoT services that require real-time processing, such as self-driving cars and augmented reality. Additionally, it is expected to reduce the overload on the IoT cloud platform, address the problem of limited communication resources, and reduce the risks from cyber-attacks. In the future, the Internet of Things and edge computing are expected to effectively provide the data and processing needed for technologies such as artificial intelligence, blockchain, metaverse, and digital twins; all of which are application technologies that comprise components of the fourth industrial revolution. In this chapter, we explore various use cases for IoT and edge computing. Table 2 shows a selection of the use cases related to edge computing developed in oneM2M. As seen in the table, edge computing is being used in various fields such as smart factories, self-driving cars, and augmented reality.

Table 2: Example use cases using IoT and Edge Computing together.

Use Case Title	Description
Accident Notification Service using Edge Computing [27]	An edge computing based IoT architecture is used to lower the processing burden on the IoT cloud nodes for a traffic monitoring service based on data collected by vehicular on-board cameras and surveillance cameras.
Smart Transportation with Edge Computing [see ref. 27]	Edge computing technologies are very suitable for enabling smart transportation because they enable a dynamic and localized environment for deploying services potentially closer to the users and/or data sources.
High-precision Road Map Service using Edge Computing [see ref. 27]	This use case introduces a High-precision Road Map Service based on data collected by vehicular on-board cameras/sensors, surveillance cameras (e.g., video camera, radar, LIDAR, GPS), and V2X data from mobile core network. Vehicular on-board cameras/sensors and surveillance cameras collect surrounding data periodically and send them to local Edge Node. Edge Node cloud processes the data, generates differential data of existing High- precision Road Map, and provide the data to vehicles in real time.
Link Binding Management for Digital Twins and Edge Computing [see ref. 29]	Digital twins of a physical product is created in cyber domain so that the status of the physical product in each stage of its lifecycle can be monitored and managed remotely. To support this scenario, a link binding between physical products in a physical domain and their digital twins or digital companions in cyber domain is needed. In this case, edge computing can be used to manage such physical IoT products.

[4] GCF and TTA officially sign agreement for oneM2M global certification solution at MWC19 https://www.globalcertificationforum.org/news/gcf-tta-sign-onem2m--agreement-at-mwc19.html

Smart Factories using Edge Computing [see ref. 30]	In factories, a lot of data are created from Programmable Logic Controllers (PLCs) every second, and data are utilized to monitor production lines. This data is available via industrial bus systems, e.g., Real-time Ethernet. In order to monitor remotely, data is gathered by the IoT/M2M service platform that needs to interface with such industrial bus systems via Edge Nodes (i.e., IoT gateways). In such cases, only necessary data is gathered depending on situations and filtering / pre-processing of the raw data needs to be performed at the Edge nodes.
Vulnerable Road User Discovery Use Case for Edge Computing [27]	The Vulnerable Road User (VRU) detection service is a vehicle domain service to detect pedestrians and cyclists on a road. The VRUs application uses accurate positioning information provided by various traffic participants. The information used for VRU detection services has to be shared by VRUs. The VRUs make their presence/location known through their mobile devices (e.g., smartphone, tablets), along with vehicle's use of that information. In this use case, Edge nodes can lower the processing burden from Cloud IoT platform via resource and processing offloading.

2.1 Smart Factory of the Future

The Smart Factory of the Future envisions increased application of advanced AI/ML techniques such as Federated Learning (FL) and Deep Learning (DL), to process the massive multi-modal factory data that will be generated. This is an application where the combined usage of MEC and oneM2M has unique advantages. Let's consider an example. A production line in a smart factory may include a wide variety of machines that may be fixed with wired connectivity or mobile (such as, collaborative robots, robotic arms, monitoring equipment, etc.). These devices will generate massive amounts of data that is impractical to transfer to the Cloud. Additionally, factory operations like real-time defect monitoring and intervention to pause or alter the production by IoT machines requires data processing, sensing inference, and subsequent actions with fast response times. Application of Federated Learning (FL), coupled with Deep Learning, offers excellent potential to address this need, especially when supported by standardized frameworks like oneM2M and MEC. FL agents, deployed on edge and IoT devices in the factory, locally process training (i.e., multi-model factory sensor information), including preserving privacy, instead of transporting all data out of the factory. The factory FL agents share their local model updates with intermediate and global model aggregators to an edge cloud or distant cloud. After the training converges, the model can be used for inference, utilizes a fractional deployment approach. This technique leverages the unique structure of Deep Neural Networks (DNN). For example, lower layers in DL DNN are computed in the factory on the device edge or in IoT devices, while higher layers are computed on edge compute resources. Augmenting DL with FL allows more accurate detection and fast convergence due to decreased number of communication rounds, attributed to computation performed on participating devices in the factory. This approach may achieve_detection and response time less than few milliseconds with greater geospatial specificity.

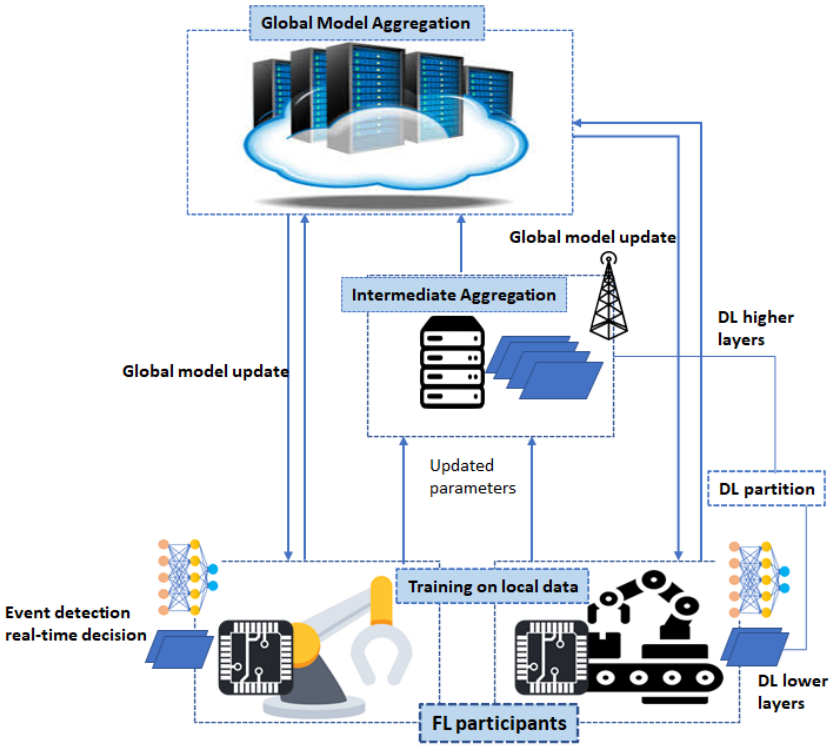

Figure 2: Smart Factory of the Future – Federated Learning Scenario

2.2 Automotive scenarios

Many automotive use cases can be associated to joint deployments of MEC and oneM2M. In fact, these heterogeneous environments comprise a number of network elements and connected devices (cars, bicycles, pedestrian users, ...) and related infrastructure elements (Road-Side Units, sensors, cameras). In the following sections a couple of MEC scenarios relevant in the IoT context are described.

2.2.1 Data transfer optimization using location/QoS information

This scenario introduces vehicular data transfer optimization of 3GPP services based on providing location information and QoS information to Edge Nodes. This enables adjustment of data transmission for individual 3GPP devices based on congestion levels of each location.

Figure 3 illustrates the scenario of vehicular data transfer optimization. Vehicles transfer their collected data to a local Edge Node based on the data type, the time period and/or the traffic volume allocated by the Edge Node. To optimize the data transfer the Edge Node collects the list of UEs active in each area and analyses the congestion level in time series.

In this scenario, vehicular data is categorized as road map data and in-vehicle data, used by a road map service provider and a vehicular service provider respectively. The road map data is collected by vehicle on-board cameras/sensors (e.g., video camera, radar, LIDAR, GPS). Those cameras/sensors collect data on the surroundings of the vehicle periodically and send the data to a local Edge Node. However, the collected data can cause huge traffic volumes and might not require low-latency communication. Thus, the road map data indicates a low-priority category for the data transfer. On the other hand, in-vehicle data contains vehicular state (e.g., fuel state, battery charging alert, warning of oil pressure, current mileage count) and

might require low-latency communications. Therefore, the in-vehicle data indicates a high-priority category for data transfer.

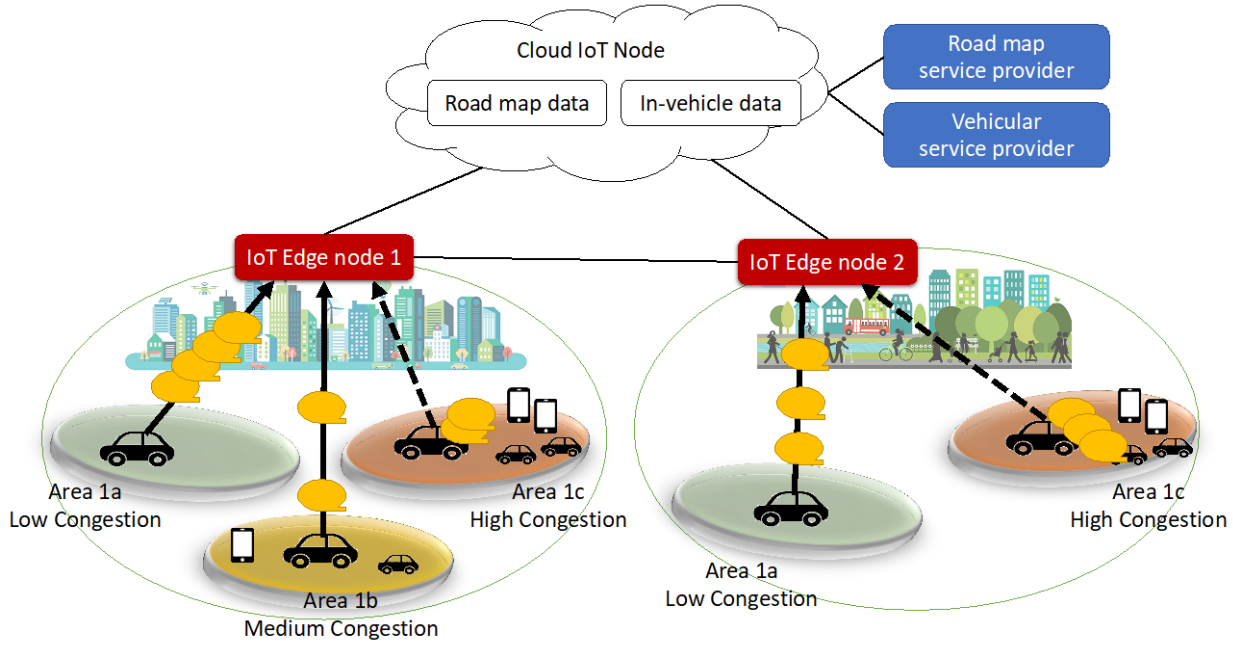

Figure 3: Scenario of vehicular data transfer optimization

This scenario intends to mitigate the burdens on a Cloud Node and optimize data transfer by moving processing to the Edge Node and leveraging the capabilities of a 3GPP Core Network. As a result, a Network/System operator can offer value-added services to a Service provider.

2.2.2 Resource and Task offloading from IoT Cloud to Edge Nodes

This scenario introduces a case where Cloud IoT server needs to delegate tasks and resources to edge computing capabilities. Figure 4 shows the scenario of resource offloading to Edge Nodes. The Vulnerable Road User (VRU) detection service is a vehicle domain service to detect pedestrians and cyclists on a road. The VRUs application uses accurate positioning information provided by various traffic participants. The information used for VRU detection services must be shared by VRUs. The VRUs make their presence/location known through their mobile devices (e.g., smartphone, tablets), along with a vehicle's use of that information. In this example, when the driver of a Host Vehicle (HV) intends to make a left turn, if there is a vulnerable user or cyclist passing the place where HV is going to pass, HV is alerted to the presence of a VRU in a safety and/or awareness message.

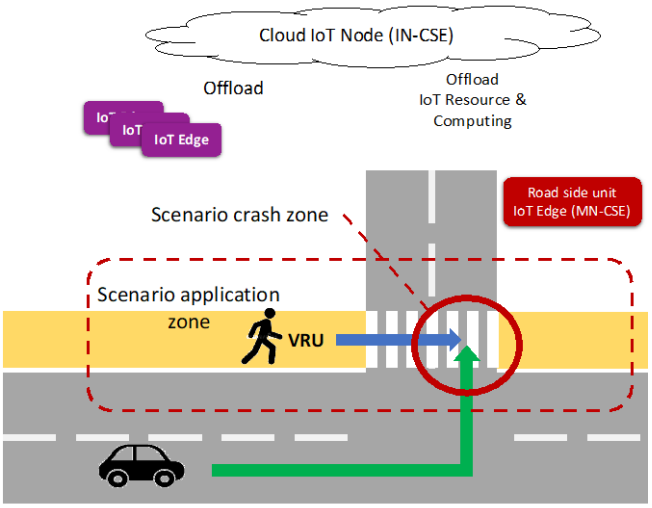

Figure 4: Scenario of resource and tasks offloading via Edge Nodes (Source oneM2M [27])

In this scenario, we can assume a HV is registered to an offloading service. Then when the HV enters a scenario application zone, that an Edge Node (i.e., MN-CSE in oneM2M) is covering, offloading procedures are performed to move the resources and tasks associated with the HV (in this case, VRU detection service and its managing resources) to the MN-CSE. The IoT Cloud (i.e., IN-CSE in oneM2M) send an indication to MN-CSE to perform resource offloading. MN-CSE then retrieves all the relevant resources and services. As the MN-CSE is the nearest node to the HV, it can immediately send a warning notification to HV as soon as it detects the VRU on the road.

An offloading procedure, relocating tasks and resources to a node close to users can be applied to this VRU detection service. In this case, the service can be provided to users with fast response times.

Figure 5 shows a high-level concept of offloading in oneM2M. Offloading concept in oneM2M allows an IN-CSE, a centralized IoT server platform, to transfer relevant resources and tasks to a target Edge MN-CSE node. Then the MN-CSE node can directly support the nearby IoT end devices with the offloaded service.

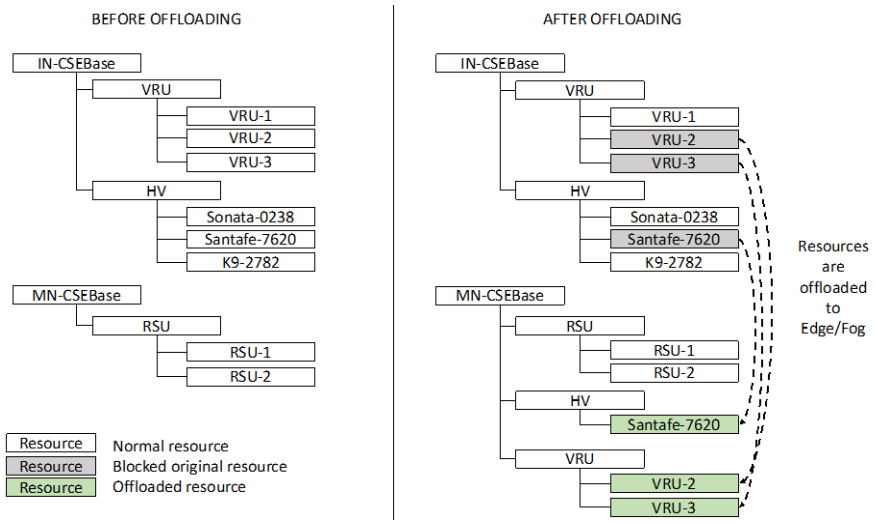

Figure 5: A high-level concept of Edge offloading in oneM2M (Source: oneM2M [27])

2.3 Deploying Edge with Constrained Devices

A key emerging use-case for edge IoT is deployment of constrained devices in MEC systems, offering edge capabilities (currently being studied in ETSI GR MEC 036). To illustrate this concept, the following figure depicts constrained or mobile edge devices in the overall edge/cloud system. The computing layer is composed of different computing tiers, namely the central cloud, the edge cloud (e.g., Telco Edge), and wireless or mobile Far Edge associated with the constrained devices (e.g., UEs, CPEs, and edge IoT devices). Constrained edge devices may be battery-powered, mobile, volatile, and have limited compute and connectivity as compared to the traditional edge clouds. The constrained edge devices may also collaborate and exchange information among themselves.

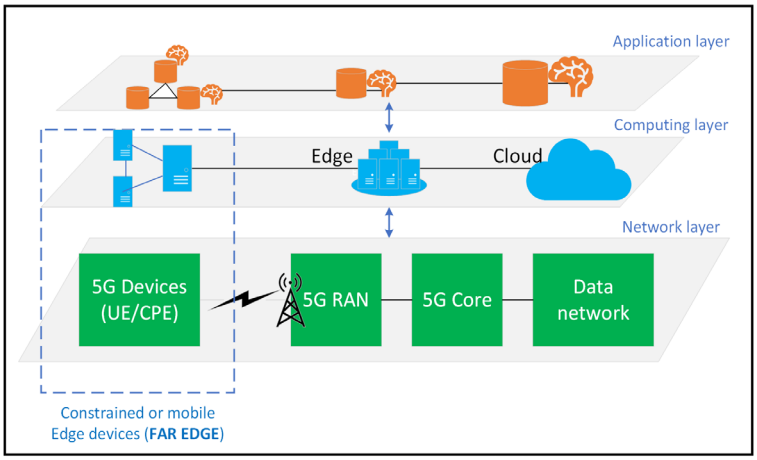

Figure 6: Constrained devices in MEC systems

In this context, edge applications can run at the *far edge,* e.g., on constrained devices, which are characterized by:

- Limited computing capabilities (note: as a particular case, also the "absence of limitations" can fall into this definition, thus in principle enabling oneM2M deployments not only in far edge, but also in edge cloud or central cloud)
- Temporary absence of connectivity with MEC management and user plane
- Absence/ non-availability of some MEC service APIs in proximity to the constrained device

To deploy edge on constrained devices, many technical aspects are still currently under discussion in ETSI MEC, as relevant in terms of implications for the presence of constrained devices, especially at the far edge. For this White Paper, it is enough to say that general applications and functions may be hosted anywhere in the computing stratum (cloud, edge or far edge devices).

3 How MEC and oneM2M can sit together

As both ETSI ISG MEC and oneM2M are leading global standards for edge computing and IoT respectively, this White Paper intends to demonstrate how the two architectures can be deployed to enable an efficient and interoperable deployment of common IoT services in edge computing environments. To do so we first describe in detail the two architectures, ETSI ISG MEC and oneM2M. Then we illustrate their technical approach to edge computing, demonstrating how both organizations' thought leaders envision the growth of IoT on the edge through collaboration. Thirdly, we provide the design of the interworking architecture that enables dynamic instantiation of oneM2M service layers on edge nodes with only the required oneM2M common service layer functions (CSFs) by using the ETSI ISG MEC virtualization infrastructure; and subsequently grant IoT devices access to required common service functions with low network latency.

3.1 MEC architecture

The ETSI MEC system (Figure 7 below) consists of the MEC hosts and the necessary MEC management functionality to run MEC applications within an operator network or a subset of an operator network. According to ETSI GS MEC 003 [1]:

- 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 functionalities 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.

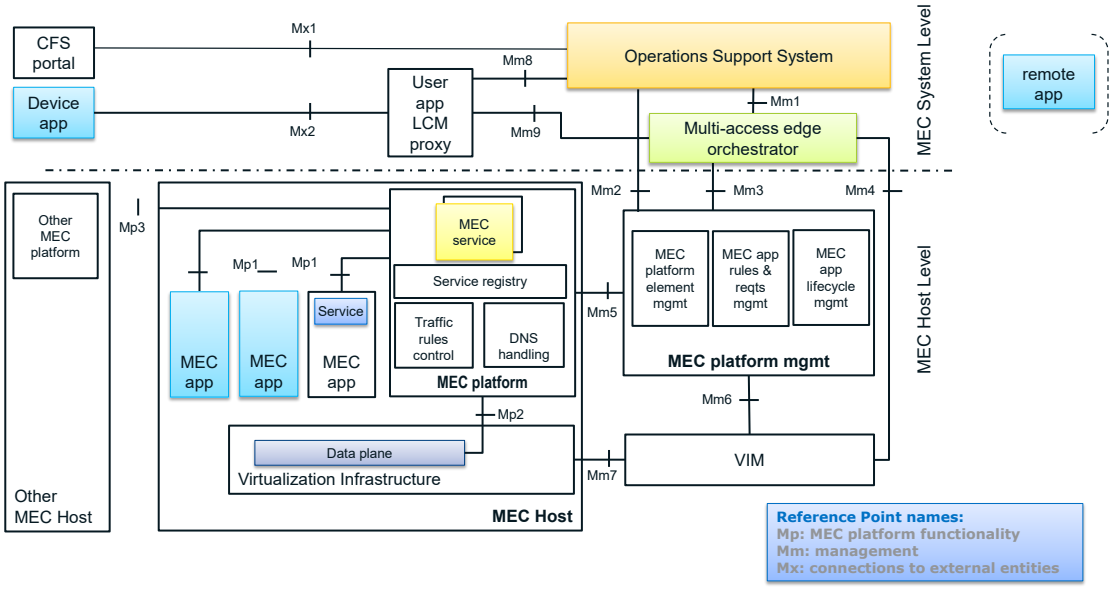

Figure 7: MEC reference architecture (Source ETSI GS MEC 003 [1])

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

- The MEC system level management includes the MEC 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.

A further MEC architectural variant (MEC in NFV) is also specified in GS MEC 003 [1], explaining how MEC can be deployed in virtualized environments. However, for the sake of simplicity, this variant is not shown here. In general, the concept of virtualization has been introduced recently in IT, and a huge percentage of workloads is running on the virtual machines. The next step for virtualization is envisioned in networking. The concept of Virtual Network functions (VNFs) improves the way telecoms providers, create, deploy, and manage networks. Therefore, ETSI ISG MEC provides the virtual function orchestration manager responsible of orchestrating oneM2M service layer functions which are presented in form of VNFs in order to be instantiated on top of ETSI ISG MEC. In addition, virtualization would allow the design of a novel interworking architecture that allows the provisioning of virtual IoT platform to run at the network edge nodes to meet requirements for mission critical IoT applications.

In this architecture, MEC applications can both consume services available in the MEC system and also produce services, which are made available by the MEC platform to other applications. This Application Enablement Framework is an essential functionality of the MEC platform (see GS MEC 011 [3]).

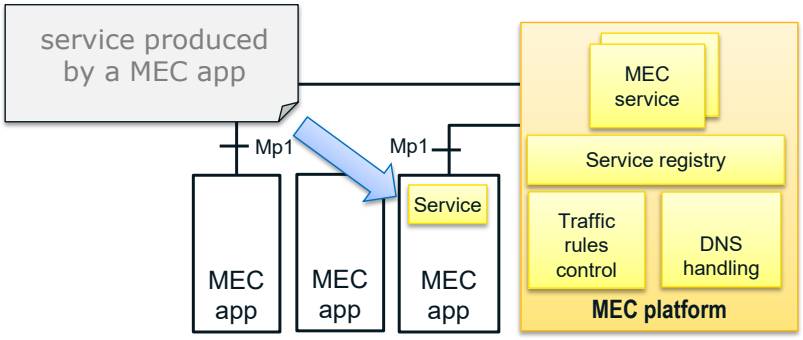

Figure 8: A MEC application instance may consume and/ or produce MEC services

The above figure is thus showing two kinds of services offered by the MEC system, i.e., services produced by MEC apps, and standard MEC services (depicted in the MEC platform box) and offered to SW developers through standardized APIs. In the context of MEC systems (and for the purpose of the present paper) it is worth remembering an important definition for these service producers in MEC:

***Service producing MEC application**: a MEC application producing a service for other MEC applications*

A MEC application is instantiated and run as a virtualized software application, within for instance a Virtual Machine (VM) or OS containers provided as part of the application package as a software image(s), on top of the Virtualization infrastructure of the MEC system. It can also potentially interact with the MEC platform to consume and provide MEC services. A MEC service is a service provided and consumed either by the MEC platform or a MEC application. When provided by an application, it can be registered in the list of services

to the MEC platform over the Mp1 reference point. A MEC application can subscribe to a service for which it is authorized over the Mp1 reference point.

3.1.1 MEC in 5G systems

In Release-17, 3GPP has specified the Application Layer Architecture and functional entities to support application deployment at the edge (3GPP TS-23558 [31]). Figure 9 represents the EDGEAPP architecture described in Release-17.

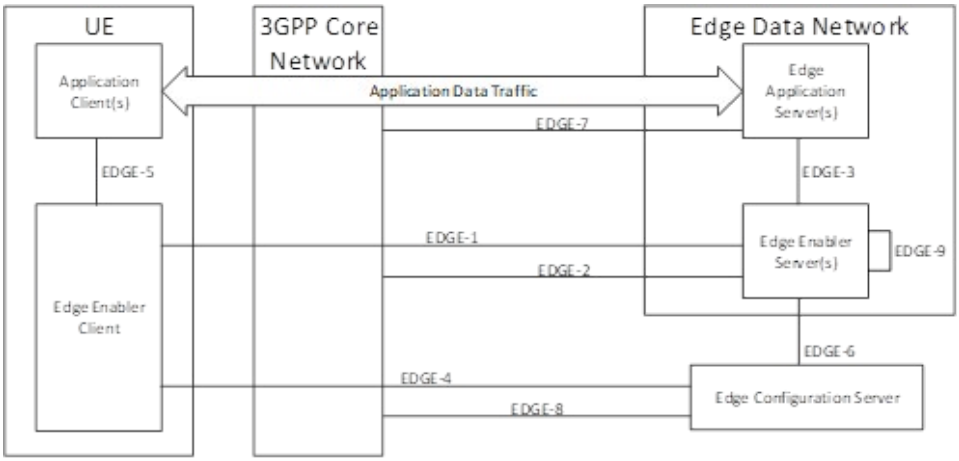

Figure 9: EDGEAPP Architecture (ref. 3GPP TS-23558 [])

In Release-17 ({3GPP TS-23558 [31]), 3GPP SA6 has also provided the relationship of ETSI ISG MEC architecture with EDGEAPP architecture (Figure 10) (see also ETSI GS MEC 033 [1]), where a similar analysis is conducted). This synergized architecture indicates that both EAS and MEC App are application servers and can provide similar application specific functionalities. Similarly, both EES and MEC platform provide application support capabilities towards the application servers.

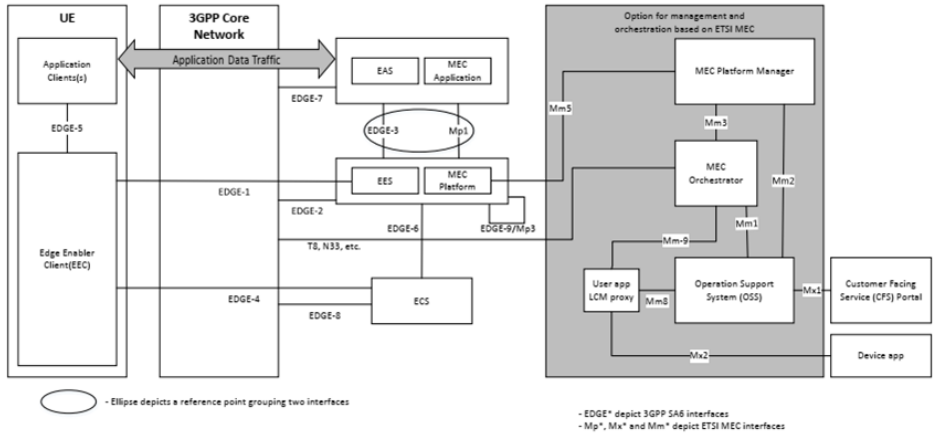

Figure 10: Relationship between EDGEAPP and ETSI MEC architectures (ref. 3GPP TS-23558 [31])

Currently, 3GPP SA6 is working on the enhanced EDGEAPP architecture (3GPP TR-23700 [32]). The major features of these enhancements are to support application provisioning and relocation across Cloud and Edge and to support Federation across multiple operators.

It is also considering the alignment of interfaces and APIs between the application servers (EAS and MEC App) and the edge platforms (EES and MEC Platforms). The CAPIF based deployment has also been proposed to expose EAS/EES service APIs to the MEC Platform and invoke MEC Services from EAS.

3.2 oneM2M architecture

The oneM2M service layer is a general-purpose standard that applies to all industry verticals. It brings together all components in the IoT solution stack and specifies a distributed software/middleware layer, sitting between applications and underlying communication networking HW/SW. oneM2M service layer can be integrated into devices, gateways, and servers. Figure 11 below shows a oneM2M simplified architecture [25]:

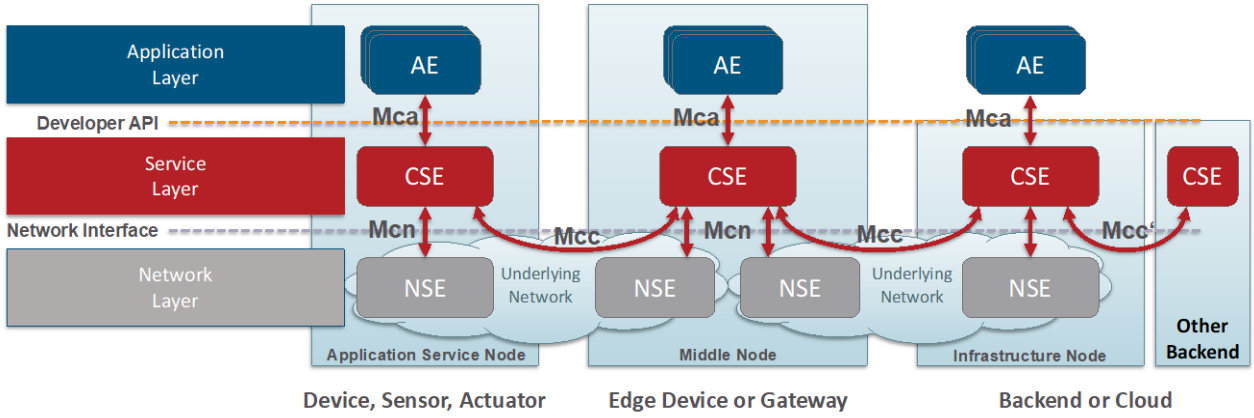

Figure 11: oneM2M simplified architecture

This architecture is composed of the following entities:

- **Reference Points**: One or more interfaces - Mca, Mcn, Mcc and Mcc' (between 2 service providers)
- **Common Services Entity**: Provides the set of "service functions" that are common to the IoT domains
- **Application Entity**: Provides application logic for the end-to-end IoT solutions
- **Network Services Entity**: Provides services to the CSEs beyond the data transport
- **Nodes**: Logical equivalent of a physical (or possibly virtualized) device

In particular, the above architecture includes different kinds of nodes, which are named based on the specific instantiation option considered:

- Application Service Node (ASN) at the Edge device / Gateway (GW)
- Middle Node (MN) at an Edge Node / Gateway (GW)
- Infrastructure Node (IN) at Cloud / company servers
- Infrastructure Node (IN) at Other Cloud / companies servers

As a note, in principle, the oneM2M architecture supports another type of node, the so-called “Non-oneM2M Node” (NoDN). This node type is not shown in the figure above. oneM2M specifications also defines a non-oneM2M Node which is any nodes that does not contain native oneM2M Entities (neither AEs nor CSEs). Typically, such nodes would host some non-oneM2M IoT implementations or legacy technology which can be connected to the oneM2M system via interworking proxies or agents.

3.2.1 Reference architecture and Common IoT Service Layer

Many IoT applications are deployed as “silos” in a vertical solution stack. At its simplest, this involves one application (e.g., asset tracking, condition monitoring, inventory tracking logic) using one communications network to interact with connected devices or sensors. This arrangement does not lend itself to operational scaling or resource reuse. Consider an IoT application that requires a device management capability as an example. If the device management function is implemented for a narrowly defined use case this could easily prevent its reuse for a second or third IoT application. The same logic applies to other service enablers necessary for the deployment and management of IoT applications. To solve the problem, the oneM2M architecture applies a horizontal model based on a common services framework. Examples of common services include communications management, device management and security functions. This architecture also ensures that devices and their data are both discoverable and accessible to more than a single parent application. Applications can be built using oneM2M-capable devices sourced from multiple suppliers, reducing the risk of vendor lock-in. This allows solution providers to build only once and reuse many times. This is a significant advantage when a lack of standardization inhibits permutations across multiple technology vendors, service providers, organizational boundaries and IoT applications.

In addition to standardizing the common services layer, oneM2M includes specifications for end device and gateway entities. This makes it possible to deploy native-oneM2M solutions, comprising oneM2M compliant end-devices communicating with one or more oneM2M platforms. It is also possible to support deployments that contain a mix of oneM2M and proprietary devices. This requires an interworking proxy gateway to manage nononeM2M devices communicating with a oneM2M platform. The oneM2M standards define a horizontal common services layer IoT platform that allows applications within a domain (e.g., a city, factory, or transportation hub) to communicate effectively, reliably, and securely. The standard supports a federated model of operation so that these benefits accrue to applications from various previously “siloed” domains (e.g., to manage transportation and environmental sensing on road networks or, utilities and wellness in offices and households) as shown in Figure 12.

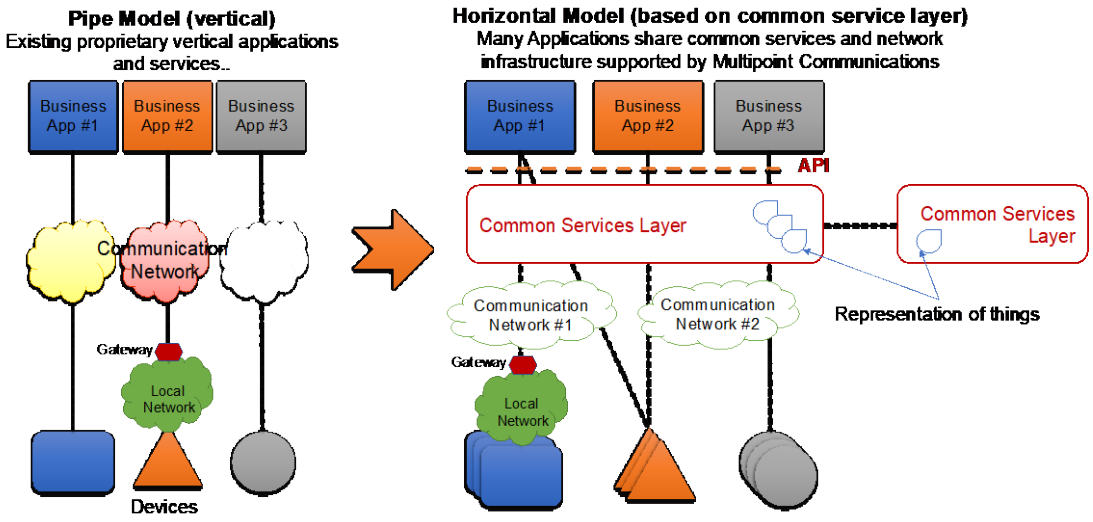

Figure 12: Overview of oneM2M Horizontal Architecture

From a functional perspective, oneM2M has defined fourteen common service functions (CSFs) as shown in Figure 12. These relate to network connectivity, device security, transport protocols, content serialization, IoT device services and management and IoT semantic ontologies.

3.2.2 oneM2M Common Service Layer Functions

Each of these oneM2M services are defined so that application developers can focus on application-specific functionality (e.g., turning a switch on or off), while relying on abstractions provided by oneM2M to mask the underlying technology-specific details, thus allowing bindings to different communications stacks and protocols such as HTTP, CoAP and MQTT. For example, a simple switch might use a fixed or Wi-Fi network, a CoAP or HTTP transport. It might use a JSON or XML serialization, an Open Connectivity Foundation (OCF) or thread service enablement, or an ontology based on Smart Appliances REFerence (SAREF) or W3C's Thing Description. For further information about oneM2M ontology, the reader is referred to the oneM2M Gitlab page[5].

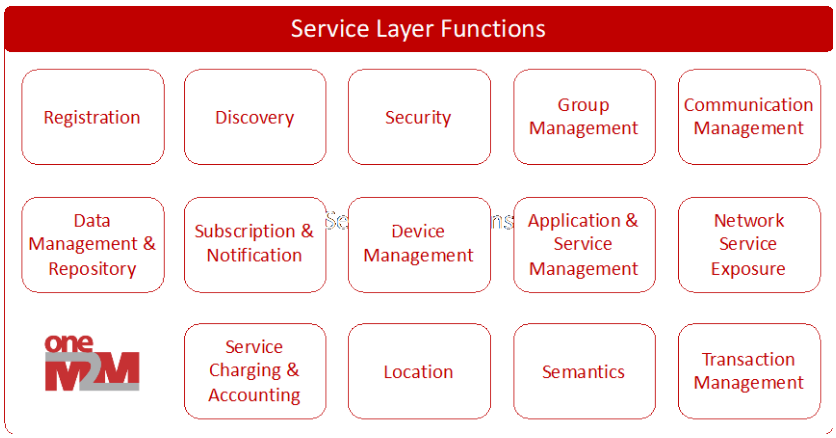

Figure 13: oneM2M Common Service Layer Functions (CSFs)

[5] https://git.onem2m.org/MAS/BaseOntology

Finer-grained capabilities underpin each service function as illustrated in the case of device management, security and application and service management functions. In keeping with a philosophy of leveraging existing standards rather than re-inventing them, oneM2M complements existing and proven security technologies to address IoT security challenges. It provides a common set of security capabilities to secure IoT devices and data; and to prevent or mitigate attacks. This is made possible by an abstracted set of security-related APIs to simplify security for devices and applications. oneM2M is a constantly evolving standard with a strategic roadmap designed to address new IoT requirements. This is made possible by the common service layer concept, which was conceived to accommodate new service functions. oneM2M works according to a release cycle to standardize new service functions. It emulates the cellular industry's 3GPP standardization model, to address new requirements and evolving technologies through progressive releases.

3.2.3 Virtualization of oneM2M Common Service Layer functions

Network Virtualization is a unique opportunity, enabling IT service providers to develop an end-to-end digital service based on network as a service (NaaS). Network Function Virtualization (NFV) targeted the oneM2M common service layer functions where the virtualization technologies are used to replace hardware servers hosting a common set of IoT functions with VNFs that can run as software on virtual machine. On top of appropriate virtualization infrastructure, VNFs of oneM2M service functions can be deployed anywhere in the network on-demand. NFV enables to slice an oneM2M common service layer in the form of multiple virtual IoT CSFs at the network edge. To achieve this, virtualization technologies allow the transformation of IoT CSFs from the common service layer to virtual IoT CSFs like software images. These virtual IoT CSFs include resources and service functions that have attributes specifically designed to meet the needs for IoT vertical markets such as smart building, industrial IoT, smart city, and, subsequently, create a network-as-a-service model for oneM2M edge computing.

3.2.4 oneM2M Edge Computing

Edge computing is expected to be employed to mitigate the burdens on data centres/core networks and decrease communication latency by processing, acquiring and storing data at the edge network near IoT devices. These technologies also enable reducing communication costs, enhancing reliability, and providing localized contents efficiently. Therefore, oneM2M has started a new work supporting Edge Computing using oneM2M standardized technology. To support this new feature, oneM2M started to develop a technical report TR-0052 – "Study on Edge and Fog Computing in oneM2M systems" [27]. This Work Item analyses use cases and requirements for edge computing and identifies related advanced features to be supported in oneM2M. More specifically, based on existing technologies and standards, issues of interest in the IoT/M2M domain to be addressed include:

- computing, storage, communication, and analytics at the Edge Nodes
- communication between Edge Nodes, and between Edge Nodes and Cloud Nodes.
- service provisioning, migration, and service-aware routing between Edge Nodes,
- service orchestration and data synchronization between Edge Nodes
- system reliability and node redundancy enabled by node pooling
- management of the Edge Nodes (including softwarization of oneM2M functions and services)

To determine oneM2M System enhancements for edge computing, service and feature use cases relying on edge technology are identified. This White Paper shows two identified use cases from oneM2M TR-0052 in subclauses 2.2.1 and 2.2.2.

3.3 Synergized MEC-oneM2M architecture

As explained in subclause 3.1, MEC applications can both consume services available in the MEC system and also produce services, which are made available by the MEC platform to other applications.

In the context of Internet of Things (IoT) and Machine-to-Machine (M2M) communications, and by considering the oneM2M simplified architecture depicted earlier in Figure 12, we can consider the following mapping of oneM2M entities with ETSI MEC entities:

- **A common service function** (e.g., Network Service Function) **or CSE** in oneM2M architecture can be represented as a **MEC Service** and/or as a service producing MEC App instance. This service would be exposed by the MEC platform to be connected to (authorized) consumer Application Entities (AE).
- Similarly, **AE** in oneM2M architecture can be seen as a **MEC App instance** by ETSI MEC system.

So, in a nutshell, architectural interworking between ETSI MEC and oneM2M is made possible by seeing the CSE and AE functional elements of oneM2M as particular instances of MEC services and applications from the point of ETSI MEC system (which is not imposing any mechanisms on how these services and applications should be designed). More specific hooks can be envisaged to effectively integrate these two systems and gain more benefits from tighter integration (for example, specific mechanisms for application onboarding and instantiation, or additional MEC service APIs, etc.). However, from an architectural point of view, the two standards are already quite compatible, allowing joint deployments of oneM2M nodes in MEC systems, where further technical details can be left to implementation.

A simplified view is provided in Figure 14, where essentially the oneM2M Edge Instance can be seen as an application by the MEC platform, which can also facilitate the connection with the IoT service Platform (see clause 4.1 for more technical details).

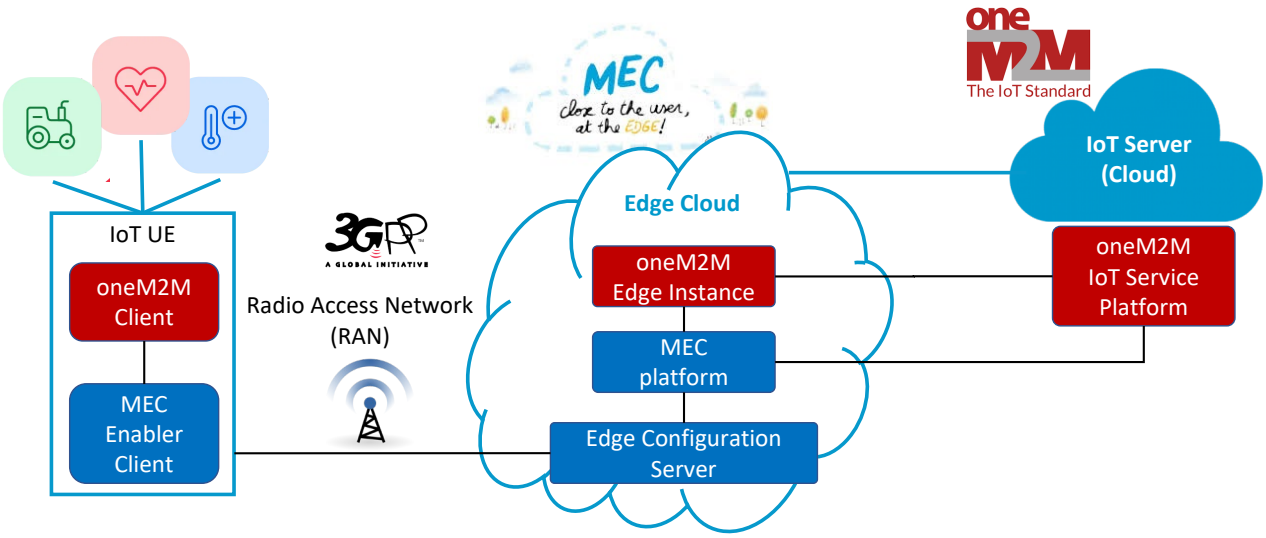

Figure 14: High-level conceptual architecture with MEC and oneM2M

4 Deployment considerations

In this clause the discussion is based on the deployment being edge computing compliant (thus, with an oneM2M and MEC at the centre of its architecture, as depicted in the figure below). Edge computing can consider several deployment options for this integration, each one having different technical and business impacts. Figure 15 details these modes of integration:

Option A: deploy the oneM2M as a cloud, MEC as an edge

- This deployment presents the IoT platform itself on the cloud side.
- This is one of the most common deployment scenarios that use cloud based IoT platforms and edge computing.
- Although some of the benefits of edge computing can be achieved through network and processing, the advantages of 100% edge computing are not fully utilized because the cloud remains the final endpoint where data is stored and managed.

Option B: oneM2M and MEC as an edge with the different physical node

- This option presents oneM2M and MEC deployed as edge nodes but located on different physical edge nodes.
- Compared to Option A, both oneM2M and MEC exist on the same edge node, allowing for all data and information exchange to be performed locally, resulting in faster processing.
- Although oneM2M IoT service providers and ETSI MEC entities are different, this scenario can still be used in the early edge computing market.

Option C: oneM2M and MEC in the same physical edge node

- In this scenario, oneM2M and MEC platforms are installed and operated on the same physical edge node, which can significantly improve service by eliminating unnecessary data and information exchange.
- A Service Level Agreement between the platform providers is required, and both platforms support dynamic deployment to various edge nodes.

Option D: oneM2M and MEC are tightly coupled in the same edge node

- This scenario involves oneM2M and MEC platforms being physically coupled through APIs, allowing for mutual interworking.
- The oneM2M platform is recognized as an MEC application, utilizing all functions provided by MEC to provide a 100% edge computing environment.
- The MEC platform can directly provide data source, processing, and multi-access networking by hosting oneM2M as an application.
- A standard document providing interoperability and interworking between the two platforms, oneM2M and ETSI MEC, is required.

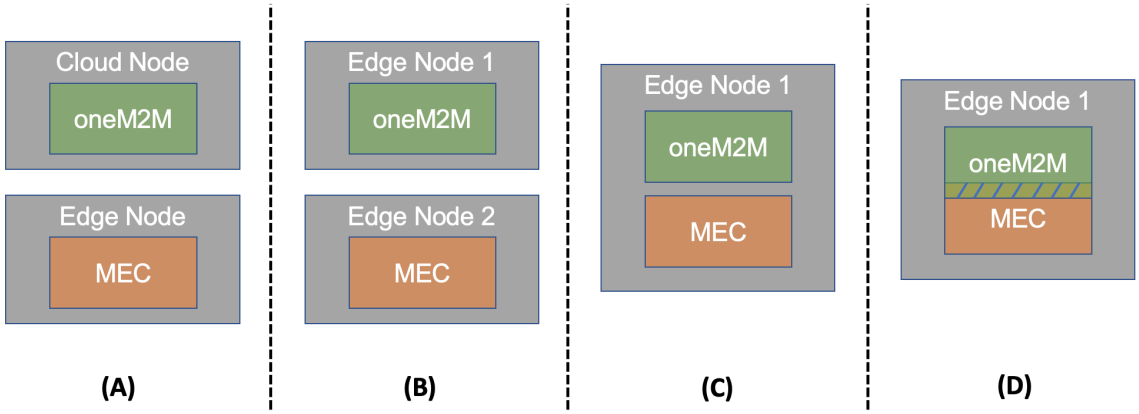

Figure 15: Integrating oneM2M and MEC with edge computing

More in detail, according to the architectural mapping described in section 3, Figure 16 below shows three possible instantiations of CSE and AE in MEC:

a) A new MEC interworking CSF is introduced to oneM2M CSE so that the CSE can behave as a single MEC App instance. An AE connected to this CSE can support IoT services via the MEC infrastructure.

b) oneM2M CSE is a service offered by the MEC platform, consumed by an AE, which is an MEC app. In this case, Mp1 interface in MEC should be enhanced to support oneM2M Mca Interface to allow communication between AE and oneM2M CSE.

c) oneM2M CSE is implemented as a service producing MEC App instance, consumed by the AE, which is implemented as a different MEC App instance. In this case, the AE should support both Mp1 and Mca interfaces to communication with MEC platform and oneM2M CSE, respectively.

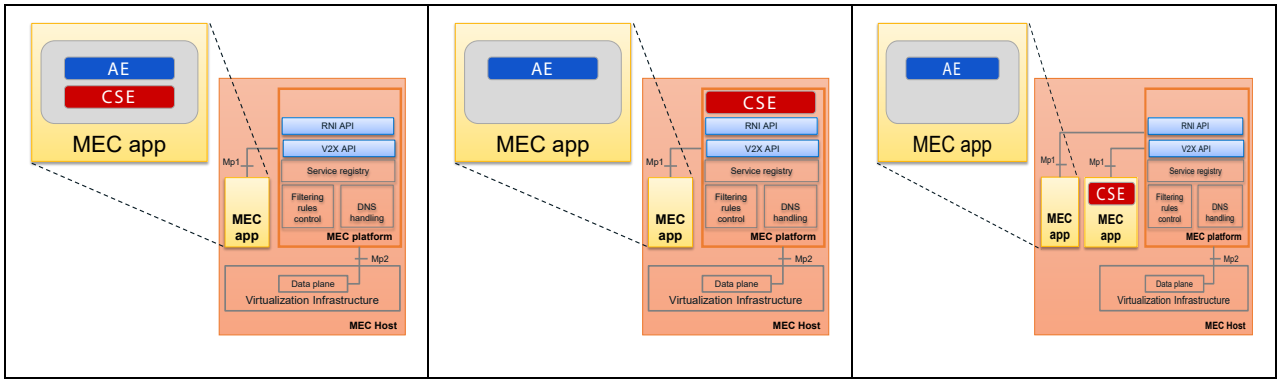

Figure 16: deployment options of CSE and AE in MEC systems:
(left) both AE and CSE as a single MEC App instance; (centre) CSE as a service in the MEC platform; (right) CSE implemented as a service producing MEC App instance (CSFs)

4.1 MEC IoT API

ETSI ISG MEC has produced a set of APIs that can be used and consumed at application level. In the context of Internet-of-Things, the group has recently published a ETSI GS MEC 033 [21], by specifying an IoT API to assist the deployment and usage of devices that require additional support in a MEC environment, e.g., due to security constraints, limited power, compute, and communication capabilities, such as IoT and MTC devices. More in detail, the introduced MEC IoT Service (IoTS) provides means to incorporate heterogeneous IoT platforms in the MEC system and exposes IoT APIs to enable the configuration of the various components of the overall IoT system.

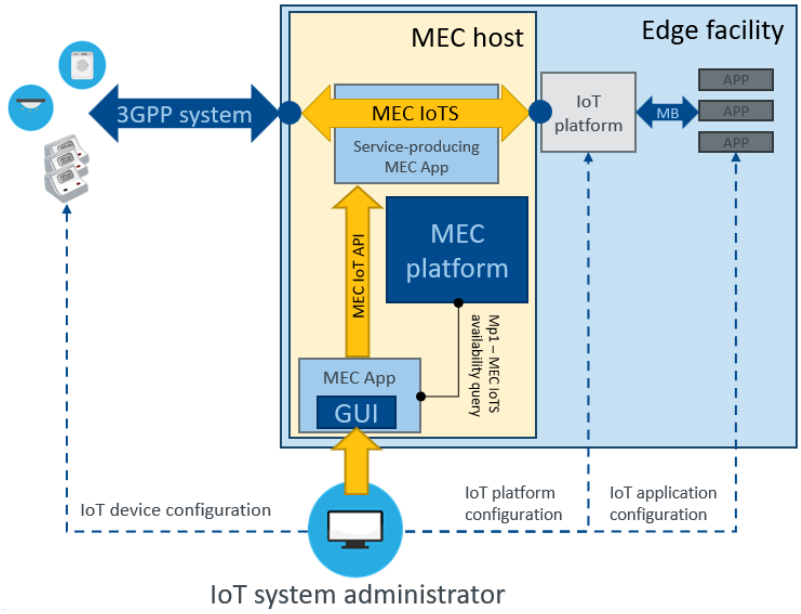

Figure 17: Usage example of IoT API (ref. GC MEC-033 [21])

IoT devices, in particular ones with limited capabilities such as, for example, the inability to store a client certificate, may be served by applications in an IoT Platform with which the devices can authenticate using client certificates, and exchange information in the form of messages with the IoT Platform acting as a topic-based message broker.

The devices can subscribe to receive messages for specific topics and can post messages on any topic to have the message forwarded by the IoT Platform to any application subscribed to that topic.

The MEC IoT Service enables an IoT device to communicate with the applications in the IoT Platforms without having to store client certificates or know the specifics of topic-based messaging. It allows for three different types of messages: Event, Uplink and Downlink messages. Event messages carry information related to the session between the device and the application. Uplink messages carry information from the device to the application. Downlink messages carry information from the application to the device.

The MEC IoT Service is configured through the MEC IoT API (see GS MEC 033 [21] it exposes. In particular, the API provides the means to manage Device and IoT Platform information registrations.

Device information registers for each device any number of device identities, including network-specific identities such as, for example, IMSI, MSISDN, IMEI in a 4G network, device authentication information, which may include a client certificate, metadata, and formats of the event and uplink messages. Message formats provide an indication of the topic and the identities and metadata to include in the message.

IoT Platform information registers for each IoT Platform the topics it supports. This will enable the MEC IoT Service to match a device to an IoT Platform if the device information does not explicitly require a specific IoT Platform. MEC IoT service sets up the sessions with the IoT Platform on behalf of the IoT devices and if required, maps information received from the devices to messages for specific topics as per the registered information. It also maps messages received from the IoT Platform to information it sends to the IoT devices. The MEC application providing the MEC IoT Service can also provide a UDP port which can be used to directly exchange information with NB-IoT devices as Unstructured Data when utilizing the Non-IP Data Delivery functionality of 3GPP networks.

In the context of this MEC IoT API, any kind of IoT platform is considered in principle (GS MEC 033 [21]): for example, proprietary IoT platforms typically offer a message bus (user transport) where IoT devices and IoT applications can publish on/subscribe to topics, a security framework, and additional features like, e.g., data analytics. MEC deployments with oneM2M can also be considered in principle, and in these cases IoT platforms supported by GS MEC 033 [21] can be represented by oneM2M compliant IoT platforms, as described previously in this paper. However, in order to enable deployments of oneM2M platform in MEC systems, and the related usage of IoT API at application level, a proper implementation of the IoTs is needed, e.g., via a service producing MEC App (refer to figure 17), to support IoT platform discovery, device provisioning, and transport configuration according to GS MEC 033 [21]. For example, the specification defines data types and attributes for the format of the messages to be published on the user transport in order to provide application-specific information about events related to the established session between the IoT device and the end IoT application(s). For example, the attribute “EventTopic” describes “Topics” where the message containing application-specific information should be published; also, the type “MBTransportInfo” defines a user transport based on a message bus, by extending the basic TransportInfo resource data type defined in Mp1 (ETSI GS MEC 011 [3]), specializing its scope to a transport of MB_TOPIC_BASED type. Consequently, the IoTS must support the definition of proper transport protocols supported by the IoT platform (as an example "MQTT” or “AMQP”).

In summary, in order to allow the consumption of the IoT API by enabling oneM2M compliant IoT platforms, an implementation of that IoTS service should be able to provide all the information (exposed by the IoT platform) to support, e.g.:

- the discovery of IoT platforms,
- the provisioning of IoT devices into the MEC system,
- the routing of communications between the devices and the requested IoT platform,
- the enablement of discovery and usability of the IoT platform's native APIs.

In all these cases, a proper service producing MEC App could usually serve the purpose (although is not in the scope of standardization work). Nonetheless, a future presence of such an IoTS service enabling oneM2M platforms integration (e.g., as open source reference implementations made available to app developers) could be appreciated by the industry, to foster better interworking among MEC and oneM2M systems.

5. Conclusion and future work

In this White Paper, we presented many use cases for IoT applications in Multi-access Edge Computing (MEC) environments, which is enabled by interworking of the two standards, i.e., oneM2M and ETSI ISG MEC. Both architectures can be based on NFV definitions, and this makes convenient the deployment of oneM2M common service layer in forms of NFV at the edge of network where a distributed cell of sites co-located with a MEC system could reliably provide value-added connectivity with QoS capabilities. However, such enablement of edge IoT using these standards requires designing a high interworking framework from these standards as well as figuring out how to deploy the IoT sensitive delay applications through this interworking for mobile edge computing based on IoT systems.

Then, this White Paper is meant to be an introductory guide to the industry, aiming at illustrating how MEC and oneM2M standards are compatible and can be integrated effectively. It first provided use cases of IoT using oneM2M and ETSI ISG MEC framework such mobility in smart city, real time analytical functions of sensed data generated by IoT devices, where some IoT platforms are instantiated as MEC applications to support IoT services in MEC environment. Then, the White Paper highlighted the importance of understanding the enablement of Multi-access Edge Computing for IoT scenarios and use-cases and the architectural interworking that this may design on to achieve the deployment of edge computing for IoT sensitive delay applications. We then provided a summary of key interworking modes and high-level overview of the orchestration of oneM2M edge instance.

Also, the White Paper described the published MEC IoT API (see GS MEC 033 [21]), that assists the deployment and usage of IoT devices which require additional support in a MEC environment, e.g., due to security constraints, limited power, compute, and communication capabilities, such as IoT and MTC devices. More in detail, the introduced MEC IoT Service (IoTS) provides means to incorporate heterogeneous IoT platforms in the MEC system and exposes IoT APIs to enable the configuration of the various components of the overall IoT system. In order to facilitate joint deployments of oneM2M platforms in MEC systems, and the related consumption of MEC IoT API at application level, some proper reference implementations of (e.g., a IoTS service producing MEC App, made available by e.g., open source communities) could be appreciated by the industry, to foster better integration among MEC APIs and oneM2M systems.

In summary, this White Paper presented a high-level overview of how the interworking between oneM2M and MEC is possible to support IoT sensitive delay applications. However, we consider this paper as a useful starting point on the journey. Therefore, oneM2M members in collaboration of ETSI ISG MEC group can work together to even improve their standardization efforts to support effective and highly profitable deployments for MEC-based IoT applications.

References

[1] ETSI GS MEC 003: "Multi-access Edge Computing (MEC); Framework and Reference Architecture".
www.etsi.org/deliver/etsi_gs/MEC/001_099/003/03.01.01_60/gs_MEC003v030101p.pdf

[2] ETSI GS MEC 010-2: “Multi-access Edge Computing (MEC); Application lifecycle, rules and requirements management”.
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/01002/03.01.01_60/gs_MEC01002v030101p.pdf

[3] ETSI GS MEC 011: "Multi-access Edge Computing (MEC); Edge Platform Application Enablement".
www.etsi.org/deliver/etsi_gs/MEC/001_099/011/03.01.01_60/gs_MEC011v030101p.pdf

[4] ETSI GS MEC 016: "Multi-access Edge Computing (MEC); Device Application Interface".
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/016/02.02.01_60/gs_MEC016v020201p.pdf

[5] ETSI GS MEC 009: "Multi-access Edge Computing (MEC); General principles, patterns and common aspects of MEC Service APIs,
www.etsi.org/deliver/etsi_gs/MEC/001_099/009/03.01.01_60/gs_MEC009v030101p.pdf

[6] ETSI GS MEC 021: “Multi-access Edge Computing (MEC); Application Mobility Service API”
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/021/02.02.01_60/gs_MEC021v020201p.pdf

[7] ETSI GS MEC 026: “Multi-access Edge Computing: Support for regulatory requirements”.
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/026/02.01.01_60/gs_MEC026v020101p.pdf

[8] ETSI GR MEC 031: Multi-access Edge Computing; MEC 5G Integration.
https://www.etsi.org/deliver/etsi_gr/MEC/001_099/031/02.01.01_60/gr_MEC031v020101p.pdf

[9] ETSI GS MEC 028: “Multi-access Edge Computing (MEC); WLAN Access Information API”,
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/028/02.03.01_60/gs_MEC028v020301p.pdf

[10] ETSI GS MEC 029: “Multi-access Edge Computing (MEC); Fixed Access Information API”.
https://www.etsi.org/deliver/etsi_gs/MEC/001_099/029/02.02.01_60/gs_MEC029v020201p.pdf

[11] ETSI GR MEC 017: “Multi-access Edge Computing (MEC); Deployment of Mobile Edge Computing in an NFV environment”.
https://www.etsi.org/deliver/etsi_gr/MEC/001_099/017/01.01.01_60/gr_MEC017v010101p.pdf

[12] ETSI GR MEC 027: “Multi-access Edge Computing (MEC); Study on MEC support for alternative virtualization technologies”

https://www.etsi.org/deliver/etsi_gr/MEC/001_099/027/02.01.01_60/gr_MEC027v020101p.pdf

[13] ETSI GR MEC 022: "Multi-access Edge Computing (MEC); Study on support for V2X Use Cases". https://www.etsi.org/deliver/etsi_gr/MEC/001_099/022/02.01.01_60/gr_MEC022v020101p.pdf

[14] ETSI GS MEC 030: "Multi-access Edge Computing (MEC); V2X Information Services API" https://www.etsi.org/deliver/etsi_gs/MEC/001_099/030/03.01.01_60/gs_MEC030v030101p.pdf

[15] ETSI GR MEC 024: "Multi-access Edge Computing (MEC); Support for network slicing". https://www.etsi.org/deliver/etsi_gr/MEC/001_099/024/02.01.01_60/gr_MEC024v020101p.pdf

[16] ETSI GR MEC-DEC 025: "Multi-access Edge Computing (MEC); MEC Testing Framework" https://www.etsi.org/deliver/etsi_gr/MEC-DEC/001_099/025/02.01.01_60/gr_MEC-DEC025v020101p.pdf

[17] ETSI GS MEC-DEC 032 parts 1 to 3: "Multi-access Edge Computing (MEC); API Conformance Test Specification": https://www.etsi.org/deliver/etsi_gs/MEC-DEC/001_099/03201/03.01.01_60/gs_MEC-DEC03201v030101p.pdf , https://www.etsi.org/deliver/etsi_gs/MEC-DEC/001_099/03202/02.01.01_60/gs_MEC-DEC03202v020101p.pdf and https://www.etsi.org/deliver/etsi_gs/MEC-DEC/001_099/03203/03.01.01_60/gs_MEC-DEC03203v030101p.pdf

[18] ETSI GR MEC 035: "Multi-access Edge Computing (MEC); Study on inter-MEC systems and MEC-Cloud systems coordination". https://www.etsi.org/deliver/etsi_gr/MEC/001_099/035/03.01.01_60/gr_MEC035v030124p.pdf

[19] ETSI GS MEC 040: "Multi-access Edge Computing (MEC); Federation enablement APIs" https://www.etsi.org/deliver/etsi_gs/MEC/001_099/040/03.01.01_60/gs_MEC040v030101p.pdf

[20] ETSI GS MEC 013:" Multi-access Edge Computing (MEC); Location API". https://www.etsi.org/deliver/etsi_gs/MEC/001_099/013/03.01.01_60/gs_MEC013v030101p.pdf

[21] ETSI GS MEC 033: "Multi-access Edge Computing (MEC); IoT API". https://www.etsi.org/deliver/etsi_gs/MEC/001_099/033/03.01.01_60/gs_MEC033v030101p.pdf

[22] ETSI GR MEC 038: "Multi-access Edge Computing (MEC); MEC in Park Enterprises Deployment". https://www.etsi.org/deliver/etsi_gr/MEC/001_099/038/03.01.01_60/gr_MEC038v030101p.pdf

[23] ETSI GS MEC 037: “Multi-access Edge Computing (MEC); Application Package Format and Descriptor Specification”. https://www.etsi.org/deliver/etsi_gs/MEC/001_099/037/03.01.01_60/gs_MEC037v030101p.pdf

[24] J. Swetina, G. Lu, P. Jacobs, F. Ennesser and J. Song, "Toward a standardized common M2M service layer platform: Introduction to oneM2M," in IEEE Wireless Communications, vol. 21, no. 3, pp. 20-26, June 2014, doi: 10.1109/MWC.2014.6845045

[25] oneM2M TS-0026, “3GPP Interworking”, V4.7.0

[26] oneM2M TS-0040, “Modbus Interworking”, V4.0.0

[27] oneM2M TR-0052, “Study on Edge and Fog Computing in oneM2M System”, V0.13.1

[28] oneM2M TS-0001, “Functional Architecture”, V4.19.0

[29] oneM2M TR-0001, “Use Cases Collection”, V5.0.0

[30] oneM2M TR-0018, “Industrial Domain Enablement”, V2.5.1

[31] 3GPP TS 23.558: "Architecture for enabling Edge Applications".

[32] 3GPP TR 23.700-98-i00: “Study on enhanced Architecture for enabling Edge Applications”.

Abbreviations

3GPP	Third Generation Partnership Program
5G	Fifth mobile network Generation
4G	Forth mobile network Generation
CSF	Customer Facing Service portal
CSFs	Common Service Functions
DNS	Domain Name Service
GCF	Global Certification Forum
GIoTs	Global IoT Services
GS	Group specifications
HV	Host Vehicle
ICT	Information Communication Technology
IoT	Internet of Things
IT	Information Technology
ISVs	Independent Software Vendors
LCM	Life Cycle management
M2M	Machine to Machine
MEAO	MEC Application Orchestrator.
NFV	Network Function Virtualization
NFVO	NFV Orchestrator.
OCF	Open Connectivity Foundation
OTT	Over the Top
PoC	Proof of Concept
SDO	Standards Development Organization
UE	User Equipment
VM	Virtual Machines
VNFs	Virtual Network Functions
VRUs	Vulnerable Road Users

ETSI
06921 Sophia Antipolis CEDEX, France
Tel +33 4 92 94 42 00
info@etsi.org
www.etsi.org