ETSI GR MEC-DEC 050 V4.1.1 (2025-11)

GROUP REPORT

Multi-access Edge Computing (MEC); ESTIMED Use Cases & Recommendations

Disclaimer

The present document has been produced and approved by the Multi-access Edge Computing (MEC) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGR/MEC-DEC050EstimedRec

Keywords
M2M, MEC, oneM2M

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members** , and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Multi-access Edge Computing (MEC).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document provides a set of selected use cases that leverage IoT and Edge computing technologies, with a particular focus on standard-based implementations where oneM2M and ETSI MEC frameworks can be applied. Building on the joint MEC-oneM2M White Paper [i.2], it refines and expands the identified synergies between these frameworks and offers a detailed analysis of architectural evolutions, including necessary components, required mappings, and potential enhancements to facilitate interoperability and integration between oneM2M and MEC-based deployments.

By analysing real-world and industry-relevant use cases, the present document highlights how the convergence of IoT and Edge computing can enable scalable, efficient, and interoperable solutions across different domains. The scope extends to evaluating the applicability of existing standards, identifying gaps, and proposing recommendations to ensure alignment with ongoing standardization efforts within ETSI and oneM2M. The present document serves as a reference for stakeholders to understand the practical implications of integrating MEC and oneM2M technologies in future IoT-Edge ecosystems.

The scope includes recommendations and proposals for MEC-oneM2M interworking deployment patterns across the edge-cloud continuum, covering the four deployment options to enable latency-critical processing and service continuity, the exposure of oneM2M IoT data and control services through the MEC platform for consumption by applications that access and actuate IoT resources, and hybrid multi-tier arrangements that distribute functions across user equipment or gateways, and centralized infrastructure for hierarchical data management and orchestration. It also encompasses cross-edge Artificial Intelligence approaches that are central to the stated goals, including federated learning for privacy-preserving collaborative model training and edge inference across MEC nodes and oneM2M-managed data spaces, and swarm computing for distributed, real-time coordination of agents and devices supported by edge-hosted IoT services and MEC compute. The resulting recommendations address the functional enablers, mappings, and enhancements required to operationalize these deployment patterns and intelligent capabilities across representative use cases.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents may be useful in implementing an ETSI deliverable or add to the reader's understanding, but are not required for conformance to the present document.

[i.1] ETSI GS MEC 003: "Multi-access Edge Computing (MEC); Framework and Reference Architecture".

[i.2] ETSI White Paper No. #59: "Enabling Multi-access Edge Computing in Internet-of- Things: how to deploy ETSI MEC and oneM2M".

[i.3] ESTIMED Project Official Website.

[i.4] ETSI GS MEC 009: "Multi-access Edge Computing (MEC); General principles, patterns and common aspects of MEC Service APIs".

[i.5] ETSI GS MEC 010-2: "Multi-access Edge Computing (MEC); MEC Management; Part 2: Application lifecycle, rules and requirements management".

[i.6] ETSI GS MEC 011: "Multi-access Edge Computing (MEC); Edge Platform Application Enablement".

[i.7] ETSI GS MEC 013: "Multi-access Edge Computing (MEC); Location API".

[i.8] ETSI GS MEC 012: "Multi-access Edge Computing (MEC); Radio Network Information API".

[i.9] ETSI GS MEC 015: "Multi-access Edge Computing (MEC); Traffic Management APIs".

[i.10] ETSI GS MEC 016: "Multi-access Edge Computing (MEC); Device application interface".

[i.11] ETSI GS MEC 021: "Multi-access Edge Computing (MEC); Application Mobility Service API".

[i.12] ETSI GS MEC 028: "Multi-access Edge Computing (MEC); WLAN Access Information API".

[i.13] ETSI GS MEC 029: "Multi-access Edge Computing (MEC); Fixed Access Information API".

[i.14] ETSI GS MEC 030: "Multi-access Edge Computing (MEC); V2X Information Services API".

[i.15] ETSI GS MEC 033: "Multi-access Edge Computing (MEC); IoT API".

[i.16] ETSI GS MEC 040: "Multi-access Edge Computing (MEC); Federation enablement APIs".

[i.17] ETSI GS MEC 045: "Multi-access Edge Computing (MEC); QoS Measurement API".

[i.18] ETSI GS MEC 046: "Multi-access Edge Computing (MEC); Sensor-sharing API".

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

3D	Three-Dimensional
3GPP	3rd Generation Partnership Project
5G	Fifth Generation mobile networks
ADN	Application Dedicated Node
ADN-AE	Application Dedicated Node - Application Entity
AE	Application Entity
AGV	Autonomous Guided Vehicles
AIS	Automatic Identification System
AI	Artificial Intelligence
API	Application Programming Interface
API GW	API GateWay
AR	Augmented Reality
ASN	Application Service Node
ASN-CSE	Application Service Node - Common Services Entity
AV	Autonomous Vehicle
CNC	Computer Numerical Control
CoAP	Constrained Application Protocol
CPE	Customer Premises Equipment
CPU	Central Processing Unit

CSE	Common Services Entity
CSF	Common Service Function
DT	Digital Twin
FL	Federated Learning
GPU	Graphics Processing Unit
HTTP	Hypertext Transfer Protocol
HV	Host Vehicle
IDT	Industrial Digital Twin
IN	Infrastructure Node
IN-AE	Infrastructure Node - Application Entity
IN-CSE	Infrastructure Node - Common Services Entity
IoT	Internet of Things
IPE	Interworking Proxy application Entity
LIDAR	LIght Detection And Ranging
MEP	MEC Platform
MN-AE	Middle Node AE
MN-CSE	Middle Node - Common Services Entity
NaaS	Network as a Service
NFV	Network Function Virtualisation
NoDN	Non-oneM2M Node
NSE	Network Services Entity
OAS	OpenAPI Specification
OCF	Open Connectivity Foundation
ROC	Remote Operation Centre
SAREF	Smart Appliances REFerence
VIM	Virtualisation Infrastructure Manager
VRU	Vulnerable Road User

4 Overview

This clause introduces the present document by clearly defining its purpose, scope, and key objectives. It lays the groundwork for the integration of the ETSI MEC and oneM2M frameworks, highlighting their critical role in enabling advanced Edge and IoT deployments across a wide range of application domains.

The ESTIMED project [i.3] takes a use case-driven approach to steer the integration of ETSI MEC and oneM2M. This strategy effectively demonstrates the tangible benefits of merging edge computing capabilities with standardized IoT architectures. By anchoring the analysis in real-world scenarios, the approach not only guides integration efforts but also informs future standardization initiatives to better meet operational demands. The project emphasizes identifying new functional requirements and potential extensions to existing specifications, thereby contributing to the continuous refinement and evolution of both MEC and oneM2M frameworks in response to emerging IoT and edge computing challenges. Following this methodology, the present document is organized into three main clauses, each addressing a core aspect of the analysis.

Clause 5 delivers a domain-centric analysis of carefully selected use cases, illustrating the practical application of MEC and oneM2M in Edge-IoT environments. Use cases are categorized by relevant domains-such as Mobility, Industrial, and Maritime to reflect their specific operational contexts. Within each domain, use cases are thoroughly examined, covering the scenario context, involved stakeholders, technical and operational requirements, and key challenges. The clause also explores how MEC and oneM2M technologies contribute to overcoming these challenges. The use cases were identified through a structured engagement with stakeholders, facilitated by a dedicated form designed to capture all critical details consistently, including descriptions, actors, requirements, preconditions and postconditions, triggers, and interaction flows.

Building on this foundation, clause 6 analyses the principal architectures of MEC and oneM2M frameworks, emphasizing their core functionalities and complementarity in supporting Edge-IoT deployments. For each use case, the clause maps relevant architectural components and capabilities from both frameworks, highlighting overlaps as well as identifying new features that could enhance support for specific scenarios. This mapping lays the groundwork for a comprehensive architectural reflection and technical assessment.

Clause 7 includes recommendations that enable a framework for the federation and orchestration of MEC and oneM2M instances. It also discusses how these capabilities can be used in Swarm Computing and Federated Learning applications.

Overall, the present document establishes a structured and cohesive foundation to understand and advance how MEC and oneM2M can jointly enable scalable, interoperable, and forward-looking Edge-IoT solutions.

5 Edge & IoT Domains and Use Cases

5.1 Introduction

This clause introduces the application domains and use cases explored in the context of the ESTIMED project, focusing on how the integration of the ETSI MEC and oneM2M frameworks enables next-generation Edge-IoT solutions. The clause builds upon a use case-driven methodology that emphasizes real-world scenarios as the foundation for architectural mapping, technical analysis, and standardization recommendations.

The goal of this clause is to present selected domains where edge computing and IoT convergence are driving tangible benefits across verticals such as mobility, industry, maritime operations, digital experiences, and smart living. For each domain, representative use cases have been identified to demonstrate the practical value of combining MEC's edge capabilities with oneM2M's standardized IoT platform. These use cases illustrate how the integration supports low-latency data processing, scalable service delivery, and intelligent, context-aware applications.

Table 5.1-1 summarizes the domains and the associated use cases covered in this clause.

Table 5.1-1: Domains covered

Domain	Use Cases	Focus
Smart City & Mobility	• Autonomous Vehicles and Edge Continuum • Vulnerable Road User Detection	Real-time data processing, V2X communication, urban mobility optimization, traffic safety
Industrial & Robotics	• Swarm-based Autonomous Ant Delivery Optimization • Smart Warehouse Automation • Industrial Digital Twins	Intelligent coordination, low-latency control, industrial automation, edge-based robotics
Maritime	• Assisted Manoeuvring for Connected Ships	Remote vessel monitoring, mission-critical edge processing, seamless edge-cloud communication
Metaverse	• Smart Virtual Shopping Service	IoT-enhanced virtual environments, edge-hosted AI analytics, immersive low-latency digital experiences
Future Home	• Advanced Smart Home Services	Real-time media, education, health, and automation services within personalized, responsive smart environments

Each use case is examined in detail within its domain context, describing how MEC and oneM2M contribute to solving specific challenges, enabling innovation, and supporting interoperability. The domains and use cases form the analytical foundation for the architectural mappings and technical recommendations that follow in subsequent clauses.

5.2 Smart City & Mobility

5.2.1 Description

This clause focuses on the Smart City and Mobility domain, highlighting two key use cases: Autonomous Vehicles and Edge Continuum, and Vulnerable Road User Detection. It examines how MEC and oneM2M frameworks support advanced urban mobility solutions by enabling real-time data processing, vehicle-to-infrastructure communication, and coordinated edge intelligence. These capabilities contribute to improved traffic management, enhanced road safety, and more efficient transportation systems within smart urban environments.

5.2.2 Autonomous Vehicle with Continuous Edge Computing

This use case demonstrates the integration of oneM2M's IoT platform with ETSI MEC's edge computing framework to enable continuous real-time processing for Autonomous Vehicles (AVs). The core system (schematically reported in Figure 5.2.2-1) consists of a cloud-based oneM2M IN-CSE (centralized IoT hub) for managing vehicle and environmental data, and MN-CSE instances deployed on MEC nodes that process time-sensitive data such as vehicle sensor inputs, road conditions, and traffic signals with ultra-low latency.

Service Continuity is a critical aspect of this use case. Initially, the IN-CSE in the cloud manages and stores all data related to the AV, including location, sensor data (e.g. LIDAR, radar), and external conditions such as road infrastructure. When an AV enters a zone covered by a nearby MEC node, the IN-CSE dynamically offloads data processing tasks (such as real-time object detection, collision risk analysis, and navigation updates) to the MN-CSE instances deployed at the edge. These instances not only handle locally generated data but also share additional real-time data from other MEC applications to enhance decision-making. The MN-CSE processes the data locally, providing immediate feedback and decision-making (e.g. braking, turning, lane-changing) with minimal latency, ensuring that the vehicle operates efficiently.

As the AV moves across different zones, it will seamlessly transition from one MEC node to another. The system continuously monitors the AV's location and detects when the vehicle approaches a new MEC zone. Upon detection, the system initiates the migration of the MN-CSE's offloaded data and applications to the new MEC node. This migration ensures that all real-time data and applications are immediately available at the new location, guaranteeing that the AV experiences uninterrupted service as it moves. The oneM2M system synchronizes between the cloud and edge, ensuring that all data is consistent and up-to-date. This edge-to-edge service continuity is critical for maintaining performance and safety in dynamic environments.

By integrating oneM2M's standardized data management with MEC's edge computing capabilities, this use case shows real-time, low-latency processing, while also enabling the migration of applications across MEC zones without compromising service quality. As the AV moves, the system adjusts dynamically, ensuring continuous service delivery through the seamless orchestration between the cloud, MEC, and edge devices. This use case not only supports static services but also mobility services, ensuring that the AV receives timely updates and services as it traverses multiple zones.

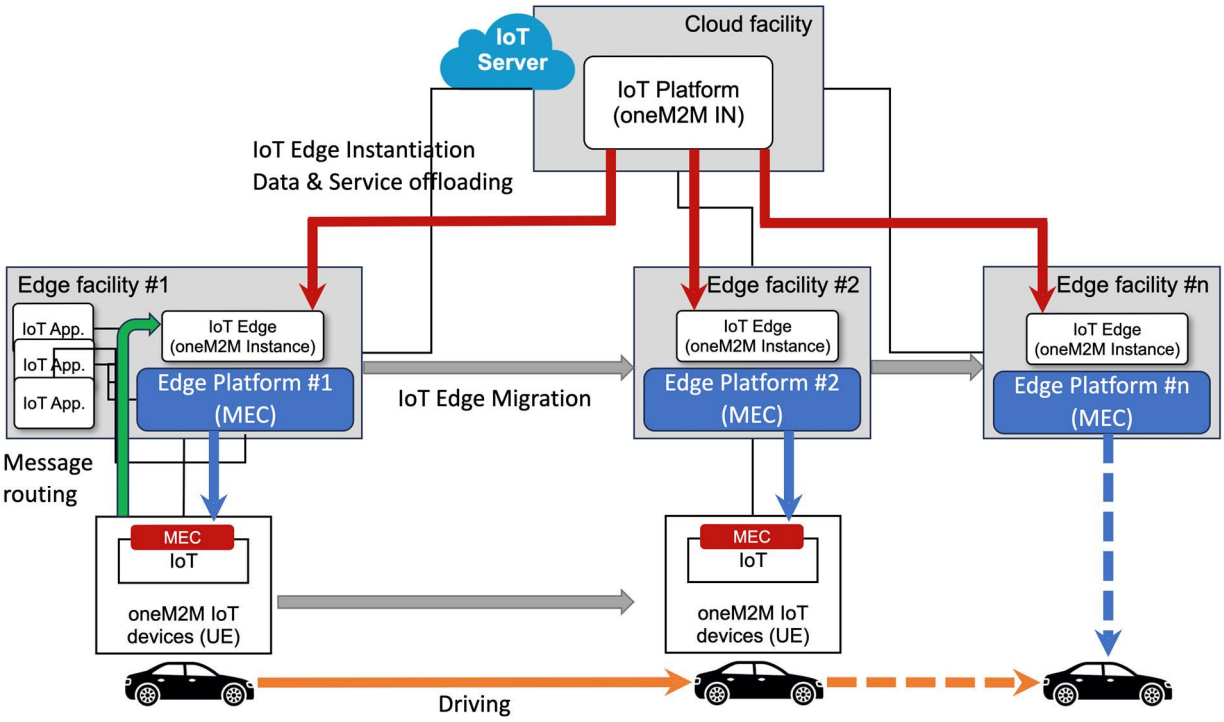

Figure 5.2.2-1: Autonomous Vehicle Edge Computing service continuity

5.2.3 Vulnerable Road Users

This scenario illustrates a collaborative architecture between oneM2M's IoT platform and ETSI MEC's edge computing framework to enable real-time Vulnerable Road User (VRU) detection for connected vehicles. As illustrated in Figure 5.2.3-1, at the core of the system, a cloud-based oneM2M IN-CSE serves as the central IoT hub, storing and managing VRU-related data such as pedestrian locations, cyclist trajectories, and road conditions.

Meanwhile, edge-located MN-CSE instances, deployed as MEC applications on distributed edge nodes, act as localized IoT platforms that process time-sensitive data near the vehicles. When a Host Vehicle (HV) enters a zone covered by a nearby MEC node, the cloud IN-CSE dynamically offloads VRU detection tasks - such as real-time analytics and safety alert generation-to the edge-resident MN-CSE. This offloading ensures ultra-low-latency processing, as the MN-CSE can immediately analyse VRU data (e.g. from smartphones, roadside sensors) and issue warnings to the HV without relying on distant cloud servers.

The integration of oneM2M's standardized IoT data management with MEC's edge computing capabilities creates a responsive and location-aware safety system. For instance, when an HV prepares to make a left turn, the MN-CSE on the nearest MEC node evaluates real-time VRU positions and sends instant collision-risk alerts directly to the vehicle. The MEC platform further enhances this process by providing network-aware optimizations, such as selecting the best available connection (5G, LTE-V2X) for data exchange. As the HV moves across different zones, the system seamlessly transitions tasks between edge nodes or back to the cloud, ensuring uninterrupted service. By combining cloud scalability with edge-level agility, this approach not only improves road safety but also demonstrates how IoT and edge computing can converge to support latency-critical automotive applications. An offloading concept locating tasks and resources to a place where close to users can be applied to this VRU detection service. In this case, a service can be provided to users with very short delay.

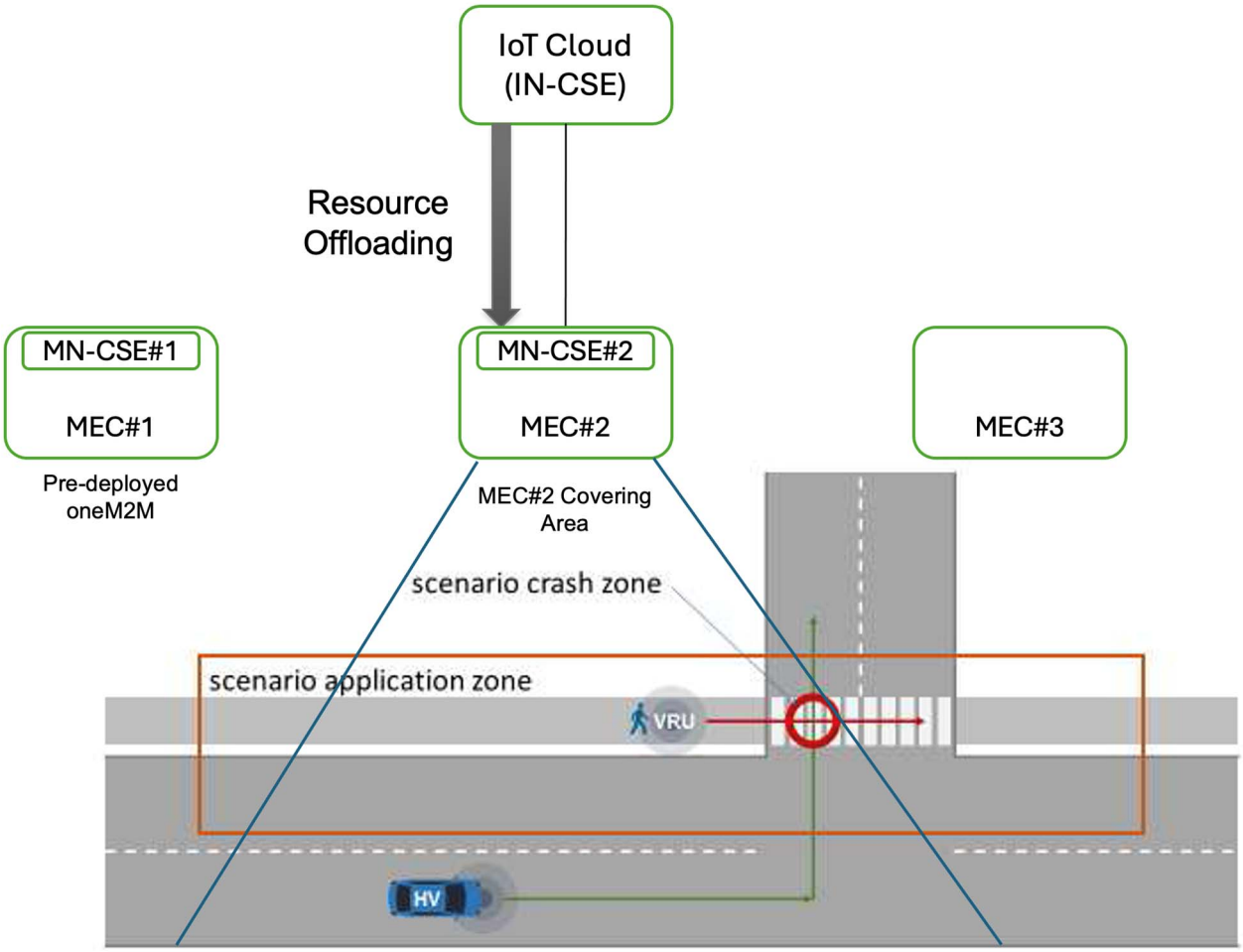

Figure 5.2.3-1: Scenario of resource and tasks offloading to support VRU application

5.3 Industrial & Robotics

5.3.1 Description

This clause covers the industrial and robotics domain with a focus on use cases such as Swarm-based Autonomous Ant Delivery Optimization, Smart Warehouse Automation, and Industrial Digital Twins. It explores how MEC and oneM2M frameworks enable intelligent coordination, real-time control, and seamless IoT integration in complex industrial and logistics environments. By leveraging edge capabilities, these use cases benefit from low-latency processing, enhanced automation, and improved operational efficiency across urban and industrial ecosystems.

5.3.2 Swarm-based Autonomous Ant Delivery Optimization

This scenario explores a hybrid swarm-based autonomous delivery system inspired by ant colony behaviour. Swarm computing is a computational model based on the collective behaviour of decentralized systems, often inspired by nature (e.g. ants, bees, or fish schools). In traditional swarm computing, multiple objects (agents) collaborate autonomously to solve problems or perform tasks by sharing information and interacting with each other without the need for centralized control. Each agent in the swarm makes decisions based on local information and interactions, leading to emergent global behaviours that solve complex tasks, such as pathfinding, resource allocation, or optimization.

However, when applied to real-world systems like autonomous delivery or energy management, fully decentralized swarm systems face several limitations. For instance, without any centralized coordination or additional guidance, the swarm may face challenges in optimizing complex tasks or adapting to dynamic, large-scale environments. Swarm systems can struggle with task allocation, resource optimization, and dealing with environmental changes when working entirely autonomously without external help. Thus, this use case explores how swarm agents, integrated with a cloud IoT platform and assisted by edge nodes, can more effectively solve given problems.

In this scenario, the edge IoT platform (e.g. MN in oneM2M), which acts as an edge node in a decentralized system, plays a critical role in overcoming these limitations. While robot (agents) in the swarm communicates and collaborates autonomously, the edge IoT platform provides indirect feedback by offering information that robots may not directly perceive, such as road conditions, elevator status, and indoor obstacles. This feedback mechanism enhances the swarm's performance and helps in optimizing the delivery task. Specifically, the edge computing platform (e.g. MEC) allows for ultra-low latency processing and ensures that robots can receive real-time updates on the environment. By integrating the edge IoT platform for efficient resource management and the edge computing platform for real-time environmental feedback, this hybrid swarm system becomes more capable of optimizing delivery paths and avoiding congestion, while still maintaining decentralized autonomy.

As the swarm operates, it gradually evolves into a highly efficient delivery network, capable of real-time obstacle avoidance and dynamic path adaptation. This process exemplifies the power of Swarm Computing when integrated with an IoT platform deployed on Edge infrastructure. In this context (depicted also in Figure 5.3.2-1), Swarm Computing refers to a decentralized approach in which multiple autonomous agents - represented as Application Entities (AEs) within the oneM2M framework - interact and collaborate using locally available information. These agents are able to solve tasks independently, without relying on centralized control, resulting in emergent collective intelligence.

The IoT platform, implemented as an edge node (such as a Middle Node, MN, in oneM2M), plays a pivotal role by providing indirect feedback to the swarm. It supplies real-time environmental data that individual agents may not directly perceive, thereby enhancing the swarm's overall performance and adaptability. This feedback mechanism enables agents to make informed decisions, optimize delivery routes, and respond effectively to changing conditions.

Complementing the IoT platform, the Edge infrastructure - exemplified by the MEC platform - processes data locally to deliver ultra-low-latency feedback. By hosting both IoT services and computational resources at the edge, the MEC platform ensures that swarm agents receive timely updates and actionable insights. This local processing capability is crucial for improving coordination among agents, optimizing task execution, and maintaining operational efficiency in dynamic environments.

Together, the integration of Swarm Computing, edge-based IoT platforms, and MEC infrastructure demonstrates a robust solution for complex delivery optimization challenges. The system's ability to self-organize, adapt to obstacles, and continuously refine its operations highlights the transformative potential of combining decentralized intelligence with real-time edge computing.

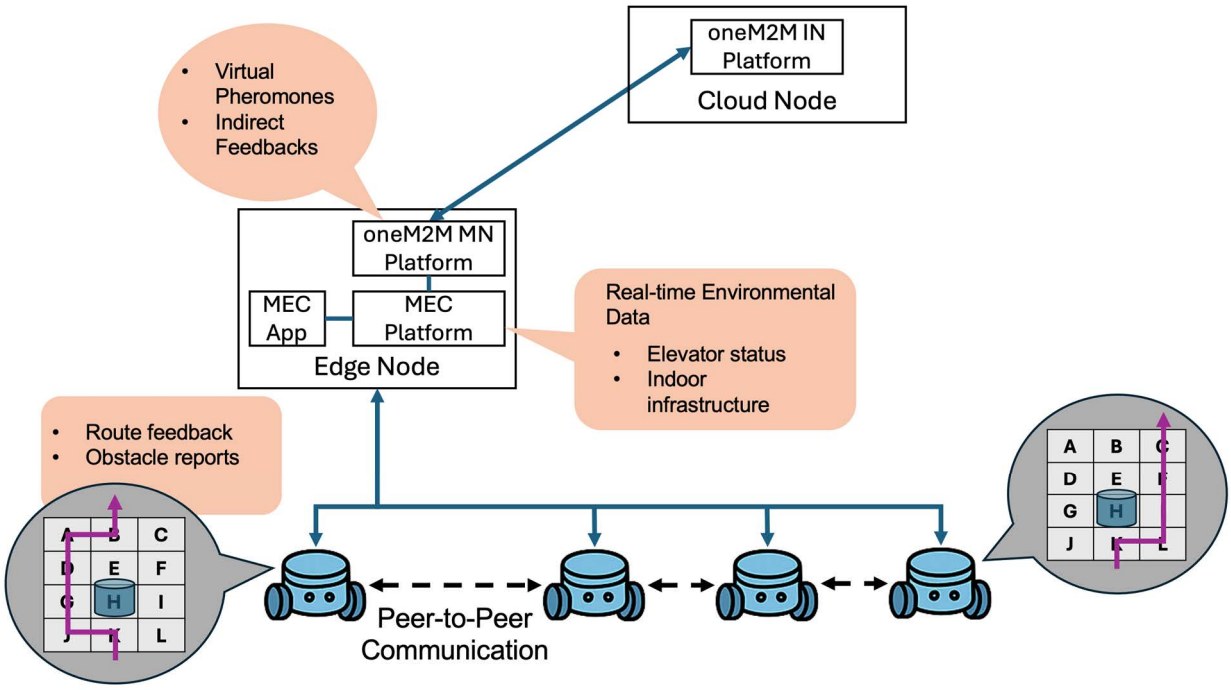

Figure 5.3.2-1: Data flow and interaction in a hybrid robot swarm system

5.3.3 Smart Warehouse Automation

This use case describes how ETSI MEC and oneM2M platforms can work together to enable real-time warehouse automation, including coordination of Autonomous Guided Vehicles (AGVs), environmental monitoring, and asset tracking. The system architecture reported in Figure 5.3.3-1 integrates a centralized oneM2M IN-CSE for managing warehouse infrastructure data and multiple MN-CSE instances deployed as MEC applications to provide low-latency control and decision-making.

The oneM2M IN-CSE collects and stores data from IoT devices such as temperature and humidity sensors, RFID tags, and AGV telemetry. When time-critical operations occur (e.g. routing AGVs, responding to a gas leak, or dynamic reallocation of assets), relevant tasks are offloaded to MEC-based MN-CSEs. The MEC platform processes this data locally, allowing immediate decisions such as rerouting AGVs, sending alerts to personnel, or controlling air filtration systems.

As warehouse operations span large areas, AGVs may move between MEC zones. The MEC platform supports service continuity by transferring control logic and task context across MEC nodes, ensuring uninterrupted automation and safety.

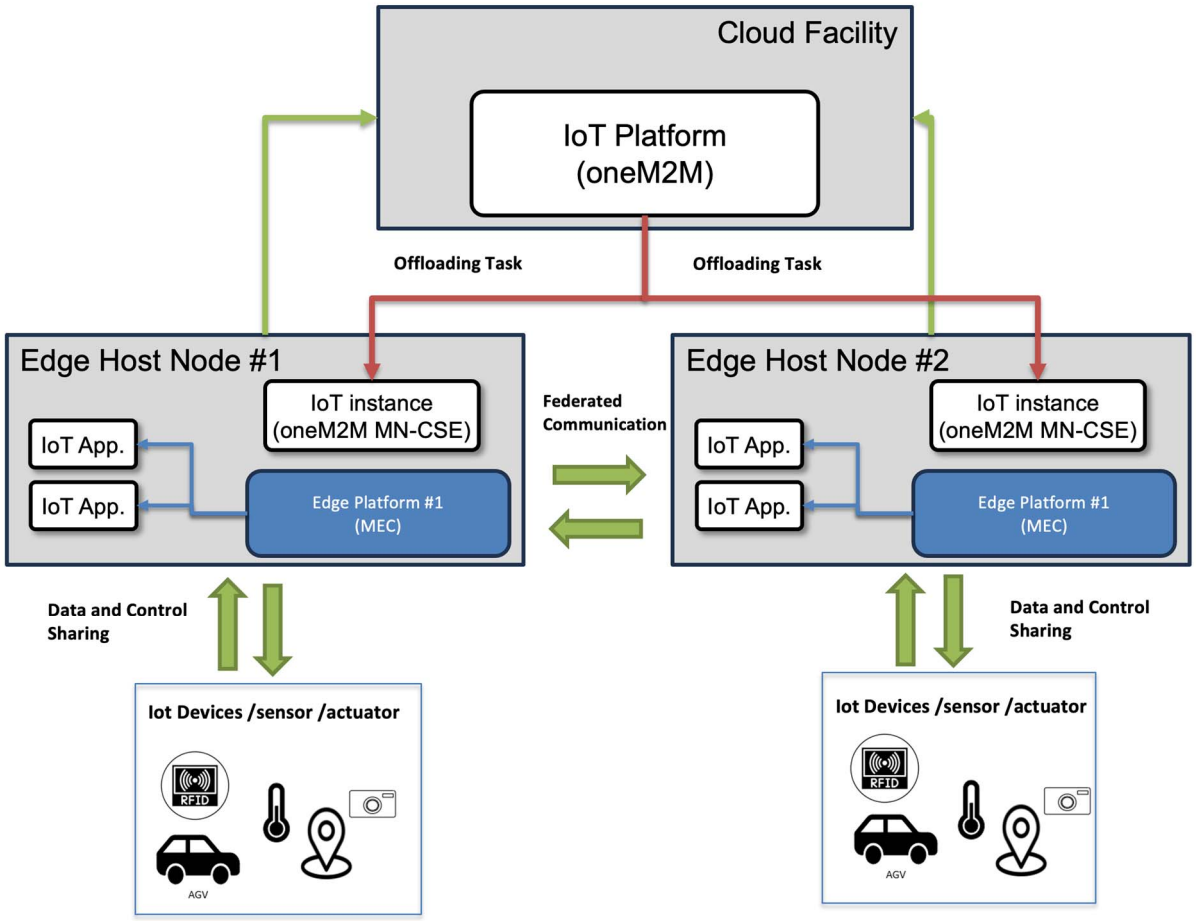

Figure 5.3.3-1: High-level overview of the warehouse automation scenario

5.3.4 Industrial Digital Twins

This use case illustrates the integration of the oneM2M IoT platform with the ETSI MEC edge computing framework to support Industrial Digital Twins (IDTs) in smart manufacturing environments. The goal is to enable continuous monitoring, analysis, and optimization of industrial processes by deploying synchronized digital representations of physical assets across both cloud and edge infrastructures. This approach is fundamental to enabling real-time decision-making, predictive maintenance, and autonomous control in dynamic and distributed industrial settings.

The overall system architecture illustrated in Figure 5.3.4-1 relies on a cloud-based oneM2M Infrastructure Node Common Service Entity (IN-CSE), which functions as a centralized orchestrator and digital repository, and multiple Middle Node Common Service Entities (MN-CSEs) deployed at the edge on MEC nodes physically co-located with manufacturing equipment or microfactories. These edge-based MN-CSEs are responsible for processing time-sensitive data streams such as sensor readings from production lines, robotic cell statuses, energy usage metrics, and environmental conditions with ultra-low latency.

Real-time insight is achieved through the ability of MN-CSEs to execute localized, time-critical functions including anomaly detection, in-line quality control, and safety compliance checks. The system ensures seamless service continuity across the edge-cloud continuum by dynamically migrating services and synchronizing digital twin data as assets move across operational zones. This continuity is crucial for uninterrupted performance of digital twins as the physical counterparts transition between factory cells or sites. The solution is grounded in oneM2M's standardized resource and data models, enabling interoperability across diverse devices and platforms.

Initially, the IN-CSE in the cloud hosts the master representations of industrial assets - such as CNC machines, collaborative robots (cobots), or Automated Guided Vehicles (AGVs) - maintaining records of their operational states, historical data, and environmental contexts. When local operations are initiated in a specific factory cell that is served by a MEC node, the IN-CSE delegates processing to the nearest MN-CSE deployed on the edge. This MN-CSE ingests live data streams including vibration patterns, temperature levels, system workload, and diagnostic logs, and performs localized analytics such as early fault detection, micro-adjustments to optimize yield, and predictive maintenance alerts. Additionally, the MN-CSE may interact with other applications hosted on the MEC platform, such as AI-based vision systems for quality inspection or real-time control systems for robotic motion, by leveraging MEC-provided APIs.

As mobile industrial assets like AGVs or modular production units relocate within or between factory environments, the system ensures that all associated digital twin data and computational contexts are seamlessly migrated to a new MN-CSE at the appropriate MEC node. This handover is coordinated by the IN-CSE and utilizes MEC's Application Mobility and Platform Services to maintain uninterrupted functionality. Throughout this process, the oneM2M system keeps cloud and edge representations synchronized, enabling both high-level analytics-such as key performance indicators and production benchmarking - and immediate operational control. The benefits of this integrated approach are significant. Sub-second response times are achieved by executing control loops directly at the edge, reducing latency to a minimum. The architecture supports uninterrupted service delivery even during the movement of assets between operational zones, enhancing system reliability. Its modular and scalable nature allows deployment across a wide range of industrial environments. Finally, adherence to oneM2M standards and MEC orchestration capabilities guarantees compatibility and future-proof integration across heterogeneous platforms.

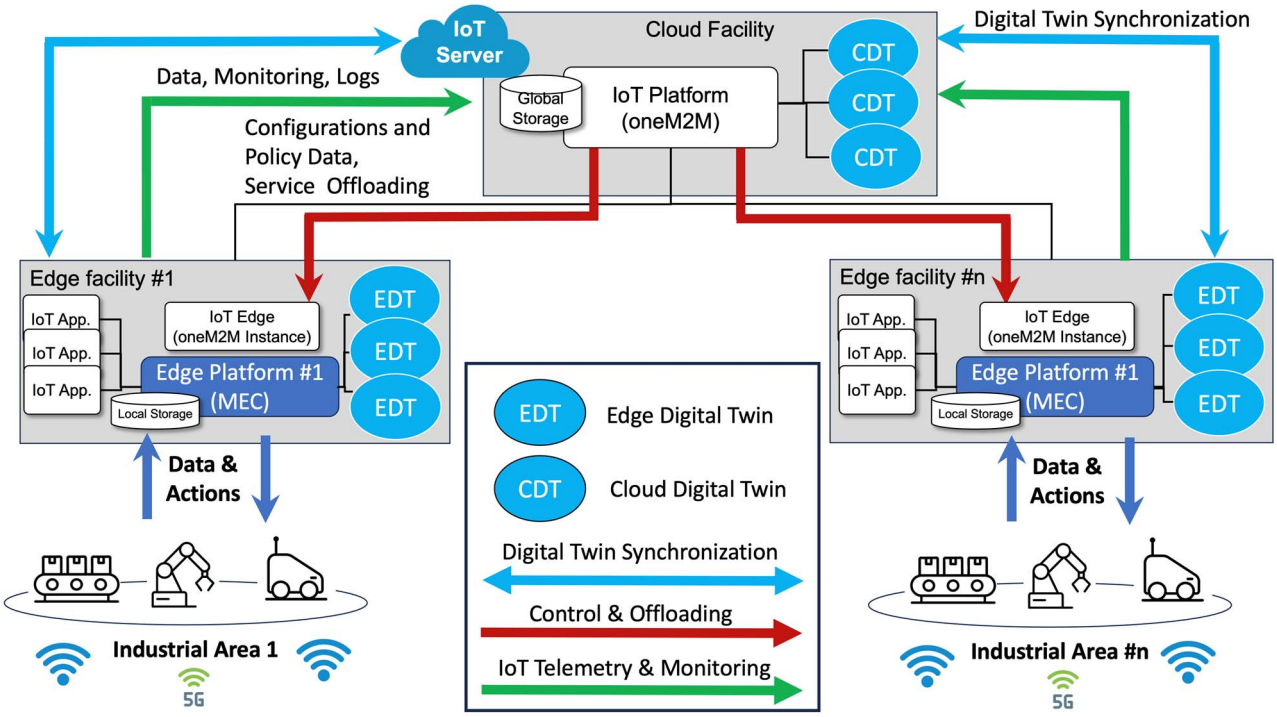

Figure 5.3.4-1: Industrial Digital Twins supported and enabled for both IoT/IIoT interactions and edge-cloud computation by the integrated capabilities of oneM2M and ETSI MEC

5.4 Maritime

5.4.1 Description

The maritime sector is undergoing significant transformation through the adoption of automation and advanced sensing technologies that enable vessels to navigate and perform tasks autonomously. These connected ships are expected to maintain continuous communication with operation centres across the edge-cloud continuum to ensure effective monitoring and decision-making. This clause presents a use case that integrates the oneM2M IoT platform with the ETSI MEC edge computing framework to deliver real-time assisted manoeuvring for unmanned vessels in seaport areas, supporting low-latency data processing, seamless service continuity, and mission-critical functionalities.

5.4.2 Assisted Manoeuvring for Autonomous Ship

Maritime Autonomous Surface Ships (MASS) technologies have been developing rapidly during last decade. Unmanned ships should be able to navigate without colliding with other vessels, considering the naval traffic and unexpected situations. These ships are usually equipped with advanced technologies like integrated automation systems based on AI-models and sensing capabilities which allow them to navigate, make decisions and perform tasks autonomously. They usually rely on the communication with the Remote Operation Centres (ROCs) serving as command hubs for monitoring, controlling and managing ships from the shore. ROCs serve as a pivotal hub for overseeing and controlling unmanned vessels remotely from onshore locations so that operators can utilize advanced sensor technologies and data telemetry to monitor vessel position, speed, course, environmental conditions, and other operational parameters in real-time.

The ROC is implemented following a cloud-based architecture, which is able to receive data coming from the on-board equipment (e.g. LIDARs, Radars, Cameras, IoT sensors, AIS, etc.), store and process it using AI-based models so that to provide early warnings and prevent risks in advance. The communication between the vessel and the ROC is performed exploiting existing low-latency communication technologies such as 5G NR and beyond. This scenario proposes a collaborative architecture between oneM2M-compliant platform and ETSI MEC edge computing framework to provide assisted manoeuvring services to the unmanned ships in real-time within the seaport waters.

In this architecture, schematically represented in Figure 5.4.2-1, edge-located MN-CSE instances are deployed as MEC applications on distributed edge nodes to process data coming from the vessel when it approaches the assigned berth or departs from it. On top of this, a cloud-based oneM2M IN-CSE instance is deployed as central hub for storing and managing data related to the vessel such as position, speed, course and environmental conditions. When the vessel enters the area covered by a MEC node, the cloud IN-CSE automatically offloads computational tasks (e.g. localization of other vessels) to the MN-CSE on the edge to guarantee low-latency processing capabilities for the generation of possible warnings. Using this approach, the MN-CSE can immediately analyse data coming from the vessel instead of relying on distant cloud servers.

As the vessel transits through different MASS zones, the proposed oneM2M/MEC architecture allows to seamlessly move tasks between edge nodes, ensuring service continuity and supporting mission-critical services for the unmanned vessels (e.g. assisted manoeuvring, collision avoidance and situational awareness).

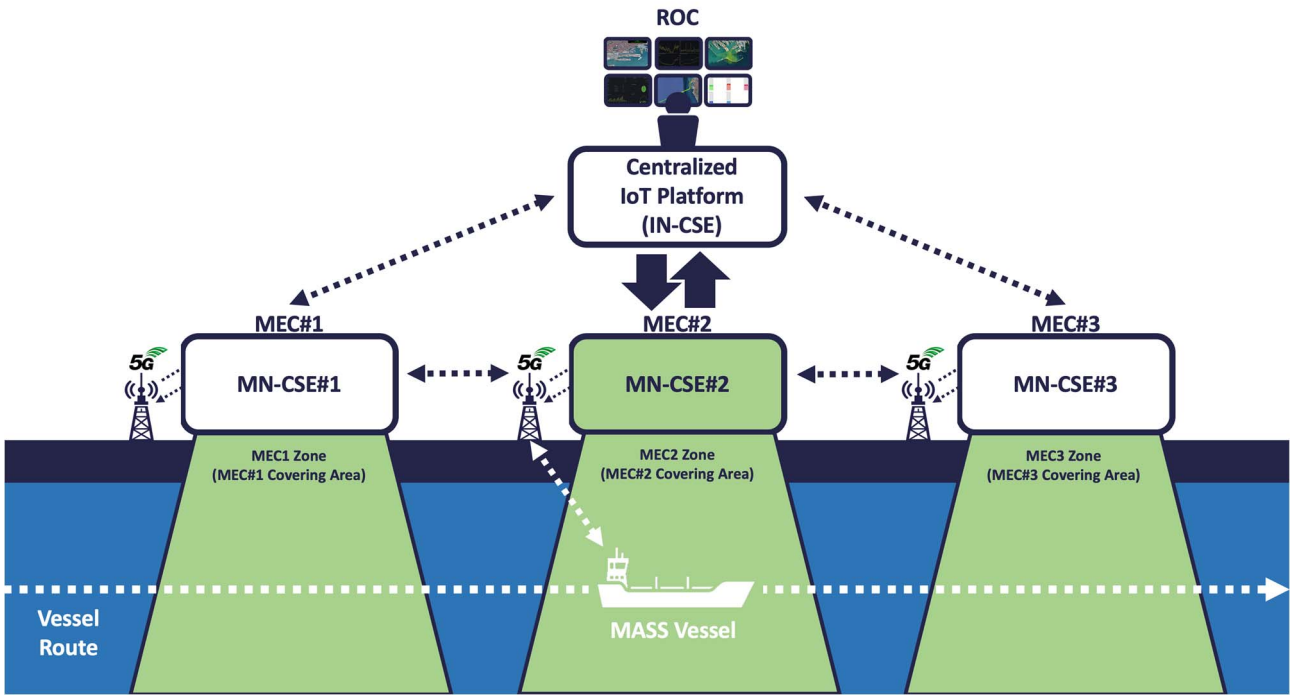

Figure 5.4.2-1: Scenario for assisted manoeuvring exploiting oneM2M and ETSI MEC

5.5 Metaverse

5.5.1 Description

The metaverse refers to a persistent, shared, and immersive digital environment where users interact with each other, digital objects, and services through avatars or virtual representations. It encompasses a range of technologies - including Virtual Reality (VR), Augmented Reality (AR), 3D web platforms, and real-time data integration - that collectively enable seamless blending of digital and physical experiences. In the context of IoT and edge computing, the metaverse can extend traditional retail, education, entertainment, and social interaction by connecting virtual spaces with live data and intelligent services from the real world. For example, a virtual shopping mall in the metaverse can mirror the inventory and layout of a physical store, allowing users to browse, interact, and purchase items as if they were present on-site. Similarly, virtual classrooms can leverage real-time sensor data and edge analytics to create adaptive, engaging learning environments. This clause introduces a use case that integrates the oneM2M IoT platform with the ETSI MEC edge computing framework to enable a smart virtual shopping service. By combining live data from IoT-enabled physical stores with edge-hosted AI analytics, the system delivers personalized, low-latency interactions within the metaverse, ensuring a synchronized, context-aware user experience that connects virtual behaviour with real-world actions.

5.5.2 Smart Shopping with Edge-AI and Cloud IoT Integration

This use case describes a smart virtual shopping service within a metaverse environment, where users can experience immersive, interactive shopping that is tightly integrated with real-world data and edge intelligence (schematically depicted in Figure 5.5.2-1). A user enters a virtual shopping mall through a metaverse interface - either via VR/AR devices or standard 3D web platforms. Inside the virtual store, the layout, available products, and store conditions reflect those of an actual physical store in real time. This synchronization is made possible through IoT devices deployed in the physical store, which are directly connected to a cloud-based oneM2M platform (IN-CSE). These devices stream data such as shelf inventory levels, product locations, environmental conditions (e.g. temperature), and user presence to the cloud IoT platform.

As the user browses virtual shelves or picks up digital items, these actions are interpreted by an application on the edge MEC platform running near the physical store. The MEC application subscribes to relevant IoT data from the IN-CSE and uses AI models to analyse both real-time device states and user behaviour. For example, if a user shows interest in a particular product - such as hovering over or virtually picking up a package - the MEC App considers stock levels, previous interaction patterns, and related product information to generate real-time recommendations ("People who looked at this also bought…") or alerts ("Only 2 left in stock!"). This feedback is sent instantly back to the metaverse interface, enhancing engagement and encouraging informed purchasing decisions. When the user confirms a purchase in the virtual store, the system triggers real-world processes: the item is marked as reserved or sold in the cloud platform (IN-CSE), and notifications are sent to logistics systems, shelf displays, or even store staff.

If needed, IoT actuators in the real store - such as digital signage or voice assistants - can respond with contextual actions (e.g. displaying pickup instructions). The entire experience feels seamless to the user: every virtual interaction is grounded in live, physical-world data, while AI-driven recommendations create a personalized, context-aware shopping journey. The user benefits from smart, immersive shopping, while the system ensures real-time linkage between the digital and physical layers, powered by oneM2M and MEC.

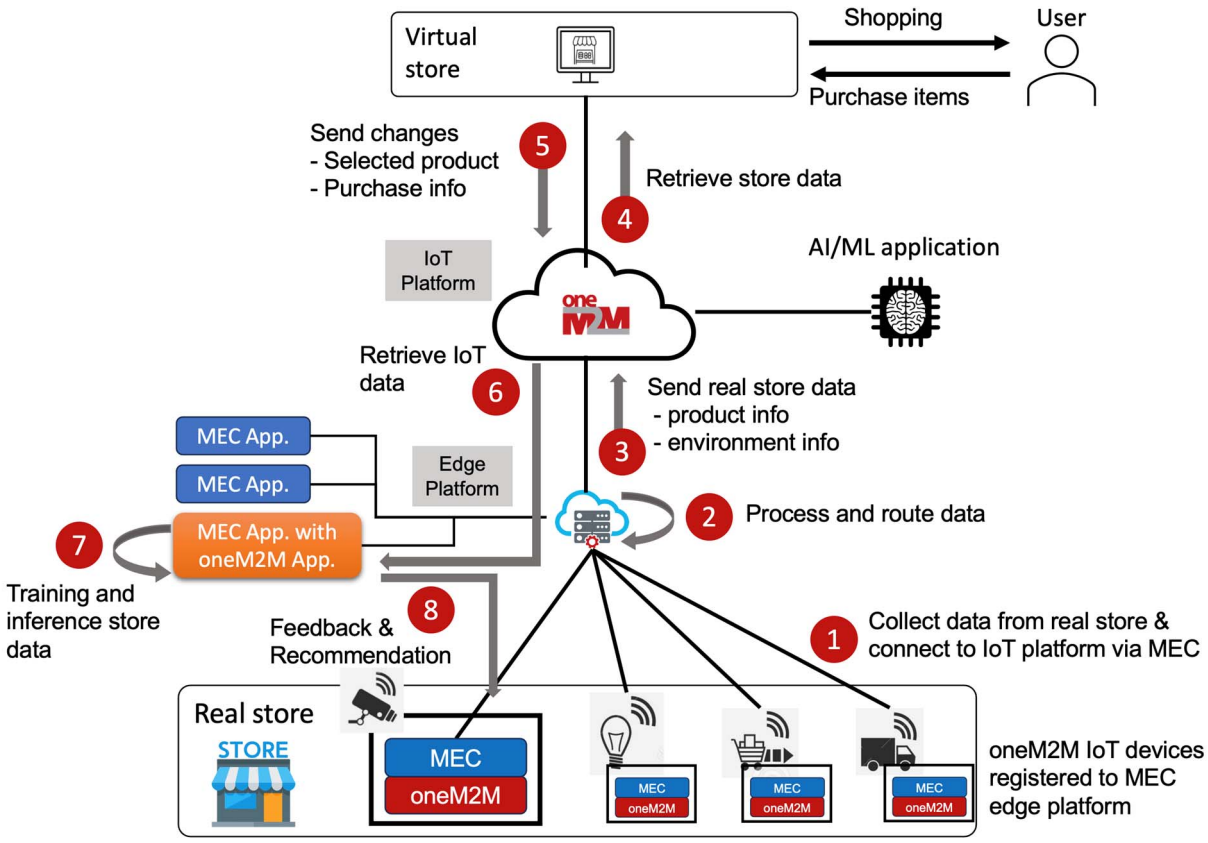

Figure 5.5.2-1: Smart Shopping with Edge-AI and Cloud IoT Integration

5.6 Future Home

5.6.1 Description

The Smart Home domain is rapidly evolving toward immersive, responsive, and personalized services powered by real-time data and intelligent automation. This clause presents a use case that integrates the oneM2M IoT platform with the ETSI MEC framework to enable advanced home experiences such as real-time media interactions, remote education, elderly care, and intelligent automation.

5.6.2 User Premises Edge and oneM2M Integration

This use case showcases the integration of ETSI MEC and oneM2M platforms to support immersive, responsive, and personalized Smart Home services. It addresses real-time media processing (e.g. holoportation), education delivery, elderly care (aging in place), and home automation. The architecture features a centralized oneM2M IN-CSE deployed on MEC node that manages smart devices across the multiple homes, and localized MN-CSEs deployed on non-MEC nodes (e.g. within an ISP gateway or smart home hub) that perform latency-sensitive tasks such as video processing, sensor fusion, and context-aware automation. If MEC supports CPEs, then MEC systems can discover them and MEP enables MEC applications to offload real time data processing tasks to this non-MEC systems.

In this extension reported in Figure 5.6.2-1, the non-MEC node is realized as a Customer Premises Edge device (have support of both computational as well as Multi-access network communication capabilities) deployed locally at the user's home (e.g. home gateway or dedicated edge server). This edge device hosts MEC applications or MN-CSE components and serves as the key enabler for local, low-latency processing of Smart Home functions, reducing dependency on centralized cloud services.

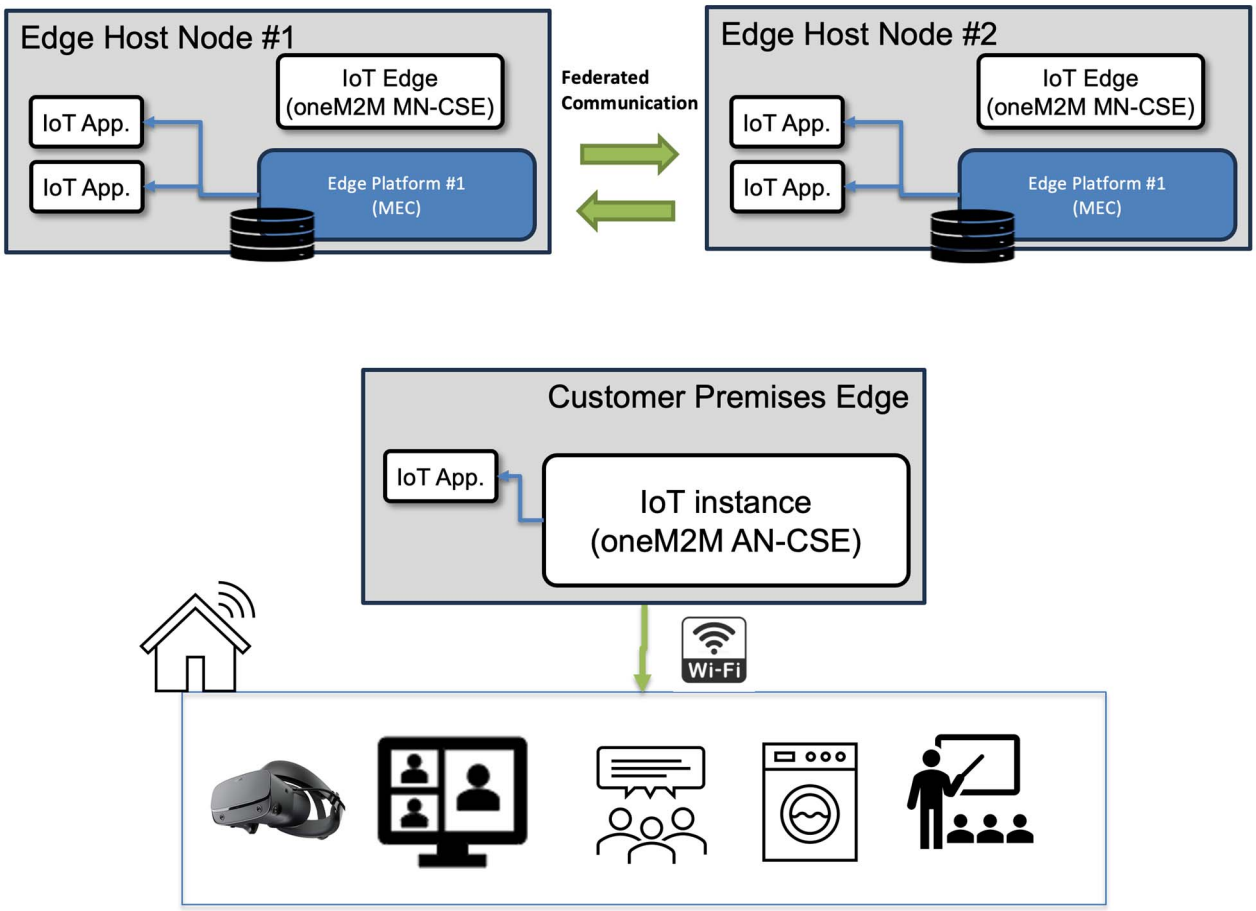

Figure 5.6.2-1: Future Home user premises scenario with integrated Edge and IoT functionalities

6 MEC-oneM2M Architectural & Use Case Mapping

6.1 Introduction

This clause provides an analysis of the main reference architectures and characteristics of MEC and oneM2M, highlighting their core functionalities and how they complement each other in the context of Edge-IoT deployments. For each of the identified use cases, the clause maps the relevant architectural elements and functionalities from both, illustrating how they contribute to the scenario. It also identifies potential new features or enhancements that could further support the use case and provides, as part of clause 7, a preliminary set of functional and technical recommendations for enabling a framework for federation and orchestration of MEC/oneM2M instances (e.g. handover, swarm computing and federated learning).

6.2 MEC Frameworks

European Telecommunications Standards Institute Industry Specification Group for Multi-access Edge Computing (ETSI ISG MEC) is the home of technical standards for edge computing. The work of the MEC initiative aims to unite the telco and IT-cloud worlds, providing IT and cloud-computing capabilities within the Radio Access Network (RAN). The MEC ISG specifies the elements that are required to enable applications to be hosted in a multi-vendor multi-access edge computing environment.

ETSI MEC framework (schematically illustrated in Figure 6.2-1) offers application developers and content providers, cloud-computing capabilities at the edge of the network, in an environment characterized by ultra-low latency and high bandwidth together with real-time access to radio network information that can be leveraged by applications.

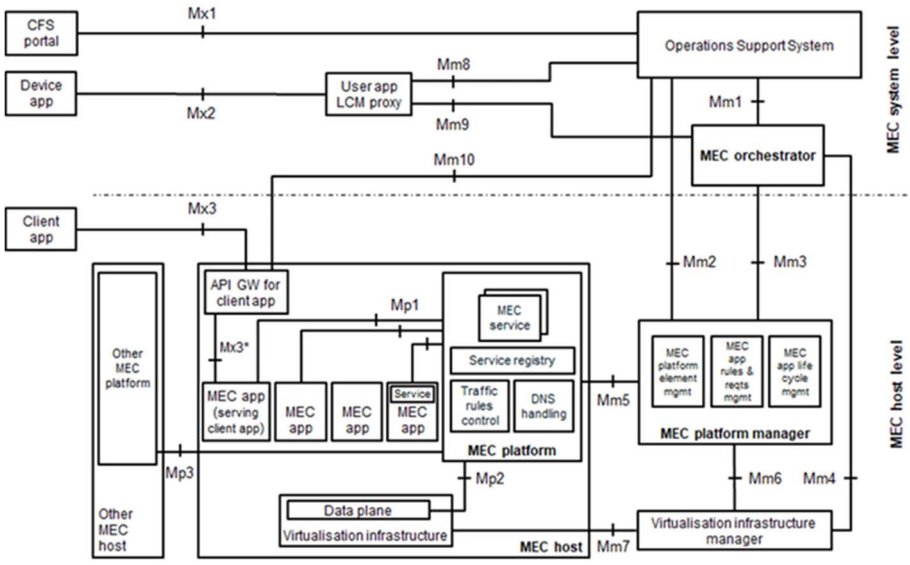

Figure 6.2-1: Multi-access edge system reference architecture

Figure 6.2-1 depicts the generic multi-access edge system reference architecture. The MEC framework defined by ETSI GS MEC 003 [i.1] includes a standardized architecture with the following key functional elements:

- MEC Orchestrator: manages resources and coordinates MEC operations.
- MEC Host: includes the MEC platform, MEC applications, and the underlying Virtualisation infrastructure.
  - MEC Platform (MEP): provides collection of essential functionalities required to run MEC applications on a particular Virtualisation infrastructure and enable them to provide and consume MEC services. The MEC platform can also provide services.
  - MEC Applications: instantiated and run on the Virtualisation infrastructure based on configuration or based on requests validated by the MEC management.
  - API GW for Clients: controls access to MEC applications by client applications.
- MEP Manager: handles the lifecycle management of the MEC platform.
- Virtualisation Infrastructure Manager (VIM): Mainly allocating, managing and releasing Virtualised (compute, storage and networking) resources of the Virtualisation infrastructure.

MEC also offers another reference architecture variant for MEC in NFV, MEC federation and Support for Security Monitoring and Management. The architecture also defines the following multiple reference points across the various provided reference architecture variants:

- Mp: connects MEC applications with the MEC platform.
- Mm: used for management operations.
- Mx: links to external systems.
- Mff: enables federation with other MEC or non-MEC systems.

To support applications running at the edge, ETSI has developed a set of standardized service 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 [i.4], when creating new service APIs. These APIs allow applications to access useful network and context information through the Mp1 interface. MEC applications (API producer) can also register new services to MEP and consume/subscribe to existing ones through this interface. Transition from MEC Phase 1 to MEC Phase 4 (period 2024-2026) provided MEC service APIs include:

- Application lifecycle, rules and requirements management (ETSI GS MEC 010-2 [i.5])
- Edge Platform Application Enablement (ETSI GS MEC 011 [i.6])

- Location API (ETSI GS MEC-013 [i.7])
- Radio Network Information API (ETSI GS MEC 012 [i.8])
- Traffic Management API (ETSI GS MEC 015 [i.9])
- Device Application Interface (ETSI GS MEC 016 [i.10])
- Application Mobility Service API (ETSI GS MEC 021 [i.11])
- WLAN Information API (ETSI GS MEC 028 [i.12])
- Fixed Access Information API (ETSI GS MEC 029 [i.13])
- V2X Information API (ETSI GS MEC 030 [i.14])
- IoT API (ETSI GS MEC 033 [i.15])
- Federation Enablement API (ETSI GS MEC 040 [i.16])
- QoS Measurement API (ETSI GS MEC 045 [i.17])
- Sensor Sharing API (ETSI GS MEC 046 [i.18])

All APIs are described using the OpenAPI Specification (OAS), ensuring consistency and ease of integration across platforms.

6.3 oneM2M Components

6.3.1 Introduction to oneM2M

oneM2M is a global standards initiative that develops specifications for a horizontal platform (or Service Layer) used for the exchange and sharing of data among Internet of Things applications. 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 provides a layer of abstraction - similar to an operating system - that shields developers from the details of network communication protocols, interfaces to IoT device and device management. It provides a framework for interworking with different technologies, for example 3GPP and Modbus. It is also designed as a distributed software layer, making it particularly suitable for use in Edge computing deployments, mitigating the burdens on data centres and core networks and improving communication latency by acquiring, processing, and storing data at the edge of network.

6.3.2 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 6.3.2-1 shows a simplified oneM2M architecture.

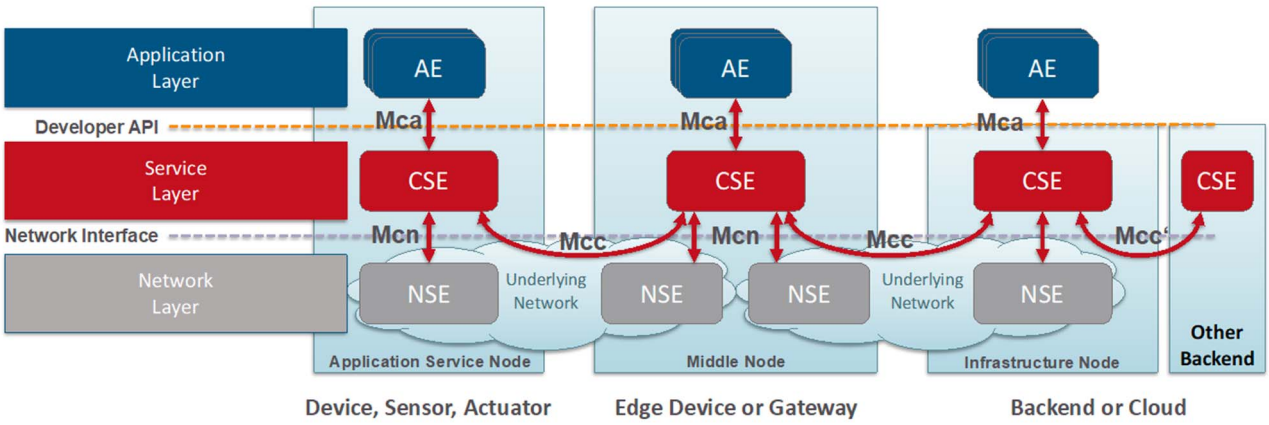

Figure 6.3.2-1: oneM2M architecture

This architecture is composed of the following entities:

- *Common Services Entity (CSE)*: Provides the set of "service functions" that are common to multiple IoT domains.
- *Application Entity (AE)*: Represents application logic for the end-to-end IoT solutions. This includes device firmware, gateway logic or backend applications.
- *Interworking Proxy application Entity (IPE)*: A specialized AE that is used to interface between oneM2M and an external system.
- *Network Services Entity*: Provides networking services to CSEs that are more than just data transport.
- *Node*: Logical equivalent of a physical (or possibly virtualised) server or device.

oneM2M defines several different kinds of node, which are named based on their use by oneM2M.

- *Infrastructure Node (IN)*: A Cloud provider / company server. It hosts a CSE that is referred to as an IN-CSE and may host AEs. There is exactly one IN in the Infrastructure Domain of each oneM2M Service Provider.
- *Application Service Node (ASN)*: A Node that contains at least one AE co-located with a CSE (an ASN-CSE).
- *Middle Node (MN)*: A Node that contains one CSE (an MN-CSE) and zero or more AEs. It is typically found in an Edge Server or Gateway box.
- *Application Dedicated Node (ADN)*: A Node that contains at least one AE but does not contain a CSE.

In addition, the oneM2M architecture defines one other type of node, the so-called "Non-oneM2M Node" (NoDN), not shown in Figure 6.3.2-1. This is a node 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.

The principal *Reference Points* defined by oneM2M are:

- *Mca*: The interface used by an AE to use the common services provided by the CSE.
- *Mcc*: Communication flows across the Mcc reference point enable a CSE to use the services supported by another CSE.
- *Mcc'*: Communication flows between CSEs that belong to different oneM2M service providers.
- *Mcn*: Communication flows between a Common Services Entity (CSE) and the Network Services Entity (NSE) allowing the CSE to use the network services (other than transport and connectivity) provided by the NSE.

6.3.3 oneM2M Architecture Common Service Layer Functions

From a functional perspective, oneM2M has defined fourteen Common Service Functions (CSFs) as shown in Figure 6.3.3-1 below. These relate to network connectivity, device security, transport protocols, content serialization, IoT device services and management and IoT semantic ontologies. A given CSE does not have to implement every CSF and CSE's can communicate with each other giving an AE or CSE the ability to access oneM2M resources that are hosted on a remote CSE.

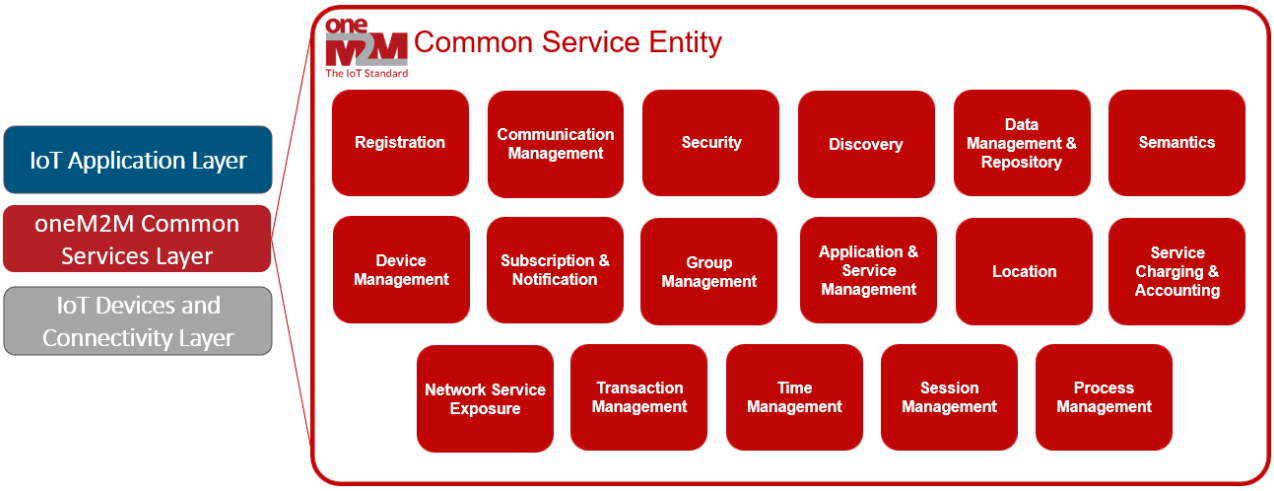

Figure 6.3.3-1: Common Service Functions

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 Applications REFerence (SAREF) or W3C's Thing Description.

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 designed 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.

6.3.4 Mapping between MEC and oneM2M

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, it is possible to consider the following mapping of oneM2M entities with ETSI MEC entities:

- A common service function (e.g. Network Service Function) or CSE in the 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 (AEs).
- Similarly, an AE in the oneM2M architecture can be implemented as a *MEC Application* in an ETSI MEC system.

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 and by providing MEC-specific IPE's. 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).

6.3.5 Virtualisation of oneM2M Common Service Layer functions

Network Virtualisation allows IT service providers to develop an end-to-end digital service based on Network as a Service (NaaS). Network Function Virtualisation (NFV) targeted the oneM2M common service layer functions where the Virtualisation 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 Virtualisation infrastructure, VNFs of oneM2M service functions can be deployed anywhere in the network on-demand. NFV enables to slice a oneM2M common service layer in the form of multiple virtual IoT CSFs at the network edge. To achieve this, Virtualisation 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.

6.4 Use Cases & Frameworks Mapping

6.4.1 Introduction to Use Cases & Frameworks Mapping

This clause provides a synthesis of the previously identified use cases in clause 5 and the architectural features of the MEC and oneM2M frameworks. It maps each use case to the relevant framework components and functionalities, showcasing how MEC and oneM2M support specific operational needs. The mapping highlights the alignment between the frameworks and the identified scenarios, while also pointing out areas where enhancements or additional integration mechanisms may be required to better support cross-domain use cases. The selection of the most appropriate deployment option for each use case is guided by insights from the ETSI white paper [i.2], where 4 deployment options are mentioned for interworking of oneM2M and MEC technologies.

6.4.2 Deployment Options

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

This option captures the MEC and oneM2M standard as they exist at the start of the effort to integrate both standards with only changes to support MEC devices and oneM2M devices (see Figure 6.4.2.1-1). This can be done by developing applications that use the oneM2M Mca reference point and the MEC Mp1 reference point:

- A oneM2M IN-CSE is deployed in the cloud and an IN-AE represents the application that is connected to the IN-CSE.
- The edge gateway connects IoT devices to the MEC framework to provide services to the devices.
- The MEC gateway is connected to the IN-CSE through an MEC IPE. The IN-CSE represents the MEC platform and devices that are registered to the MEC platform using the MEC IPE:
  - devices may use MEC services and APIs (Mp1) with support from the MEC IPE hosted as an MEC APP;
  - devices may use non-MEC services, e.g. UPC-UA devices, using a 3rd party IPE hosted as an MEC APP;
  - devices may use oneM2M services through a Mca proxy. This Mca proxy can be hosted as an MEC APP. These oneM2M devices can also take advantage of exposed MEC services.

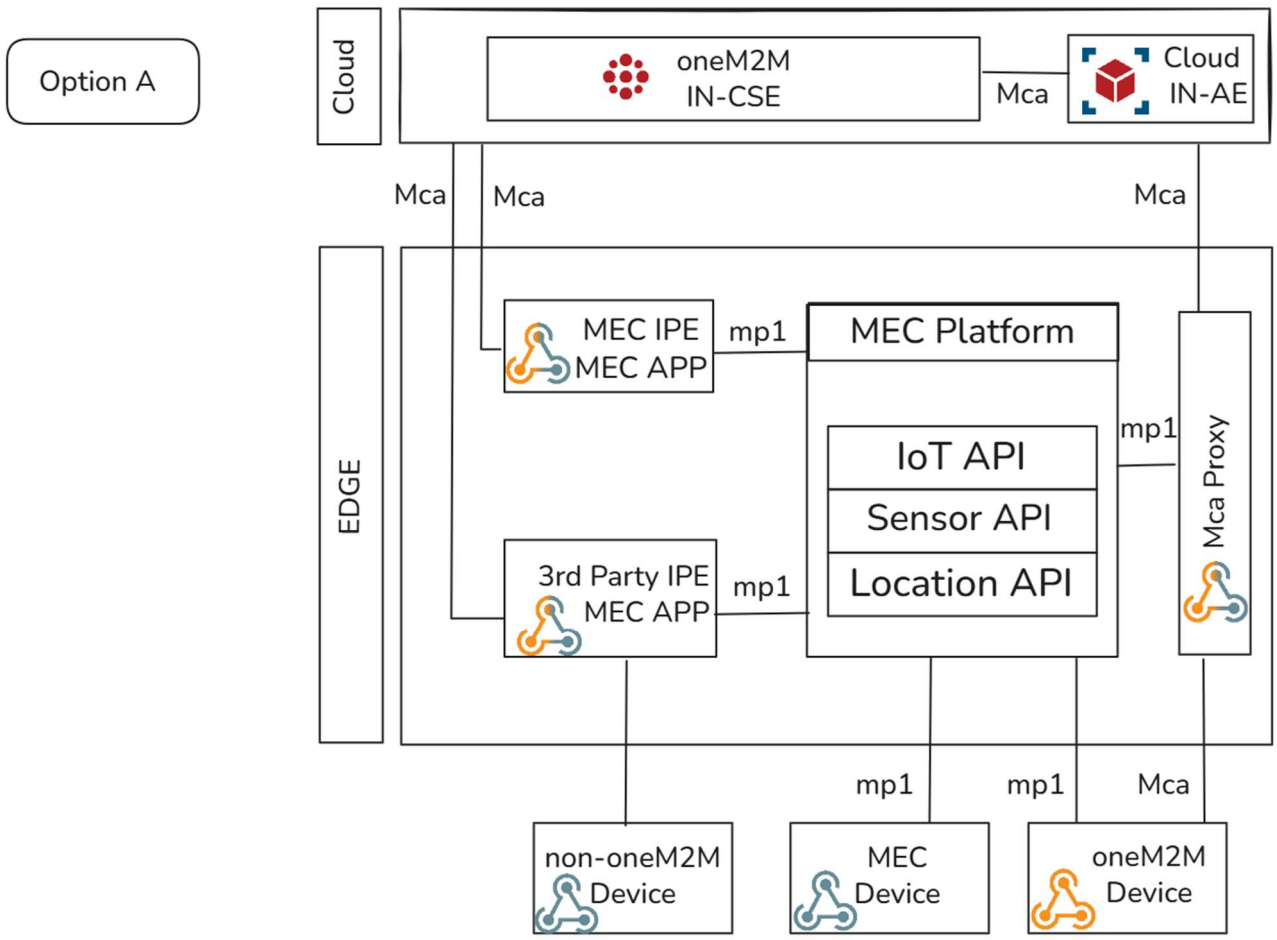

Figure 6.4.2.1-1: Deployment Option A

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:

- A standard document providing interoperability and interworking between the two platforms, oneM2M and ETSI MEC, is needed.

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

This option illustrated in Figure 6.4.2.2-1 enhances the performance of the edge deployment by introducing additional services through the integration of a oneM2M MN-CSE. A oneM2M IPE is added that interworks MEC services and devices. This can be realized using an exemplar setup consisting of one platform with MEC framework and a second platform hosting the MN-CSE:

- A oneM2M IN-CSE is deployed in the cloud and an IN-AE represents the application that is connected to the IN-CSE.
- The edge consists of the MEC platform that connects devices using MEC.
- The edge also includes a oneM2M MN-CSE that offers services to oneM2M devices and other types of devices:
  - The MEC IPE is deployed with the oneM2M platform or the MEC platform.
  - Additional services can be exposed by oneM2M AEs as such interworking with other devices, swarm computing and federated learning.
  - These additional oneM2M AEs can be deployed on either the oneM2M platform or the MEC platform.

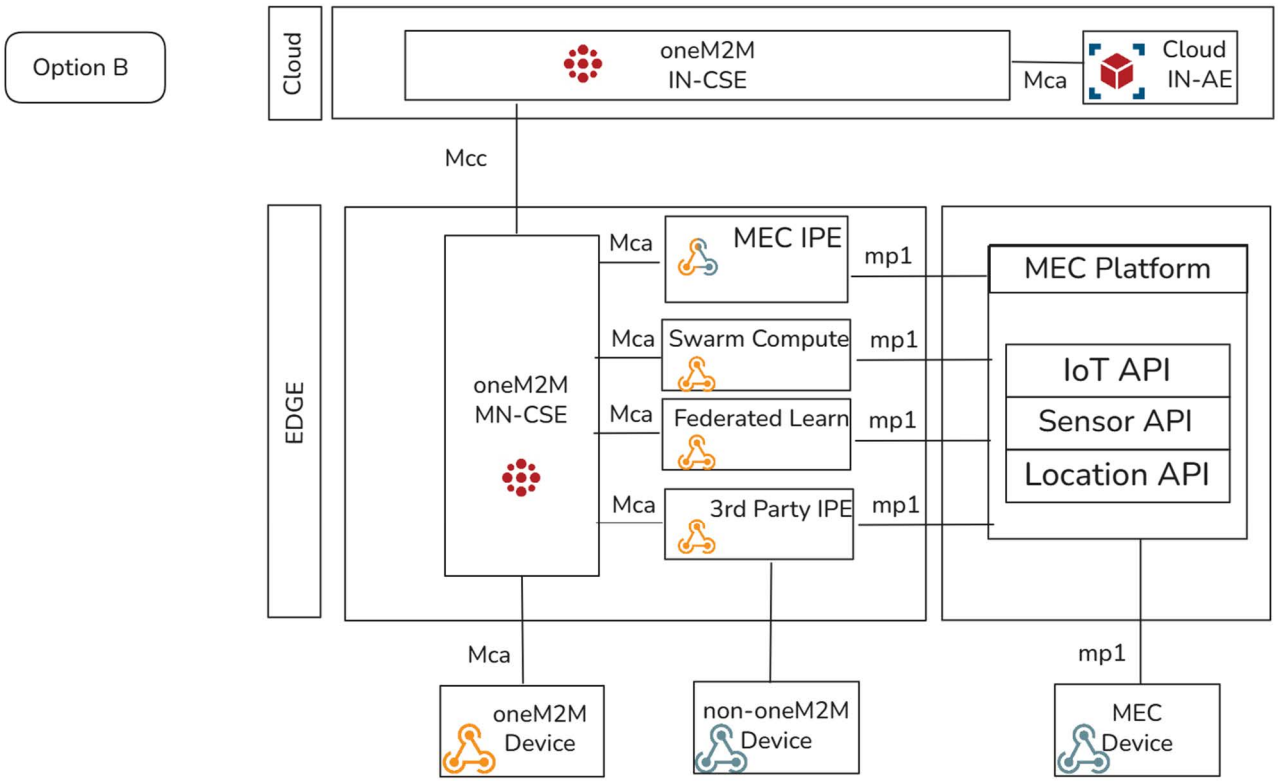

Figure 6.4.2.2-1: Deployment Option B

This deployment uses oneM2M IoT service providers and ETSI MEC entities that may be different, this scenario can still be used in the early edge computing market by using the same custom MEC IPE that was defined in option A. Additionally, new capabilities from oneM2M can be deployed to support compute intensive services:

- A standard document providing interoperability and interworking between the two platforms, oneM2M and ETSI MEC, is needed.

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

This option focuses on integrating the services of oneM2M and ETSI MEC as reported in Figure 6.4.2.3-1. This achieves better performance by combining the oneM2M MN-CSE and the MEC framework. The MEC IPE is no longer needed because the oneM2M and MEC framework are integrated:

- A oneM2M IN-CSE is deployed in the cloud and an IN-AE represents the application that is connected to the IN-CSE.
- The edge consists of the MEC platform that connects devices using MEC apis and the oneM2M MN-CSE to support a variety to devices.
- The MEC IPE is replaced with new APIs that take advantage of new standards work from ETSI MEC and oneM2M.

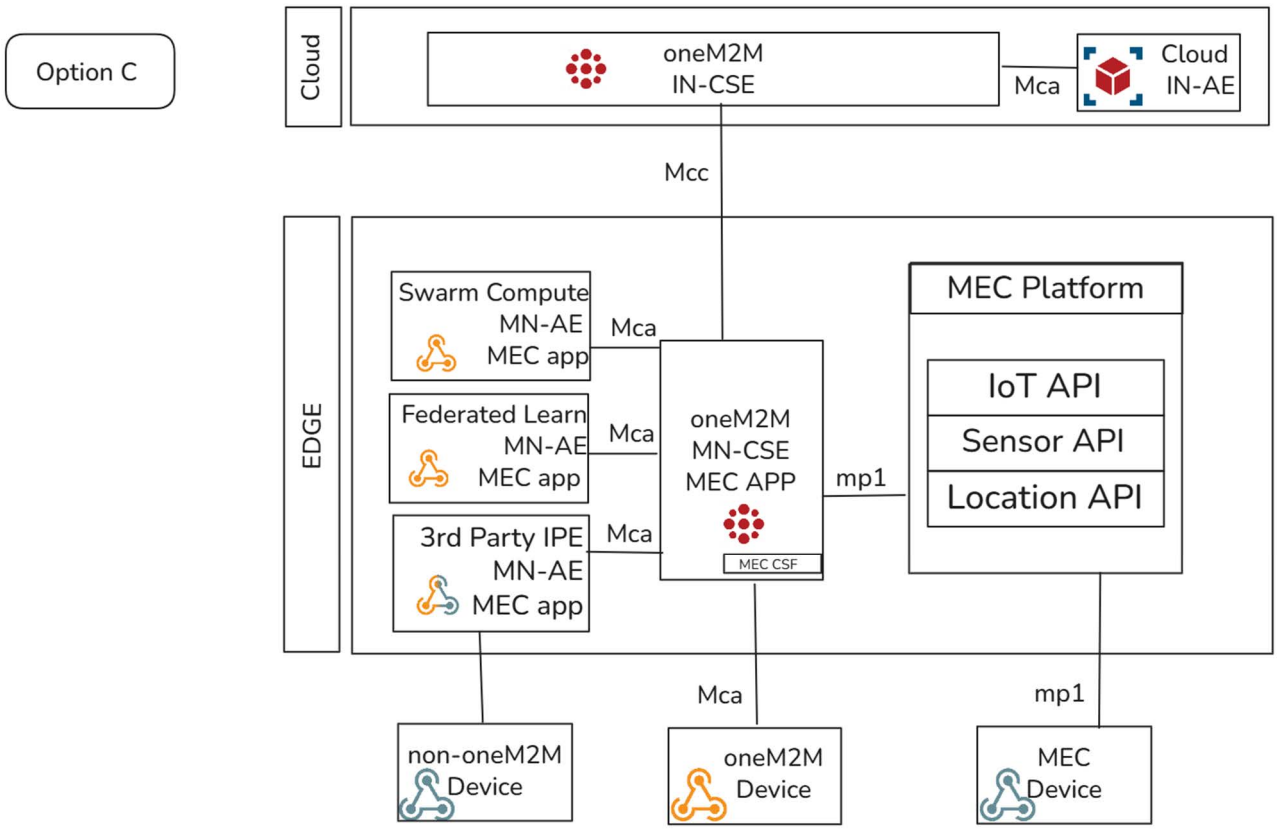

Figure 6.4.2.3-1: Deployment Option C

In this scenario, oneM2M and MEC platforms are installed and operated on the same physical edge node. This can significantly improve services by eliminating unnecessary data movement and information exchange. It also supports customizing the system to take advantage of high compute platform capabilities that may be needed for federated learning or swarm computing:

- A standard document providing interoperability and interworking between the two platforms, oneM2M and ETSI MEC, is needed.

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

This option attempts to simplify the services offered by ETSI MEC and oneM2M by fully integrating the services as schematically illustrated in Figure 6.4.2.4-1:

- A oneM2M IN-CSE is deployed in the cloud and an IN-AE represents the application that is connected to the IN-CSE.
- The edge consists of the MEC platform that connects devices using MEC and the oneM2M MN-CSE to support a variety of devices.
- MEC exposes APIs that integrate the oneM2M services.

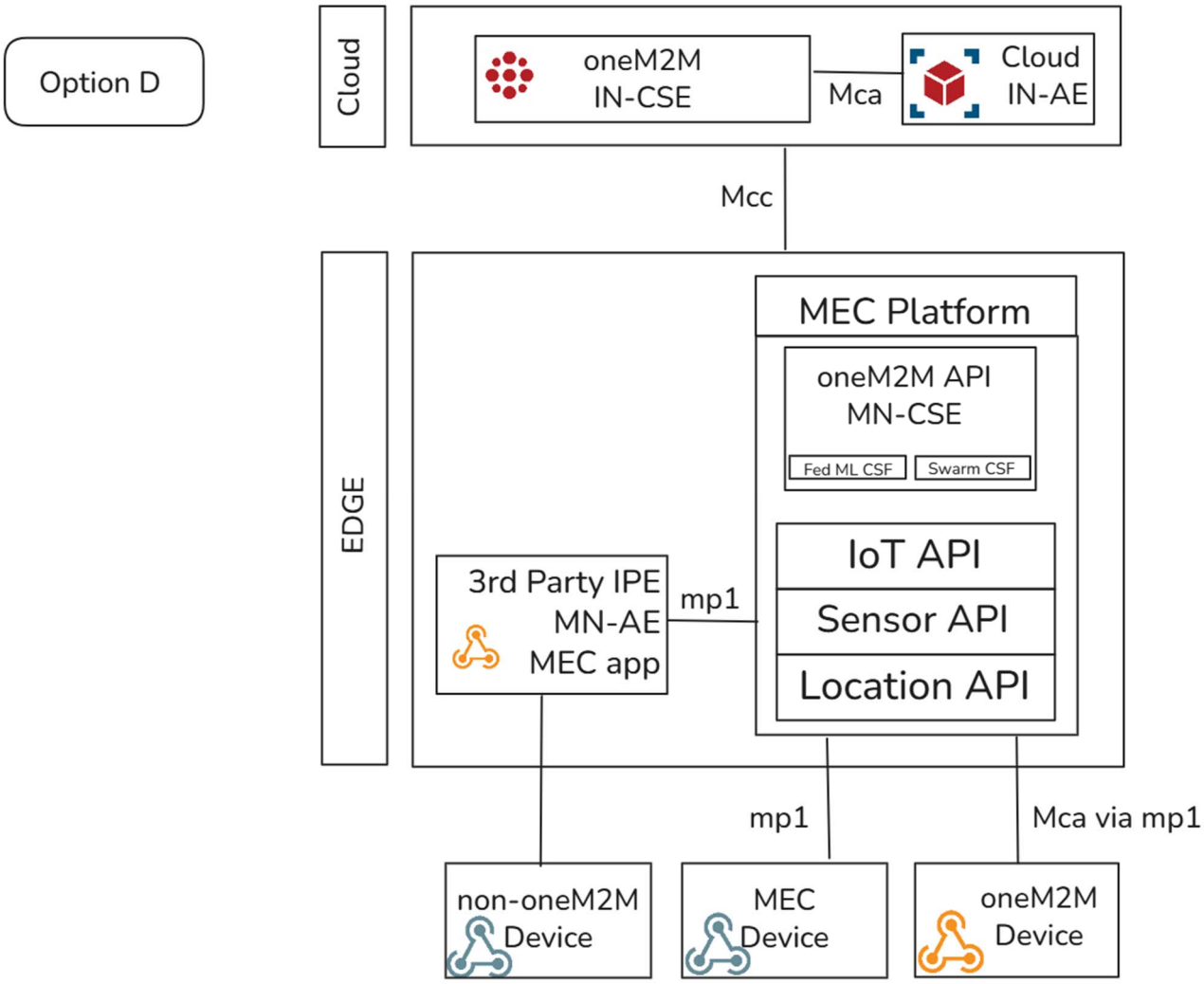

Figure 6.4.2.4-1: Deployment Option D

In this scenario oneM2M integrates to MEC and offers services that support directly providing data source, processing, and multi-access networking:

- A standard document providing interoperability and interworking between the two platforms, oneM2M and ETSI MEC, is needed.

6.4.3 Autonomous Vehicle with Continuous Edge Computing

6.4.3.1 Use Case Driving Deployment

For the "Autonomous Vehicle with Continuous Edge Computing" use case, achieving ultra-low latency, context awareness, and localized processing is essential to handle data from nearby sources and enable real-time analytics. To meet these requirements, certain tasks can be offloaded to oneM2M Edge Instances (MN-CSEs). As autonomous vehicles move across MEC zones, orchestration and synchronization across multiple oneM2M platforms is needed. For this, a centralized IN-CSE deployed in the cloud is suitable to maintain global context, coordinate edge instances, and ensure seamless service continuity.

In Table 6.4.3.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.3.1-1: Operational Requirements and Platform Support for Autonomous Vehicle with Continuous Edge Computing

Operational Requirement	Support in MEC	Support in oneM2M
Up and running IoT platform	Not Required	Supported: oneM2M platform can be deployed at cloud or any edge node
Registration of vehicles, IoT devices, and applications	Not Required	Supported: oneM2M features provide Common Service Functions (CSFs) to register both devices and Application Entities (AEs)
Data collection from IoT Devices (Vehicle sensors, Road sensors, LIDAR)	Not Required	Supported via oneM2M's Common Service Layer using Mca interfaces for data exchange
Instantiate of MN-CSE on Edge Node	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiation of AE on Edge Node	AE can be instantiated as MEC Application on MEC Host.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application

6.4.4 Vulnerable Road Users

6.4.4.1 Use Case Driving Deployment

The "Vulnerable Road Users" (VRU) use case shares deployment characteristics with the "Autonomous Vehicle with Continuous Edge Computing" scenario, where achieving ultra-low latency, localized processing, and context awareness is critical. This is necessary to effectively process data from nearby roadside units, sensors mounted on traffic signals, and other urban infrastructure to enable real-time analytics and immediate response. To support this, specific processing tasks can be offloaded to oneM2M Edge Instances (MN-CSEs) operating close to the data sources. As vehicles and VRUs dynamically move across different geographic zones, coordination across multiple MEC nodes becomes essential. Therefore, a centralized IN-CSE deployed in the cloud is ideal for maintaining global context, orchestrating and synchronizing edge instances, and ensuring uninterrupted service continuity. Meanwhile, MN-CSEs should be deployed close to the edge to provide low-latency decision support

In Table 6.4.4.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.4.1-1: Operational Requirements and Platform Support for Vulnerable Road Users

Operational Requirement	Support in MEC	Support in oneM2M
Up and running IoT platform	Not Required	Supported: oneM2M platform can be deployed at cloud or any edge node
Registration of vehicles, IoT devices, and applications	Not Required	Supported: oneM2M features provide CSFs to register both devices and Application Entities (AEs)
Data collection from IoT Devices (Vehicle sensors, Road sensors, LIDAR)	Not Required	Supported via oneM2M's Common Service Layer using Mca interfaces for data exchange
Instantiate of MN-CSE on Edge Node	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiation of AE on Edge Node	AE can be instantiated as MEC Application on MEC Host.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application

6.4.5 Swarm-based Autonomous Ant Delivery Optimization

6.4.5.1 Use Case Driving Deployment

The "Swarm-based Autonomous Ant Delivery Optimization" use case enables optimized coordination of autonomous delivery robots using distributed intelligence and real-time data exchange. The oneM2M IN-CSE is deployed in the cloud to manage global swarm state, while MN-CSEs are instantiated at the edge to handle local swarm behaviour. Swarm applications (e.g. route optimization, obstacle detection) are deployed on edge nodes and interact with MN-CSEs for low-latency data processing. Robots continuously report environmental data, pheromone map updates, and coordination feedback to the platform. As robots move across delivery zones, the system ensures seamless session handover and synchronization of swarm intelligence using MN-CSE coordination. This setup supports dynamic optimization, real-time feedback, and resilient autonomous operations as schematically reported in Table 6.4.5.1-1.

Table 6.4.5.1-1: Operational Requirements and Platform Support for Swarm-based Autonomous Ant Delivery Optimization

Operational Requirement	Support in MEC	Support in oneM2M
Up and running IoT platform	Not directly required	Supported: oneM2M IN-CSE can be deployed at cloud
Registration of devices (Swarm Robots)	Not required	Supported: oneM2M enables registration of sensors, actuators, controllers as either IN-AEs or ADN-AE via standardized resource structures and CSFs
Registration of applications	Not required	Supported: oneM2M enables registration of Applications as AEs via standardized resource structures and CSFs
Real-time data ingestion from AEs (pheromone map updates, route feedback and obstacle reports)	Not required	Supported via oneM2M's CSF and Mca interfaces
Instantiate of MN-CSE on Edge Node	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiate of AE (e.g. Swarm optimization) on Edge Node	AE can be instantiated as MEC Application on MEC Host.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application
Low-latency control for Hybrid Swarm Robots (feedback for Swarm optimization)	Supported: MEC Apps handle time-sensitive operations with minimal delay.	Supported: MN-CSE enables fast coordination through subscriptions, notifications, and data routing
Offloading low latency tasks of IoT Platform for real time data processing	Not required. But MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: Tasks can be offloaded to MN-CSE instances using Mcc interface mechanisms.
Service continuity during Hybrid Swarm Robot zone transitions	Supported via ETSI GS MEC 013 [i.7] Location API (tracks UE movement) and ETSI GS MEC 040 [i.16] (supports MEC Federation and cross-MEP orchestration).	Supported: oneM2M handles session handover and task migration to a new edge (MN-CSE) instance coordinated by IN-CSE

6.4.6 Smart Warehouse Automation

6.4.6.1 Use case Driven Deployment

The architecture integrates a centralized oneM2M IN-CSE, responsible for managing infrastructure-level data (sensors, tags, operational policies), and multiple MN-CSEs deployed at the MEC edge, enabling real-time, latency-sensitive decision-making close to the action. For time-critical events such as rerouting AGVs, responding to hazards, or optimizing resource allocation, processing tasks are offloaded to MEC-hosted MN-CSEs. These instances provide ultra-low-latency data processing. In Table 6.4.6.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.6.1-1: Operational Requirements and Platform Support for Smart Warehouse Automation

Operational Requirement	Support in MEC	Support in oneM2M
Warehouse IoT platform deployment	Not directly required	Supported: oneM2M IN-CSE can be deployed at cloud or control centre
Registration of sensors, AGVs, and applications	Not required	Supported: oneM2M features provide CSFs to register both devices and Application Entities (AEs)
Collection of environmental and telemetry data	Not required	Supported via oneM2M's Common Service Layer and Mca interfaces
Instantiate of MN-CSE on Edge Node	MEC provides the infrastructure to host oneM2M as a service producing MEC application (using Mp1 interface or as a new MEC service inside the MEC Platform). MEC IoT API (ETSI GS MEC 033 [i.15]) enables registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiate of AE (e.g. AGV controller, sensor manager) on Edge Node	AE can be instantiated as MEC Application on MEC Host.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application
Low-latency control for AGVs and asset tracking	Supported: MEC apps handle time-sensitive operations with minimal delay.	Supported: MN-CSE enables fast coordination through subscriptions, notifications, and data routing
Offloading low latency tasks of IoT Platform for real time data processing	Not Required	Supported: Tasks can be offloaded to MN-CSE instances using Mcc interface mechanisms.
Service continuity during AGV zone transitions	Supported via ETSI GS MEC 013 [i.7] Location API (tracks UE movement) and ETSI GS MEC 040 [i.16] (supports MEC Federation and cross-MEP orchestration).	Supported: oneM2M handles session handover and task migration to a new edge (MN-CSE) instance coordinated by IN-CSE

6.4.7 Industrial Digital Twins

6.4.7.1 Use case Driven Deployment

The "Industrial Digital Twin" use case requires tight integration between oneM2M and MEC to support low-latency analytics, real-time control, and dynamic migration of digital twin services across production zones. The IN-CSE is deployed in the cloud to manage global digital twin state and analytics, while MN-CSEs are instantiated on MEC nodes located near industrial assets. MEC hosts edge applications (e.g. quality inspection, anomaly detection) that interact with MN-CSEs. As mobile assets like AGVs move, the MEC, MN-CSE and IN-CSE coordinate handovers and synchronize twin states across MN-CSEs, ensuring seamless operation and minimal downtime.

In Table 6.4.7.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.7.1-1: Operational Requirements and Platform Support for Industrial Digital Twins

Operational Requirement	Support in MEC	Support in oneM2M
IoT platform deployment	Not directly required	Supported: oneM2M IN-CSE can be deployed at cloud or control centre
Registration of Industrial devices (Digital Twin (DT))	Not required	Supported: oneM2M enables registration of industrial sensors, actuators, controllers as either IN-AEs or ADN-AE via standardized resource structures and CSFs using <flexContainer> data
Registration of Industrial applications	Not required	Supported: oneM2M enables registration of industrial
Applications as AEs via standardized resource structures and CSFs		
Real-time data ingestion from factory floor (sensors, robots, AGVs)	Not required	Supported via oneM2M's CSF and Mca interfaces
Instantiate of MN-CSE on Edge Node	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiate of AE (e.g. AGV controller, sensor manager, analytics & controller) on Edge Node (Edge Digital Twin)	AE can be instantiated as MEC Application on MEC Host	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application
Low-latency control for AGVs and asset tracking.	Supported: MEC Apps handle time-sensitive operations with minimal delay.	Supported: MN-CSE enables fast coordination through subscriptions, notifications, and data routing
Offloading low latency tasks of IoT Platform for real time data processing	Not required. But MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: Tasks can be offloaded to MN-CSE instances using Mcc interface mechanisms.
Service continuity during AGV zone transitions	Supported via ETSI GS MEC 013 [i.7] Location API (tracks UE movement) and ETSI GS MEC 040 [i.16] (supports MEC Federation and cross-MEP orchestration).	Supported: oneM2M handles session handover and task migration to a new edge (MN-CSE) instance coordinated by IN-CSE
Synchronization of Digital Twin state between edge and cloud	Not required	oneM2M ensures data consistency and context synchronization between MN-CSE (edge) and IN-CSE (cloud) whereas DT state plays vital role

6.4.8 Assisted Manoeuvring for Autonomous Ships

6.4.8.1 Use case Driven Deployment

The "Assisted Manoeuvring for Autonomous Ships" use case supports autonomous ship operations through coordinated deployment of oneM2M and MEC components. A cloud-hosted IN-CSE collects and processes global vessel data (e.g. location, speed, sensor feeds), while edge-hosted MN-CSEs (ROC) are deployed near ports to assist with real-time decision-making. When a ship enters port waters, MEC hosts localized MEC/Edge applications (e.g. assisted manoeuvring, collision avoidance) that interact with MN-CSEs for low-latency analytics. As the vessel moves, tasks are seamlessly offloaded across MN-CSEs, with the IN-CSE ensuring synchronization and service continuity. This deployment enables responsive control, enhanced safety, and uninterrupted autonomous operation of unmanned ships near coastal and harbour zones.

In Table 6.4.8.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.8.1-1: Operational Requirements and Platform Support for Assisted Manoeuvring for Autonomous Ships

Operational Requirement	Support in MEC	Support in oneM2M
IoT platform deployment	Not directly required	Supported: oneM2M IN-CSE can be deployed at cloud
Registration of devices on the Ships	Not required	Supported: oneM2M enables registration of sensors, actuators, controllers as either IN-AEs or MN-AE or ADN-AE via standardized resource structures and CSFs
Registration of applications	Not required	Supported: oneM2M enables registration of Applications as AEs via standardized resource structures and CSFs
Real-time data ingestion from AEs (vessel localization, its speed, course and other environment variables)	Not required	Supported via oneM2M's CSF and Mca interfaces
Instantiate of MN-CSE on Edge Node plays role ROC with AI capabilities	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiate of AE (e.g. generate warnings in real-time and optimize its transmission to the vessel) on Edge Node	AE can be instantiated as MEC Application on MEC Host.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application
Low-latency control for unmanned Vessels (i.e. assisted manoeuvring, collision avoidance and situational awareness)	Supported: MEC Apps handle time-sensitive operations with minimal delay.	Supported: MN-CSE enables fast coordination through subscriptions, notifications, and data routing
Offloading low latency tasks of IoT Platform for real time data processing	Not required. But MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: Tasks can be offloaded to MN-CSE instances using Mcc interface mechanisms.
Service continuity during vessels transitions	Supported via ETSI GS MEC 013 [i.7] Location API (tracks UE movement) and ETSI GS MEC 040 [i.16] (supports MEC Federation and cross-MEP orchestration).	Supported: oneM2M handles session handover and task migration to a new edge (MN-CSE) instance coordinated by IN-CSE

6.4.9 Smart Metaverse Shopping with Edge-AI and Cloud-IoT Integration

6.4.9.1 Use case Driven Deployment

The "Smart Metaverse Shopping" use case requires the oneM2M IN-CSE deployment in the cloud to manage IoT devices and to store data from the physical retail environment. MEC hosts an AI-powered application at the network edge, close to the store, enabling real-time interpretation of user interactions within the metaverse. The MEC App subscribes to live data from the IN-CSE and processes it to deliver immediate, personalized feedback in the virtual shopping environment (e.g. recommendations, stock alerts). Upon purchase, the system updates the IN-CSE and triggers real-world actions such as product reservation or notifications to store staff and actuators. Optionally, an MN-CSE may be instantiated on the MEC platform to further reduce latency and improve service continuity. This hybrid deployment ensures real-time, synchronized experiences across the digital and physical retail layers.

In Table 6.4.9.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.9.1-1: Operational Requirements and Platform Support for Smart Metaverse Shopping

Operational Requirement	Support in MEC	Support in oneM2M
IoT platform deployment	Not directly	Supported: oneM2M IN-CSE can be deployed at cloud
Registration of IoT devices deployed in the physical store	Not required	Supported: oneM2M enables registration of industrial sensors, actuators, controllers as either IN-AEs or ADN-AE via standardized resource structures and CSFs
Registration of IoT applications	Not required	Supported: oneM2M enables registration of industrial Applications as AEs via standardized resource structures and CSFs
Real-time data ingestion from Stores (shelf inventory levels, product locations, environmental conditions)	Not required	Supported via oneM2M's CSF and Mca interfaces
Instantiate of MN-CSE on Edge Node with AI capabilities (If required)	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiate of AE (e.g. sensor manager, analytics & controller) on Edge Node	AE can be instantiated as MEC Application on MEC Host. It can use AI models for analysing real time data and providing feedback. Virtual Store App instantiate as an MEC application/Edge Application on MEC Host.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application
Low-latency control for real time data processing	Supported: MEC Apps handle time-sensitive operations with minimal delay.	Supported: MN-CSE enables fast coordination through subscriptions, notifications, and data routing

6.4.10 Future Homes

6.4.10.1 Use case Driven Deployment

The "Future Homes" use case delivers immersive and personalized smart home services through ETSI MEC and oneM2M integration. A centralized CSE (IN-CSE or MN-CSE) runs on a Cloud or MEC node to manage devices across homes, while local CSEs (MN-CSE/ASN-CSE) are deployed on non-MEC nodes such as home gateways or CPEs. Latency-critical tasks like video analytics and sensor fusion are offloaded to Customer Premises Edge devices hosting MEC apps or CSEs, with MEC platforms leveraging the MEP interface for discovery and task delegation.

In Table 6.4.10.1-1, all the relevant operational requirements for this use case are considered.

Table 6.4.10.1-1: Operational Requirements and Platform Support for Future Homes

Operational Requirement	Support in MEC	Support in oneM2M
IoT platform deployment	Not directly	Supported: oneM2M IN-CSE can be deployed at cloud or as MN-CSE elsewhere
Registration of devices at the home premises (lights, thermostats, cameras, wearables)	Not required	Supported: oneM2M enables registration of sensors, actuators, controllers as either IN-AEs or MN-AE or ADN-AE via standardized resource structures and CSFs
Registration of applications	Not required	Supported: oneM2M enables registration of Applications as AEs via standardized resource structures and CSFs
Real-time data ingestion from AEs (location data, Health alerts or anomalies, room temperature, energy consumption, object movements)	Not required	Supported via oneM2M's CSF and Mca interfaces
Instantiate of MN-CSE/ASN-CSE on Edge Node (Customer Premises Edge)	MEC provides the infrastructure to host oneM2M as a service producing MEC application using Mp1 interface. Can provide support to integrate a new MEC IoT service inside the MEC Platform which should be more coupled with oneM2M standards. Whereas MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms. Not Supported in MEC: If MEC supports CPEs, then MEC systems can discover them, and MEP enables MEC applications to offload real time data processing tasks to this non-MEC systems.	Supported: oneM2M platform needs to include a CSF to integrate with the MEC platform
Instantiate of AE (e.g. Energy Optimization, Real-time Safety Alerts, Contextual Scene Adaptation, Manage home appliances) on Edge Node	AE can be instantiated as MEC Application on MEC Host or as Edge application on CPEs.	Supported: oneM2M platform needs to include a CSF to instantiate an AE as MEC application
Offloading low latency tasks of IoT Platform for real time data processing	Not required. But MEC IoT API (ETSI GS MEC 033 [i.15]) enables minimal registration and discovery of IoT platforms.	Supported: Tasks can be offloaded to MN-CSE instances using Mcc interface mechanisms.
Service continuity during devices movements.	Supported via ETSI GS MEC 013 [i.7] Location API (tracks UE movement), ETSI GS MEC 021 [i.11] (Application Mobility service) and ETSI GS MEC 040 [i.16] (supports MEC Federation and cross-MEP orchestration).	Supported: oneM2M handles session handover and task migration to a new edge (MN-CSE/ASN-CSE) instance coordinated by IN-CSE/MN-CSE

7 Proposed Recommendations for Federation and Orchestration Mechanisms

7.1 Introduction

The main focus of integration between **ETSI MEC** and **oneM2M** is to enable efficient data processing, low-latency communication, and enhanced service delivery in edge computing environments. The major capability of this integration includes **federation of the ETSI MEC platforms** and **interaction with oneM2M instances**, allowing for distributed computing and data management across multiple edge nodes. Both ETSI MEC and oneM2M provide some basic APIs and protocols in order to facilitate such a federation. However, orchestration mechanisms are still required to ensure seamless operation across multiple MEC/oneM2M instances. This clause describes a preliminary set of functional and technical recommendations for enabling a framework for the federation and orchestration of MEC/oneM2M instances. These recommended processes, which not only complement many use cases in clause 5 but also unlock additional value by introducing capabilities that extend and enhance their intended scope, including:

- **Handover:** refers to the process of transferring control of a device or service from MEC/oneM2M instance to another, ensuring seamless connectivity and service continuity.
- **Swarm Computing:** refers to the coordination of multiple MEC/oneM2M instances to perform distributed computing tasks, leveraging the capabilities of edge devices and networks.
- **Federated Learning:** refers to a machine learning approach where multiple MEC/oneM2M instances collaboratively train and share models across multiple decentralized nodes without sharing row data, enhancing privacy and reducing latency.

The federation and orchestration framework will explore the existing APIs for discovery and resource management principles defined in both standards, while proposing extensions to them with federation and synchronization capabilities. Before orchestrating handover, swarm computing, or federated learning mechanisms, MEC/oneM2M instances are expected to be able to discover, register, and organize themselves into collaborative groups:

- **Instances Registration:** MEC/oneM2M nodes (e.g. IN-CSE, MN-CSE, MEC Platform, MEC Host, etc.) should be able to register to a federation registry, that may include its identity, capabilities, and available services.
- **Instances Discovery:** MEC/oneM2M nodes should be able to advertise its capabilities through standard discovery mechanisms (e.g. oneM2M Discovery resource, MEC location, etc.). This might be achieved by publishing metadata about the MEC/oneM2M node's processing power, storage capabilities, network connectivity, and supported APIs.
- **Instances Federation:** once discovered, MEC/oneM2M instances can be logically grouped into a federation group. Such a group might be a collection of MEC/oneM2M instances that are allowed to intercommunicate and cooperate in executing distributed tasks. Federation groups could be defined based on geographic proximity, application domain or QoS requirements.
- **Roles Assignment:** based on their capabilities, the MEC/oneM2M nodes could be selected for specific roles such as handover anchor, swarm computing worker or federated learning participant. For the instance, MEC/oneM2M nodes with high processing power and storage capacity could be selected for swarm computing and federated learning training tasks, while nodes with good connectivity could be used for handover, service continuity and low-latency communications.

Therefore, the orchestration framework should:

- Implement resource replication mechanisms to mirror or move device/application registration between MN-CSE instances or IN-CSE or across MEC hosts including the reference criteria for doing that.
- Ensure a handover-safe mechanism for migrating subscriptions (first establish a new subscription and then terminate the old one).
- Establish synchronization mechanisms between MN-CSE and IN-CSE to ensure that the recent device states are propagated upward and historical data are migrated asynchronous, when necessary.

- Identify and discover MEC/oneM2M instances from logical federation group.
- Match available resources (e.g. storage, compute, connectivity, etc.) with service requirements (e.g. handover, swarm computing or federated learning).
- Assign tasks to federation nodes.
- Keep registrations, subscriptions, and data synchronized across MN-CSEs/IN-CSEs and MEC nodes.
- Continuously monitor nodes performances and workload by reassigning tasks or triggering handover when needed.

The orchestration framework should create a federated, service-aware, and resource-optimized environment where ETSI MEC and oneM2M instances jointly deliver advanced distributed computing and IoT data management, by addressing handover continuity, swarm scalability, and federated learning privacy. It may also support dynamic service discovery and registration, load balancing across multiple instances, and failover and redundancy mechanisms.

7.2 Handover

The Handover mechanism refers to the process of transferring control of a device or service from one MEC/oneM2M instance to another, ensuring seamless connectivity and service continuity. There three main scenarios where handover is essential:

- Optimizing the connection of wireless IoT devices as they move geographically, ensuring they remain connected to the most appropriate edge node. This process, illustrated in Figure 7.2-1, is critical when devices move between different edge nodes or when services are migrated to maintain low latency, reliability, and high availability.
- Maintenance of services or servers that need to be migrated between MEC/oneM2M instances due to load balancing, resource optimization, or fault tolerance.
- Emergency scenarios where rapid handover is needed to maintain service continuity during network disruptions or failures, such as a failure of an edge node or a sudden surge in demand.

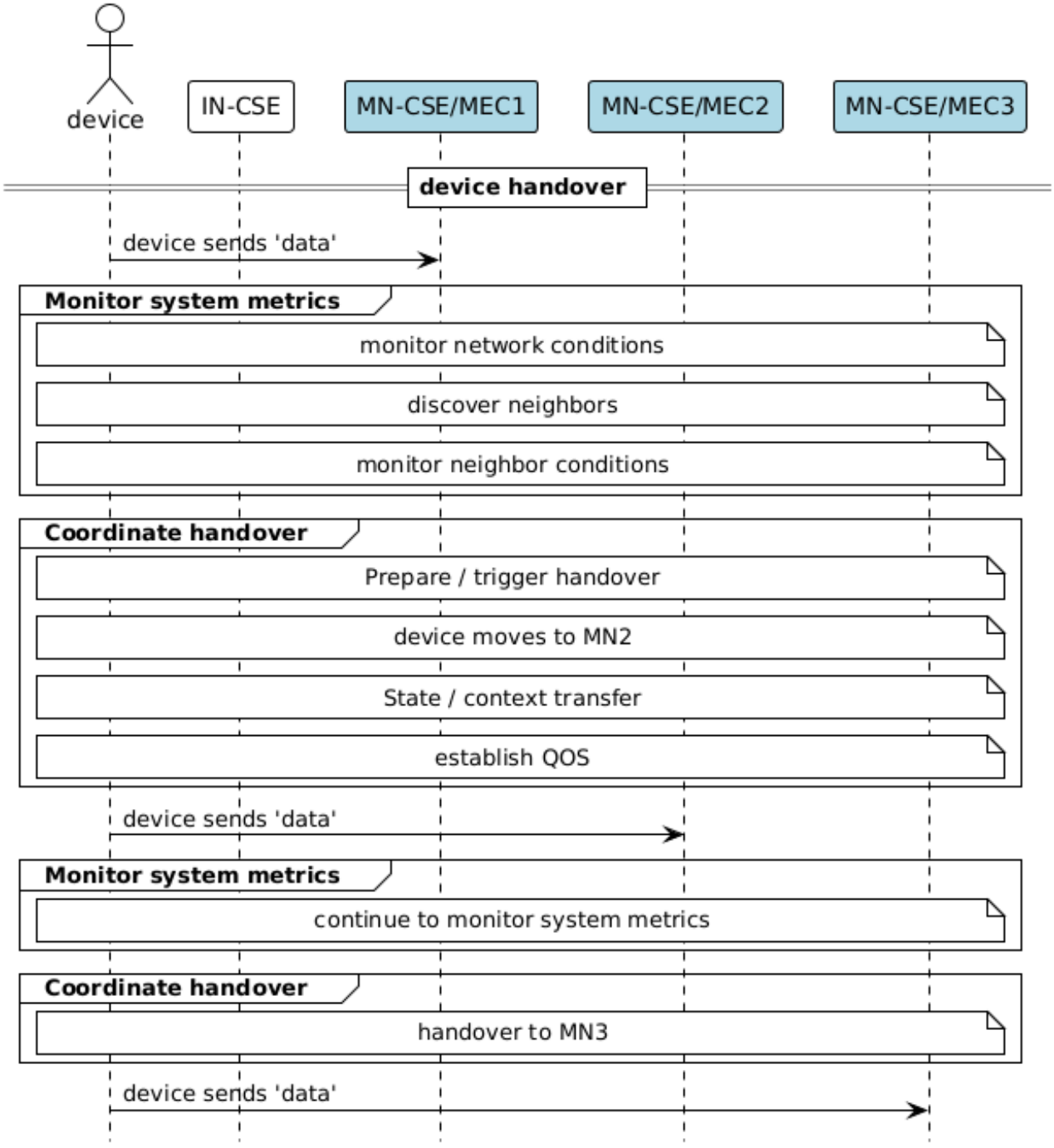

Figure 7.2-1: IoT device handover

Depending on system design and device capability, handover control can follow one of two approaches:

- Device-controlled handover: The device itself detects conditions that trigger a handover (e.g. deteriorating radio quality, mobility, policy changes). It initiates discovery, re-registration, and session updates with the target MEC/oneM2M instance. This approach offers autonomy but assumes the device has sufficient resources to evaluate conditions, make decisions, and handle signalling.

- Network-controlled handover: The MEC/oneM2M infrastructure (e.g. MN-CSEs or gateways) monitors device connectivity and system load, then initiates or enforces a handover. The device simply follows the instructions (e.g. re-register to a new MN-CSE) without needing complex logic or extensive signalling capability. This approach could be suitable for constrained IoT devices with limited processing or decision-making capacity.

In practice, the chosen strategy should balance device capability, network intelligence, and the requirements of the application domain. For resource-constrained IoT devices, offloading decision-making to the MEC/oneM2M layer may reduce device complexity and power usage, while device-controlled handovers may provide faster reaction times and greater autonomy in heterogeneous deployments. To design an effective handover mechanism, the following high-level considerations should be addressed:

- **Identity & Security:** The device is expected to maintain a consistent identity across MEC/oneM2M instances, with fast and secure re-authentication (e.g. token-based methods). Security credentials and keys should be managed automatically to ensure a smooth transition.
- **Session & State Continuity:** Handover is intended to preserve session continuity and data consistency. This includes mechanisms for state replication (buffers, acknowledgments, sequence numbers), subscription reinstatement, downlink catch-up, and, where possible, context transfer of QoS or profiles between gateways.
- **Performance & Latency:** Interruptions are expected to be minimized with well-defined handover budgets (e.g. ≤ 200 ms for critical telemetry). Local buffering, ordering policies, duplicate suppression, and time synchronization help maintain low latency and reliable delivery.
- **Network Awareness & Adaptation:** The mechanism should leverage network conditions such as bandwidth, signal strength, and congestion when selecting target gateways. Neighbour awareness, proactive discovery, and policy-based triggers (RSSI/SNR thresholds, cost/latency trade-offs) are key to ensuring optimal handover decisions. Admission control policies are typically considered, allowing gateways to accept or defer connections gracefully.
- **Monitoring & Reliability:** System reliability depends on the ability to measure and manage handovers. KPIs such as attach/authentication time, packet loss, and latency deltas should be tracked. Tracing, alarms for frequent or failed handovers, and robust retry/backoff logic ensure stability and resilience.
- **Device Constraints & Efficiency:** IoT devices, often resource- or power-constrained, require lightweight solutions. Gateway-local registrations can reduce complexity, while power-aware scanning and cached neighbour reports minimize RF cost. Feature negotiation and versioned message contracts further improve compatibility and efficiency across heterogeneous deployments.

These considerations will be further analysed in oneM2M and ETSI MEC.

7.3 Swarm Computing

Swarm Computing refers to the coordination of multiple MEC/oneM2M instances to perform distributed computing tasks, leveraging the capabilities of edge devices and networks. In this paradigm, individual MEC/oneM2M nodes act like members of a swarm, each contributing processing power, storage, connectivity, or sensing capabilities to achieve collective goal. The system operates in a decentralized and adaptive manner, where tasks can be dynamically partitioned, distributed, and recombined across nodes depending on resource availability, network conditions, and application requirements. This enables resilient, scalable, and low-latency processing, as tasks are executed closer to the data sources and devices while ensuring cooperative load balancing, fault tolerance, and energy efficiency. When mapped onto MEC/oneM2M interworking, distributed instances could be implemented as Application Entities (AEs) hosted on Middle Nodes (MNs) or Application Dedicated Nodes (ADNs). In this scenario, local coordination within a swarm could be supported by nearby Common Service Entities (CSEs). In ETSI MEC terms, there can be MEC Services or MEC Applications which provide contextual information aggregated from other swarm nodes. In this clause, the entities involved are described as follows:

- **Swarm Agent (Local/Edge):** is a software entity running on a device, MEC host, or oneM2M Application Entity that performs local computation and participates in cooperative swarm behaviour. It executes subtasks of a global distributed task, shares local state, sensor data, or intermediate results with peers. The local Swarm Agent may run on constrained devices or ADNs, focusing on lightweight processing and sensing, while the Edge Swarm Agent may run on MEC hosts or MNs, handling more complex subtasks and acting as a bridge to cloud/federated nodes.
- **Swarm Entity**: is any participating user or platform (e.g. device, MEC host, AE, MN, or ADN) that contributes computing, storage, or sensing resources to the swarm. It executes assigned subtasks and exchanges state updates with other nodes via oneM2M group communication or MEC APIs.

- **Swarm Collector:** software (e.g. CSE or MEC host) that aggregates results from multiple swarm nodes and produces a unified output. It collects partial results from distributed swarm nodes, performs aggregation (e.g. fusion of sensor data), and provides the final output back to the orchestrator, applications, or end-users.

- **Swarm Orchestrator:** is the central coordination entity of the swarm that manages task distribution, resource allocation, synchronization, resilience, and policy enforcement. It splits global tasks into subtasks, assigns subtasks to swarm nodes based on capacity, connectivity, and energy, detects node failures and reassigns tasks, and ensures QoS, latency, energy, and fault-tolerance goals. It may operate centrally (e.g. cloud IN-CSE) or decentrally (e.g. at MEC/MN-CSE) depending on deployment.

Consistent with the deployment options described in clause 5, swarm computing may be implemented through the following options:

- **Option 1:** In this configuration, Swarm Entities ask Swarm Agents to perform local tasks and interact with a Swarm Collector located nearby at the edge. The Swarm Orchestrator coordinates task assignment and resource allocation. Swarm Agents can execute subtasks locally and return intermediate results to the Swarm Collector, which consolidates outputs and sends them back to the Swarm Orchestrator. The Swarm Orchestrator and the Swarm Collector may be co-located or deployed separately on the edge. This option emphasizes low-latency collaboration between distributed Swarm Nodes and is well-suited for scenarios where tasks require real-time responses and localized aggregation as illustrated in Figure 7.3-1.

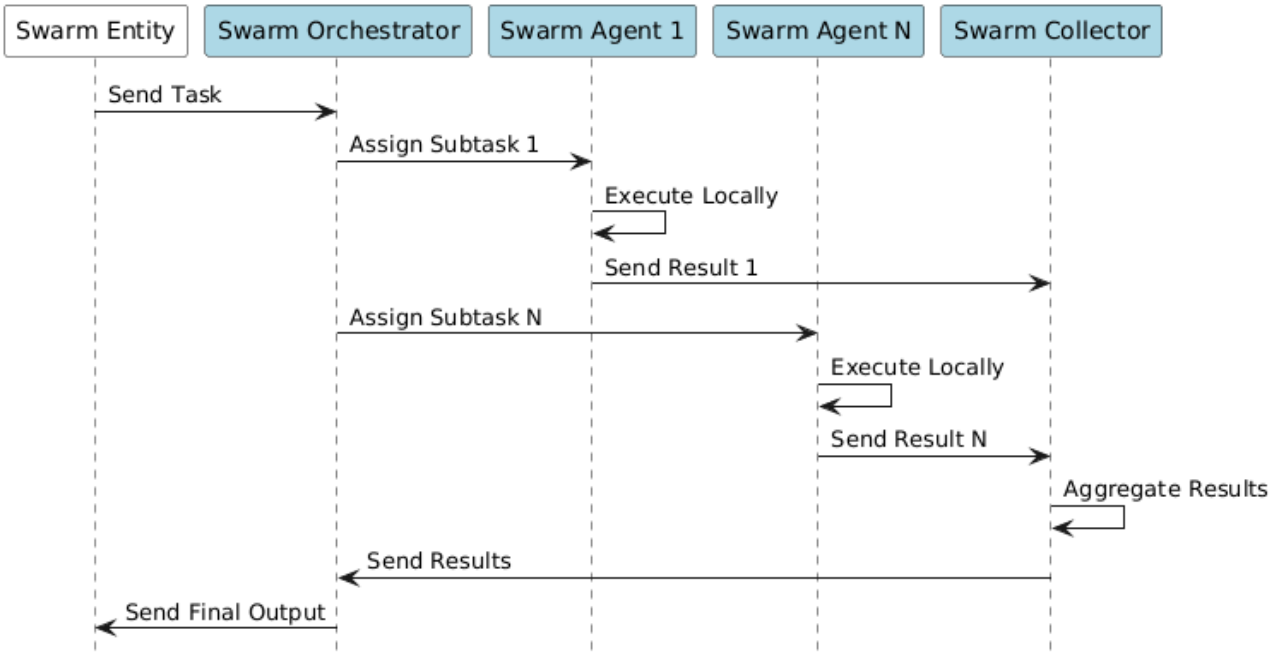

Figure 7.3-1: Swarm Computing Implementation - Option 1

- **Option 2:** Swarm Entities could be resource-constrained devices running lightweight processing. These devices can only execute simple tasks and rely on the Swarm Agents for heavier processing and the Swarm Collector for results aggregation. The Swarm Orchestrator manages tasks' distribution and offloading policies, ensuring that constrained Swarm Entities are not overloaded. The Swarm Collector aggregates results from multiple Swarm Agents (e.g. deployed on the edge, namely Swarm Agent 1 and Swarm Agent 2) in charge of processing of the different sub-tasks, and returns consolidated outputs. This result is then shared with the Swarm Orchestrator as well as with the Swarm Agent that sent the original request. Finally, the initial Swarm Agent (Local) performs post-processing operations according to the requirements of the initial request, as shown in the Figure 7.3-2. In this option, the Swarm Collector and the Swarm Orchestrator may be co-located on the edge platform.

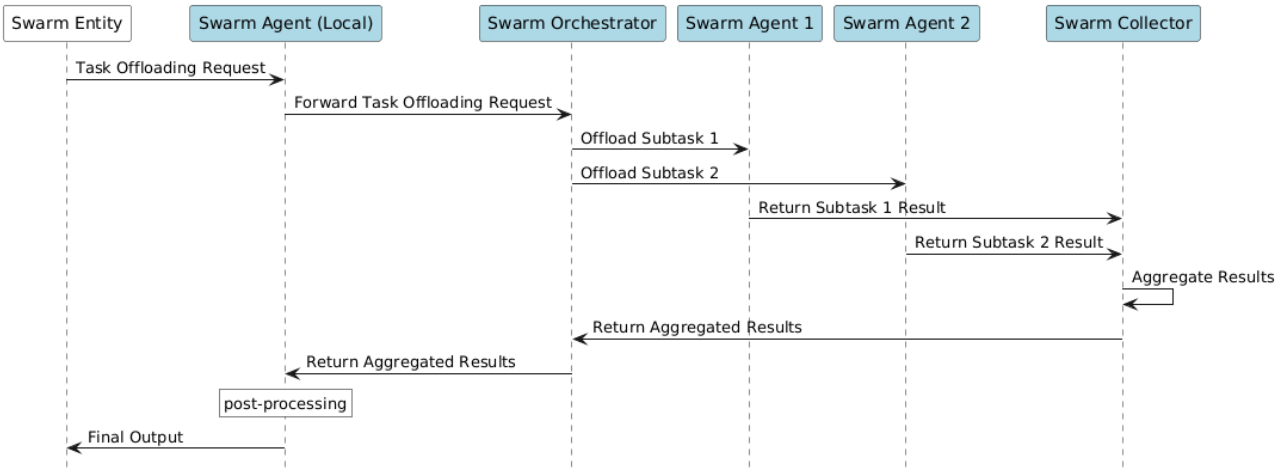

Figure 7.3-2: Swarm Computing Implementation - Option 2

- **Option 3:** This option extends orchestration beyond the edge by combining local processing with cloud-level intelligence. Swarm Entities interact with Swarm Agents at the edge (close to the devices) to perform latency-sensitive processing. In addition, the Swarm Collector, located in a more centralized or cloud domain, executes high-level analytics such as large-scale pattern recognition or long-term optimization using datasets that have been previously collected. The Swarm Orchestrator ensures synchronization between local results coming from the Swarm Agent at the edge and global insights from the cloud, as illustrated in Figure 7.3-3. The Swarm Collector (Global) receives the data produced by the Swarm Entity via the Swarm Orchestrator, and then performs application-specific pattern recognition which is shared with the Swarm Entity via the Swarm Orchestrator and the Swarm Agent. This hybrid model leverages the strengths of both edge and cloud: low latency at the edge and high computational power in the cloud, enabling scalable and adaptive swarm intelligence. This option is more application-specific because it divides intelligence between cloud and edge based on the workload. While Options 1 and 2 mainly vary by how close computation is to devices or how much is offloaded, Option 3 requires a split between real-time behaviour at the edge and high-level reasoning in a cloud. Therefore, the placement of functions depends on the type of analytics, latency constraints, and data-handling rules of each application.

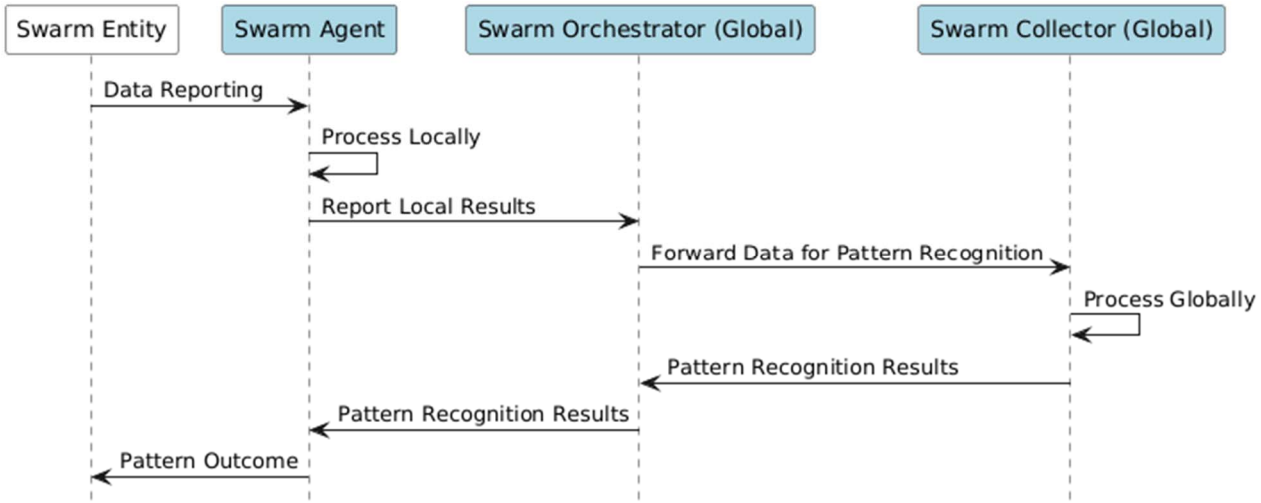

Figure 7.3-3: Swarm Computing Implementation - Option 3

To allow multiple MEC/oneM2M nodes to collaboratively perform distributed tasks, the orchestration mechanisms should take into account the following aspects:

- **Task Decomposition:** a global computation task is divided into smaller subtasks. The orchestrator distributes these subtasks to participating MEC/oneM2M nodes based on their declared processing and storage capacity, network connectivity, or CPU/GPU availability.

- **Synchronization:** appropriate synchronization mechanisms (e.g. publish/subscribe over oneM2M and MEC APIs) should ensure swarm nodes operate on consistent views of data. To maintain consistent swarm knowledge across federated nodes, a distributed state synchronization is needed.
- **Task offloading:** the orchestration mechanism should support task offloading policies where swarm members dynamically delegate high-complexity processing to nearby MEC/oneM2M nodes.
- **Resilience:** in case of node failures, the orchestrator dynamically reallocates unfinished subtasks to other swarm nodes.
- **Coordination and Aggregation:** results from swarm nodes are aggregated at a designed MEC host, IN-CSE or MN-CSE which acts as a collector node. The oneM2M's group communication primitives could be used to coordinate actions across swarm members, while MEC APIs could be used to optimize routing and QoS for tasks exchanges.

The swarm computing orchestrator is the coordinator, load balancer, and fault manager of swarm computing. It does not always execute tasks itself, instead it manages how, where, and when the tasks are executed, ensuring that the swarm MEC/oneM2M nodes behave like a single resilient and adaptive system.

7.4 oneM2M and MEC support for federated Learning

Federated Learning is a distributed machine learning approach in which model training is performed locally on multiple nodes and only model's parameters or weights, not row data, are exchanged with aggregators. Instead of sending sensitive data to a central location, each node trains a local model using its own data and then shares only the model updates with a central location, which aggregates these updates to improve the global model. This enhances privacy, reduces bandwidth usage, and allows model adaptation close to data sources and IoT devices.

In the MEC/oneM2M interworking context, local training can be executed within Application Entities (AEs) on Middle Nodes (MN-CSE) or even on constrained edge devices, while aggregation roles can be hosted on Infrastructure Nodes CSEs (IN-CSE) in the cloud environment, or on ETSI MEC applications acting as intermediate servers at the edge. ETSI MEC services could provide capabilities such as Location API, Radio Network Information API, or IoT API to enrich local training with contextual features.

In this clause, the entities involved are described as follows:

- **FL Client:** a participating device or node (e.g. IoT sensor, oneM2M AE, oneM2M MN-CSE, MEC node or application entity) that trains a machine learning model locally using its own private data. It holds raw data (never shared outside the device), performs local model training based on the global model received from the server/aggregator, and sends only model updates/gradients (not raw data) back to the FL Server or Aggregator.
- **FL Server:** coordinates the training process across all clients by distributing the initial global model, collecting updates, and managing the learning rounds. It selects which clients participate in each training round, sends the current global model parameters to selected clients, receives updated parameters or gradients from clients after local training, and passes these updates to the FL Aggregator for combination. The FL Server could be hosted on a MN-CSE (at the edge) acting as a local server distributing models to nearby AEs, on an IN-CSE (in the cloud) serving as a central FL Server for large-scale coordination, on a MEC Application or MEC Hosts playing the role of a distributed FL Server close to the edge FL Clients, or on a MEC Platform hosting control logic for FL Clients selection and management/orchestration.
- **FL Aggregator:** is responsible for combining model updates from multiple clients into a new, improved global model. It performs model aggregation (e.g. weighted averaging of client models), handles data heterogeneity by balancing contributions from clients (e.g. clients with more data may weigh more), and ensures robustness against noisy updates, stragglers, or malicious clients (e.g. secure aggregation techniques). The FL Aggregator could be hosted as an IN-CSE in the form of a central orchestration entity coordinating across multiple CSEs, on a MN-CSE for local orchestration, or on a MEC Platform exposing APIs for monitoring resources and optimizing FL Client/Fl Aggregator placement.

- **FL Orchestrator:** manages the overall lifecycle and policies of the federated learning process, ensuring scalability and resilience across distributed clients and servers. It decides which clients participate in each round (e.g. based on resource availability, network conditions, or data diversity), optimizes how training workloads are distributed, especially in heterogeneous edge/cloud environments, handles privacy rules, security requirements, energy constraints, fault tolerance, and orchestrates interactions across multiple FL Servers/Aggregators. The FL Orchestrator maps to high-level coordination functions in MEC Platform or in oneM2M IN-CSE but can also exist locally at MN-CSE for edge deployment scenarios.

Deployment of federated learning may be implemented through the following options:

- **Option 1:** In this option, the FL Orchestrator might be hosted in a cloud-based environment, while the FL Aggregator and FL Server might run on the edge. Finally, FL Clients could be hosted on the edge or device, depending on the computational capabilities. The FL Orchestrator, as depicted in Figure 7.4-1, initiates the first training round and delivers the initial global model to the FL Server which is in charge of distributing the model to each FL Client. Each FL Client trains the model locally on private data and sends its model updates back to the FL Server. The FL Server forwards the collected client updates to the FL Aggregator (Global) which combines the updates to produce the global model. The global model is returned to the FL Server, which passes it back to the FL Client and the FL Orchestrator. The FL Orchestrator starts then the next training round. In this configuration, the FL Server coordinates participation in each round and forwards collected updates to a FL Aggregator (Global) located at the edge, while the FL Orchestrator supervises the process by initiating training rounds, distributing models, and ensuring synchronization across all participants. The Option 1 is suitable for applications that require rapid response as the aggregation is completed at the edge only (locally), and it addresses low latency requirements since the process remains close to the data sources.

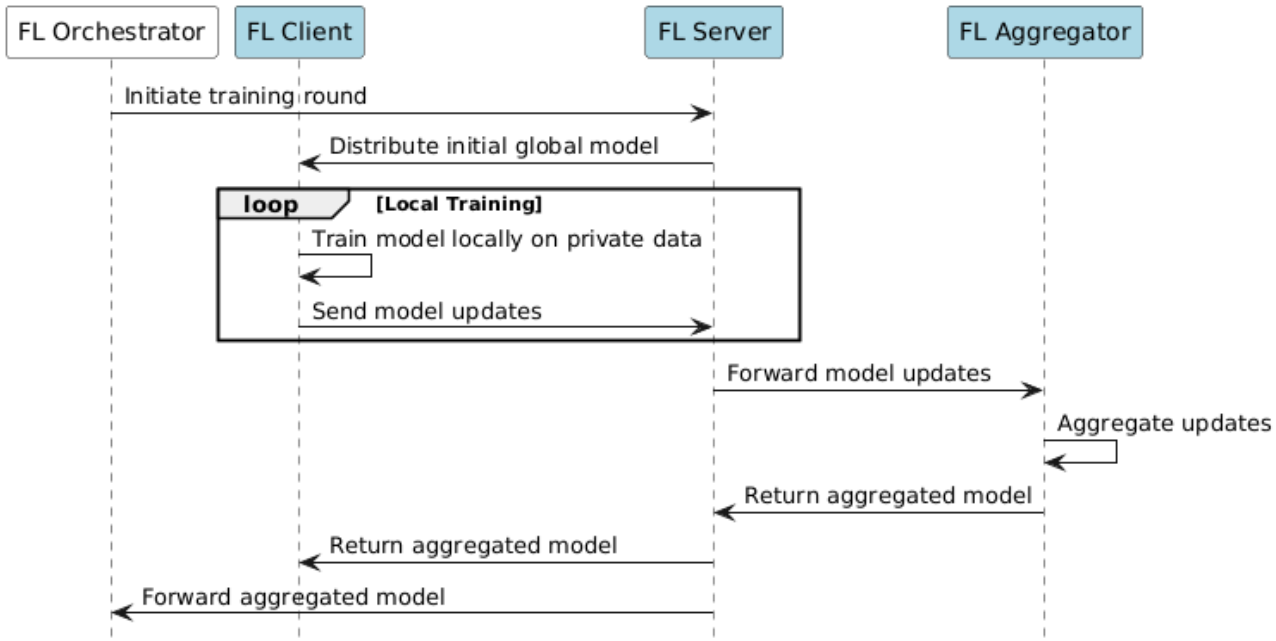

Figure 7.4-1: Federated Learning Orchestration - Option 1

- **Option 2:** In this option, both the FL Orchestrator and the FL Aggregator (Global) might be hosted in a cloud-based environment, while the FL Aggregators (Local) and FL Servers might run on the edge. Finally, FL Clients could be hosted on the edge or device, depending on the computational capabilities. The FL Orchestrator, as depicted in Figure 7.4-2, initiates the first training round and distributes the initial global model to the FL Server which is in charge of distributing the model to each FL Client. Each FL Client trains the model locally on its private dataset and sends model updates to the FL Aggregator (Local) which performs partial aggregation of the client updates to reduce communication overhead. The aggregated results are forwarded to the FL Server, which transmits the aggregated updates to the FL Aggregator (Global). The FL Aggregator (Global) performs the final global aggregation, producing the improved global model. This model is then returned to the FL Server and then to the FL Orchestrator. The FL Orchestrator initiates the next training round. In this option, the FL Servers act as the local coordination layer, while the FL Orchestrator manages the end-to-end process. The Option 2 is suitable for large-scale and geographically distributed deployment scenarios as the aggregation occurs in two stages (first at the edge, then globally).

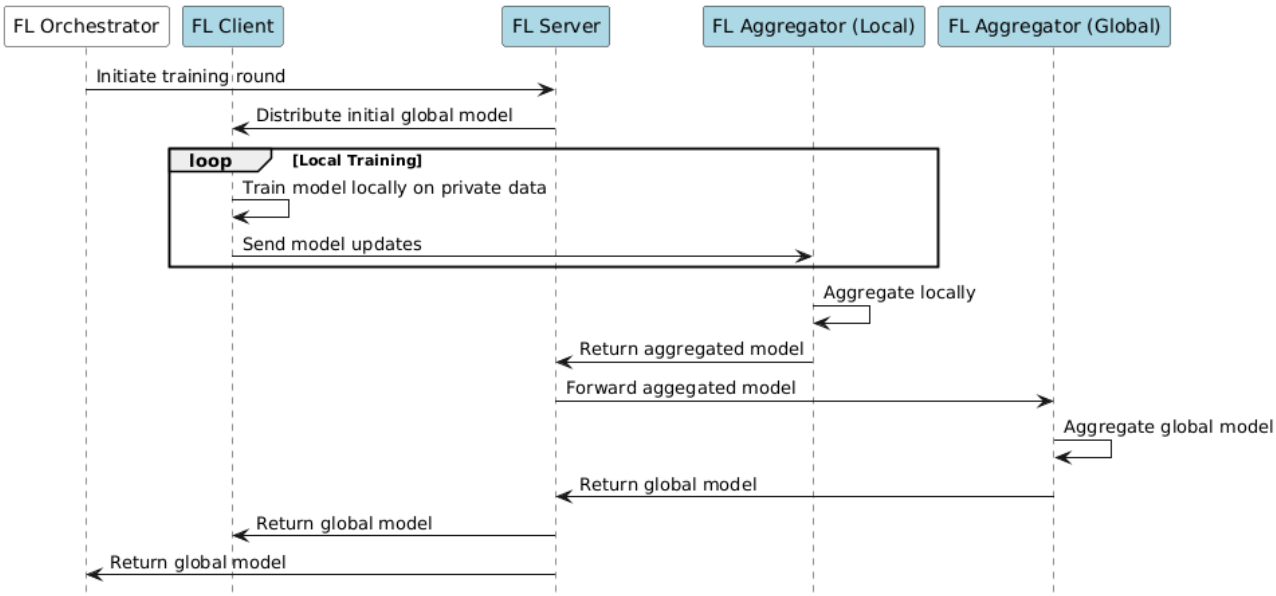

Figure 7.4-2: Federated Learning Orchestration - Option 2

The orchestration mechanism should be able to perform the following steps:

- **Federation Group Selection:** based on the federation registry, MEC/oneM2M nodes or instances with sufficient compute and local training capabilities are selected as participants to the federated learning process.
- **Model Distribution:** a global training model initialized at IN-CSE or MEC Orchestrator is distributed among selected participants.
- **Local Training:** each selected node trains the model locally using its own data. The training parameters (e.g. weights, gradients, etc.) are stored locally and not exposed outside the node.
- **Aggregation and Synchronization:** model's updates are periodically transmitted to the aggregation nodes which might be either an IN-CSE or a MEC host. The aggregator combines updates using secure aggregation protocols and redistributes the improved global model. This allows to maintain consistency between IN-CSE/MEC host, MEC/oneM2M nodes and MEC orchestrator (aggregator).

History

Version	Date	Status
V4.1.1	November 2025	Publication