Bi-Directional APIs: Advancing MEC

Issue 1.0

30th January 2024

1. Table of Contents

2. Abstract

Today, many Telecommunication Service Providers (TSPs) worldwide are actively developing network Application Programmable Interfaces (APIs) to expose their capabilities to developers for value-added applications. Specifically, the GSMA Open Gateway provides a framework of common network APIs, under which the Linux Foundation CAMARA project is creating developer-facing APIs and the TM Forum's Open API project is defining APIs for network operation and management. These developer-facing APIs are network-centric, and often called "Network as a Service" (NaaS) APIs. By design, they do not involve the aspect of cloud services, which are widely used by cloud or Multi-access Edge Compute (MEC) centric applications.

This whitepaper aims to increase industry awareness of the value of Bi-Directional APIs beyond the NaaS APIs exposed to application developers. The purpose of Bi-Directional APIs is to enable information sharing between TSPs and Cloud Service Providers (CSPs), empowering each to deliver innovative MEC services to their customers at the right location and with the right compute power and network performance and to take full advantage of the capabilities that MEC has to offer. The whitepaper intends to stimulate discussion of Bi-Directional APIs with MEC service deployment models, Developer-TSP-CSP interaction models and several use cases. It seeks feedback from the GSMA Open Gateway and developer communities and Cloud Service Providers. Only when the Bi-Directional API use cases are recognised and requirements are solidified can progress on Bi-Directional API specifications begin.

1. Introduction

1.1. The Bi-Directional API Concept

The introduction of 5G and Multi-access Edge Compute (MEC) has transformed the way that services are delivered to customers by bringing powerful compute capability to the edge of the network. Bringing computing power closer to end-users allows for the data generated to be processed locally, unlocking a wealth of potential for new and enhanced enterprise and consumer applications that require faster response times, greater resiliency, flexible data residency and a better overall customer experience for the next wave of customer use cases. However, distributing compute services close to the end-user brings unique technical challenges in seamlessly weaving the network and compute together to bring MEC use cases to life and allowing Telecommunication Service Providers (TSPs) and Cloud Service Providers (CSPs) to drive value from their platforms and services. For example:

- Physical site resource constraints (e.g., rack space, power and air-conditioning) can place an upper limit on MEC platform capacity and thus the level of compute elasticity available to applications at a particular MEC location;
- Optimal workload placement will need to consider both network and MEC infrastructure characteristics (e.g., real-time compute resource availability, network topology, available network capacity and quality-of-service capability);
- Maintaining a seamless user experience as cellular customers move throughout the network may require a change in workload placement to maintain the end-to-end experience across network, compute, and application.

Bi-Directional APIs for MEC play a key role in addressing these challenges by enabling information sharing between TSPs and CSPs to support discovering, assigning, and managing MEC and network resources and services effectively. Such information and capabilities may include:

- **From CSPs:** Real-time and historical information around compute resource and platform services availability, capacity trends and MEC service locations.
- **From TSPs:** Real-time and historical information related to the communication medium (such as network or connection metadata, or anonymised details of the end customers like device location or demographics) and the ability to request network resources critical for fulfilling the end customer's quality of experience.

1.2. Industry API landscape

It is paramount that Bi-directional APIs are implemented in alignment with Industry Bodies and Standard Development Organisations (SDOs) to avoid duplication and take advantage of existing frameworks. The GSMA is one such body at the forefront of Industry innovation, launching the Open Gateway program aimed at providing universal access to TSP networks for developers. The Linux Foundation's CAMARA project is creating the developer-facing APIs used by Open Gateway, and the TM Forum's Open API project is focused on how API services are to be operated and managed.

The GSMA's whitepaper, "The Ecosystem for Open Gateway NaaS API Development", describes the current ecosystem and the types of APIs needed to align the Industry. The document defines the following three categories of APIs:

- **Service APIs** - Developer-oriented APIs that provide purpose-specific capability (e.g., edge discovery, quality on demand, device location) and are defined in CAMARA sub-projects;
- **Operate APIs** - Management-oriented APIs that provide programmable access to Operation, Administration and Management capabilities, and are typically defined through the TM Forum; and
- **Technology-specific APIs** - Operator-internal APIs offering programmable access to telco infrastructure and network, service, and IT capabilities, typically defined by specific SDOs like 3GFF, ORAN, ETSI etc.

The Open Gateway ecosystem and API categories described in the GSMA whitepaper are based on the compute infrastructure being within the Telecommunication Service Provider's domain. Bi-Directional APIs address the scenario where the compute infrastructure is within a Cloud Service Provider's domain. They are an additional category of API designed to enable information sharing between TSPs and CSPs to support discovering, assigning, and managing MEC and network resources and services effectively. These APIs operate in the background, behind the GSMA Open Gateway and CSP Service APIs, and are not directly visible to developers.

The figure on the next page describes an API call model capturing the relationship between the API Consumer (Developer's application), TSP APIs, and CSP APIs:

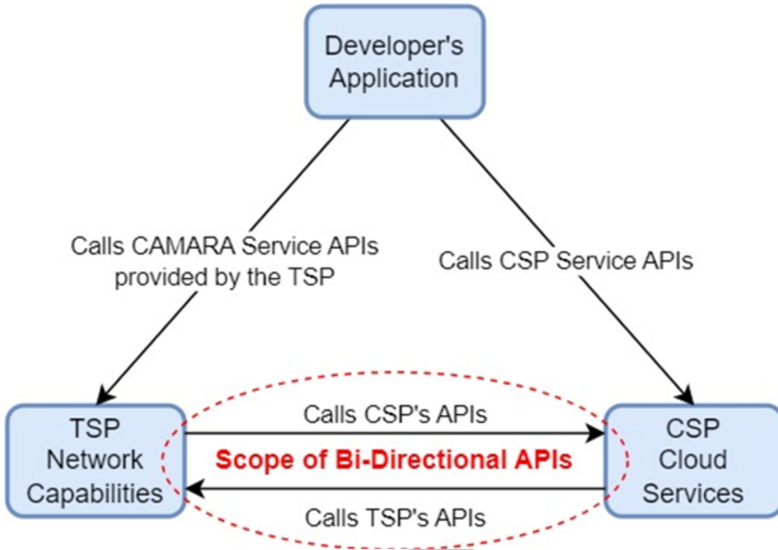


Figure 1: API Relationship between Developer, TSP, and CSP

In this model, Developers access network capabilities exposed by TSPs through CAMARA Service APIs as described in the GSMA whitepaper. Further, the developer accesses CSP services like cloud compute, storage and CI/CD application deployment pipelines for part of their application features.

Bi-Directional APIs can then help facilitate the sharing of necessary information between TSPs and CSPs, thus allowing each of them to deliver the MEC experience that the developer requires. For example:

- Telecommunications Service Providers access CSP APIs that provide information about the availability of edge services or service locations, so that they may allocate and manage developers' MEC applications.
- Cloud Service Providers access TSP APIs such as location or network performance monitoring, which the CSP services are relying on.

The 5GFF looks forward to working closely with our Industry partners to drive development of Bi-directional APIs, enabling both TSPs and CSPs to deliver innovative services to developers that take full advantage of the capabilities that MEC has to offer.

1.3. About the 5G Future Forum

Founded in 2020, the 5GFF aims to accelerate the delivery of 5G and MEC enabled solutions for developers and multinational customers around the world. Adopting a partnership-focused approach enables operators to create interoperable specifications, driving reach and scale, while empowering members to jointly innovate 5G cutting edge applications and use cases.

The 5GFF Experience and Exposure Management workstream is developing and testing APIs that allow developer applications to seamlessly transition between MEC platforms across operators and regions. The technical teams work with leading technology companies and platform providers across the world, ensuring that the 5GFF APIs are applicable across a host of use cases and can be rapidly scaled, whenever and wherever needed.

In addition, the 5GFF technical teams are also collaborating with the API development teams at GSMA-led initiatives such as Open Gateway and Project CAMARA, to contribute to the global effort of promoting global MEC interoperability. By complementing inter-operator connectivity development work across the ecosystem, the 5GFF is also working towards removing a siloed approach to solving a common problem.

The Commercial Ecosystems workstream engages with the global 5G community, validating and testing emerging 5G use cases where the 5GFF APIs can provide the most value. The workstream focuses on potential 5G use cases that are feasible today and in the near-term, while also evaluating the application of 5G beyond the current technological horizon. This allows the workstream members to determine the network and technology advancements required to make future use cases a reality. The Ecosystems workstream also engages with other leading technology organisations, ensuring the relevance and increased adoption of the 5GFF APIs.

All workstreams are comprised of leading technical and commercial teams from the 5GFF member base, including Verizon, Rogers, Telstra, Vodafone, America Movil, KT, and Bell Canada. This enables the 5GFF to adopt a global and holistic view of where 5G is today and where it can be tomorrow.

Learn more about the 5GFF today and how we aim to revolutionise the deployment and adoption of 5G MEC in your home market: 5GFF website.

2. Purpose and Scope

The purpose of this whitepaper is to increase industry awareness of the ways in which bi-directional information exchange between Telecommunications Service Providers (TSPs) and Cloud Service Providers (CSPs) can enable both TSPs and CSPs to deliver innovative MEC services to their customers, at the right location and with the right compute power and network performance to take full advantage of the capabilities that MEC has to offer.

This whitepaper describes potential MEC service deployment models, the role of Bi-Directional APIs, potential bi-directional information flows and example use cases for Bi-Directional APIs between CSPs and TSPs.

The definition of specific Bi-Directional APIs is outside the scope of this whitepaper. Once industry consensus is reached on the value of Bi-Directional APIs, the 5GFF proposes that a new working group be established within CAMARA to drive Bi-Directional API development.

Note that some of the information that could be shared between CSPs and TSPs will require a level of abstraction, for example, information that is complex or privacy-sensitive in its raw form. The abstraction or obscuring of information is outside the scope of this whitepaper.

3. MEC Deployment models

Generally, two popular approaches of deploying MEC have emerged globally as described below.

3.1. TSP-hosted MEC services with embedded CSP platforms

Several TSPs globally are partnering with CSPs to deploy CSP MEC infrastructure within TSP premises, with responsibilities shared between both parties. Primarily, power, space, cooling, and network infrastructure are the responsibility of TSPs and MEC infrastructure deployment and maintenance are the responsibility of CSPs.

In this model, CSP edge infrastructure is embedded in TSP networks. TSP subscribers benefit with local access to MEC applications and CSP cloud services without requiring the traffic to be delivered all the way to the cloud region. The TSP and CSP work together to provide Quality of Service (QoS) end-to-end with respect to the profile of subscribers' application performance such as network latency, jitter, and throughput.

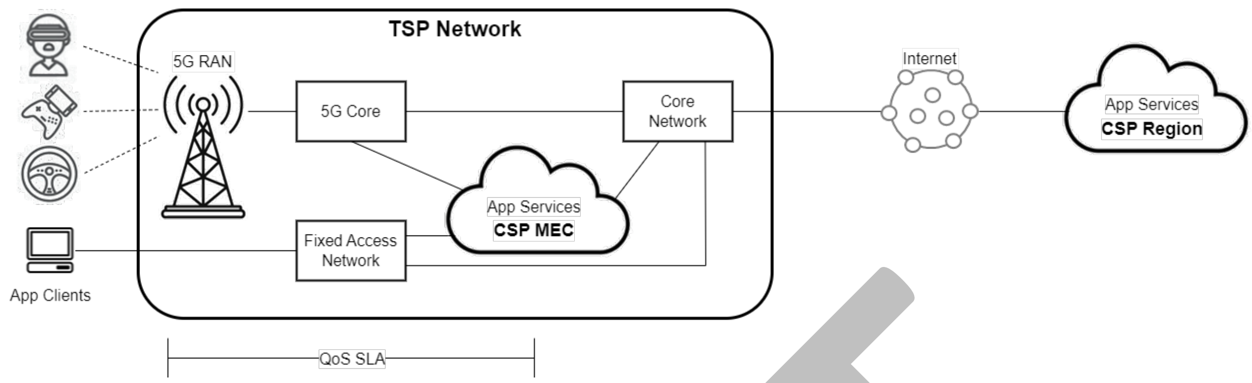


Figure 2: TSP-hosted MEC services with embedded CSP platforms

3.2. CSP-hosted MEC services with optimised TSP network connections

The TSP brings its network to the CSP MEC with direct connections. The TSP may optimise the connectivity between its subscribers' devices and the MEC application service endpoints in the cloud edge, for example, with the least number of hops or lowest latency path.

This model also leads to improved latency for the subscribers in comparison with cloud access through the Internet. The TSP and CSP serve the subscribers with value-added QoS across the TSP network, TSP and CSP interconnect, and CSP MEC.

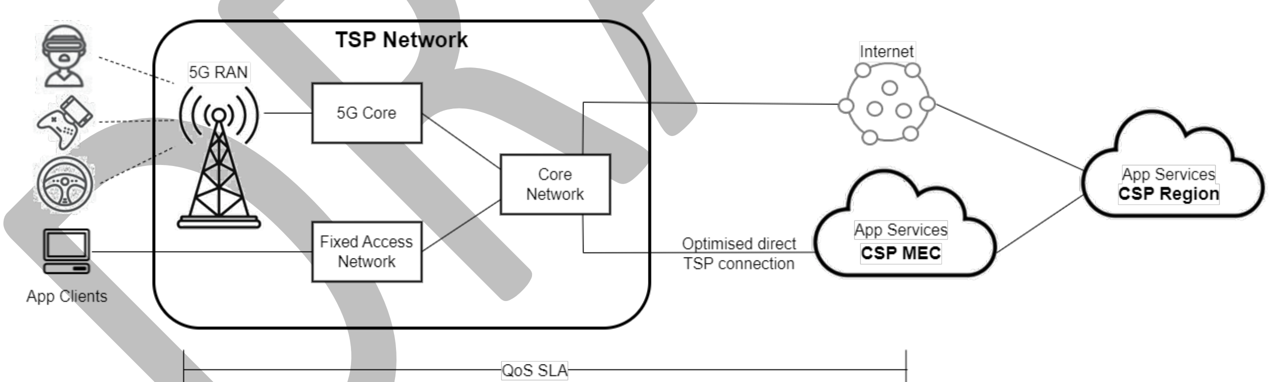


Figure 3: CSP-hosted MEC services with optimised TSP network connections

3.3. Relationships between TSPs and CSPs

In addition to the two deployment models described above, it is important to consider that the relationship could potentially be different among the different TSPs and CSPs partnering together, leading to more complexity due to the permutations and combinations possible with the MEC deployments. For example, potential relationships between TSPs and CSPs can be "one to many". In the case of TSP-hosted MEC, one TSP may host MEC infrastructure provided by multiple CSPs. With CSP-hosted MEC, one CSP MEC may have network connections with multiple TSPs. Such relationships may require new APIs to support TSP and CSP interactions for MEC application developers and users.

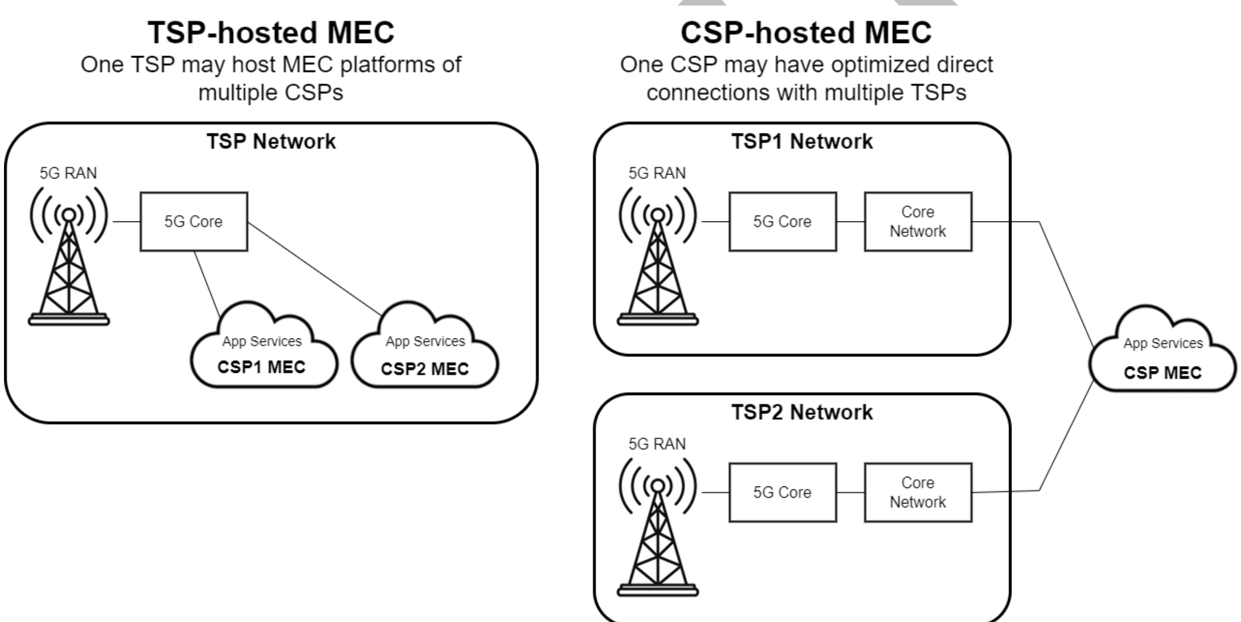


Figure 4: Relationships between TSPs and CSPs

4. The Value of Bi-Directional APIs

4.1. Developer Challenges

Today, CSPs have a mature set of APIs for interacting with their cloud services and the TSP community is working towards providing developers with universal access to network intelligence through standardized, developer-friendly network APIs. Specification of these developer-facing network APIs is being driven through the Linux Foundation CAMARA initiative, in which the 5G Future Forum (5GFF) is actively contributing.

While these standardized APIs will address a range of pain-points that developers face in deploying MEC applications, there remains the challenge that the TSP and CSP APIs operate independently of each other, i.e., there are no APIs that support capabilities like edge discovery, service discovery or performance guarantees spanning both TSP network and CSP MEC infrastructure. This is a potential barrier to adoption, as the developer must deal with the complexity of bringing the network and compute services together to create the desired end-user experience.

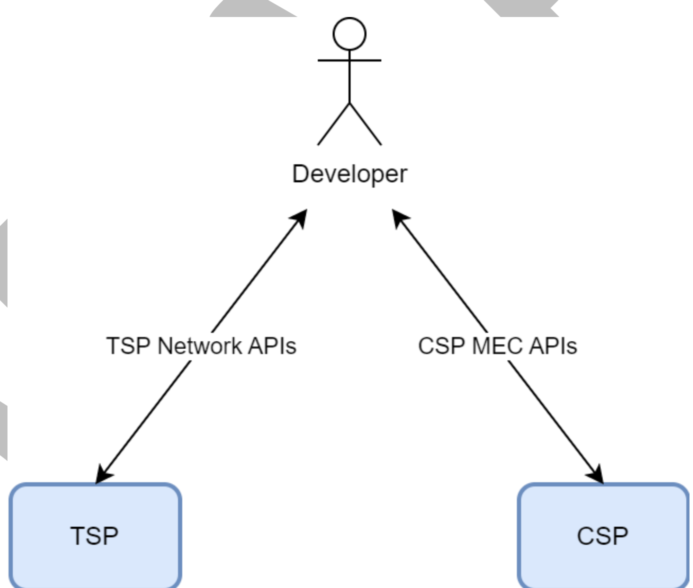


Figure 5: The Developer Challenge - Bringing network and compute services together

For example, to provide a cellular end-user with an application latency guarantee, the developer may need to query TSP APIs to identify the end-user location, followed by CSP APIs to identify the nearest MEC location with the required compute resource availability. Even then, there is no guarantee that the chosen MEC is optimal because this approach does not account for the underlying network topology or network conditions. Adding to this complexity, the developer may also be faced with doing this across multiple TSPs and/or CSPs.

Managing this complexity will become increasingly difficult as MEC becomes more distributed over time. An implication of greater distribution is that physical site resource constraints like rack space, power and air conditioning will place an upper limit on MEC capacity and thus the level of compute elasticity available to applications at MEC locations. As a result, an application may need to scale or switch-over across multiple MEC sites, all while ensuring that application performance (spanning network and MEC infrastructure) remains within SLA.

Mobility adds another layer of complexity for the developer. For example, to provide optimum application performance for mobile end-users, it may be necessary to re-home end-user connections as the end-users move to different locations. In more advanced use cases, workloads may also need to be dynamically migrated across MEC locations in response to end-user mobility. In each case, real-time coordination across both TSP network and CSP MEC infrastructure is required.

4.2. The Role of Bi-Directional APIs in Addressing These Challenges

The design complexity surrounding the build, deployment, and lifecycle management of MEC services and associated network connectivity services needs to be abstracted, so that developers have the freedom to create new products and services without needing to understand the intricate details of the underlying technology and network connectivity.

Bi-Directional APIs enable the exchange of information between TSPs and CSPs so that each can have the information required to offer developer-facing edge-native APIs that support serving applications at the right location, with the right computing power and the right network connectivity to maximise the value of the MEC offering. For CSPs, they also provide the benefit of a consistent level of abstraction and consistent APIs for network intelligence across all TSPs, accelerating global rollout.

Depending on various criteria, developers will have the flexibility to approach either TSP or CSP to consume edge-native developer APIs, accelerating innovation while the TSPs and CSPs take on the burden of stitching together the network and compute.

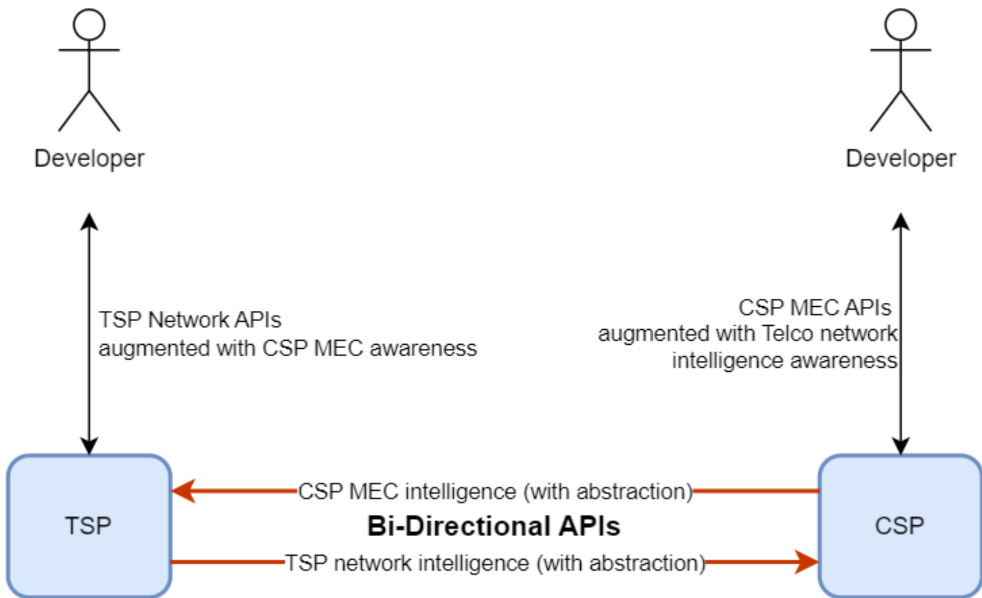


Figure 6: Information exchange through Bi-Directional APIs

Note that the developer-facing APIs offered by TSPs and CSPs should be focused on giving developers the necessary capability to fulfill their value proposition when building MEC applications. The information exchanged between TSPs and CSPs through Bi-Directional APIs should be abstracted and transparent to the developer experience.

5. Bi-Directional API Information Exchange

Bi-Directional APIs can facilitate information sharing between TSPs and CSPs, enabling them both to deliver enhanced developer APIs capable of supporting service discovery, workload management and other services spanning both TSP network and CSP MEC infrastructure.

5.1. Example Information Exchange from TSP to CSP

TSPs generate, and are party to, information which has the potential to provide useful data sets to improve the deployment and operation of applications and services deployed on CSP MEC infrastructure. This data may include the communication medium (such as network or connection metadata) or anonymised details of the end-user (such as device location or demographics).

This section describes some potential areas of information exchange from TSP to CSP to support CSPs in offering edge-native APIs augmented with Telco network intelligence. However, note that an appropriate level of abstraction will be required.

For example, TSPs may not wish to expose data that is complex or privacy-sensitive in its raw form. Possible abstractions of the information described here are outside the scope of this whitepaper.

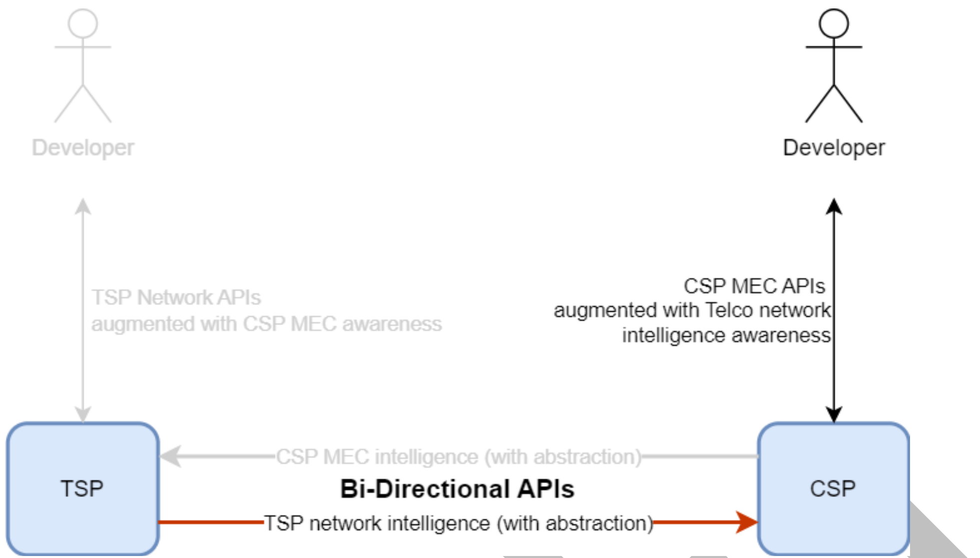


Figure 7: Information Exchange Through B-Directional APIs - TSP to CSP

5.1.1. Edge Discovery Augmented with TSP Network Information

Edge discovery is an intent-based service that a developer triggers to determine the optimal MEC location(s) to place their application. When edge discovery is performed, the response will be a MEC (or multiple MECs) that satisfies the developer intent, encompassing both network and compute requirements of the application.

As MEC becomes more distributed, network factors such as the underlying network topology, network capacity and quality-of-service capability become increasingly important in the placement of MEC applications.

TSPs can provide a range of network intelligence data to CSPs to assist in identifying the most appropriate MEC platform(s) for application deployment. For example, by combining knowledge of its MEC infrastructure with network intelligence from TSPs, CSPs will be able to offer developers an edge discovery service that ensures that the qualified MEC location(s) to place the workload has the appropriate network capacity and network performance to meet the application requirements.

5.1.2. Service Discovery Augmented with TSP Network Information

Service discovery addresses the problem of identifying the optimal application service endpoint to serve an end-user client application at runtime. As the volume of services increases and MEC deployments become more distributed, traditional approaches like DNS-based IP geolocation will become increasingly inaccurate.

A decision on the optimal service endpoint to serve a particular end-user client connection depends on several factors, including:

- Where the application is deployed: The application may be deployed on only a subset of available MEC platforms
- The state of the network: A range of factors, such as the underlying network topology, current level of network utilisation/congestion, cellular device location and anchor point, etc., could mean that the optimal service endpoint is not the one that is geographically closest to the end-user
- In some cases, where existing users of an application are connected: e.g., users of a remote-music collaboration application with demanding latency requirements may benefit from connecting to the same MEC platform via the optimal network path, as distributing such users over different MEC platforms may hinder performance

TSPs can provide a range of network intelligence data to CSPs to assist in identifying the most appropriate service endpoint to serve a client application at runtime. By combining knowledge of the services running on its MEC infrastructure with network intelligence from TSPs, CSPs will be able to offer developers a service discovery capability that ensures end-user clients are connected to the optimal service endpoint at all times.

5.1.3. TSP Network Analytics

TSPs generate domain-specific metadata during their operations, and sharing this data with CSPs may enable CSPs to enrich their services and improve user experience. Where it is not appropriate to share raw data (for example, data that is complex or privacy-sensitive), then TSPs may be able to share the results of data analysis which simplifies, abstracts and/or anonymises the data. Examples of network analytics that TSPs may share could include:

- Utilisation and availability of infrastructure and network resources across uplink/downlink, night/day or regional variance;
- Subscriber concentration at a MEC location (for example, to potentially assess and anticipate congestion);
- Historical network utilisation trends and significant variances / anomalies due to recurring or predictable events (based on data described in the above points);
- Near-real time signals that conditions may change, e.g., user nearing cell edge;
- Network performance analytics such as latency, jitter or cell-congestion conditions against a defined set of destinations;

- Cellular device-related analytics such as location (with an appropriate level of abstraction) and radio conditions; and
- Predicted network conditions for the duration of a session, or along a geographical route.

5.1.4. TSP Lifecycle Events

Certain TSP events may trigger a life-cycle management action within the CSP domain. Such events include network outages, planned maintenance windows, network congestion, a network utilisation "threshold met" or an end-user moving into a region best served by a different MEC. Receiving information about such events from the TSP will enable the CSP to effectively assist their customers to lifecycle manage their workloads, or to lifecycle manage workloads on behalf of their customers.

5.1.5. API Based Discovery for TSP Network Services

As MEC becomes more distributed, MECs may be deployed into areas with different levels of network capacity and capability. For example, bandwidth will be scarcer in regional areas than metropolitan areas, and the availability and capability of network services like network slicing and quality-on-demand may vary in different regions and across different TSPs. In order to perform an effective discovery and workload placement, it will be necessary to programmatically retrieve supported TSP network services at a given MEC location.

5.2. Example Information Exchange from CSP to TSP

This section describes some potential areas of information exchange from CSP to TSP to support TSPs in offering developer-facing edge-native APIs augmented with CSP MEC awareness, as highlighted in the diagram below.

Note that an appropriate level of abstraction will be required, as CSPs may not wish to expose the information in its raw form. Possible abstractions of the information described here are outside the scope of this whitepaper.

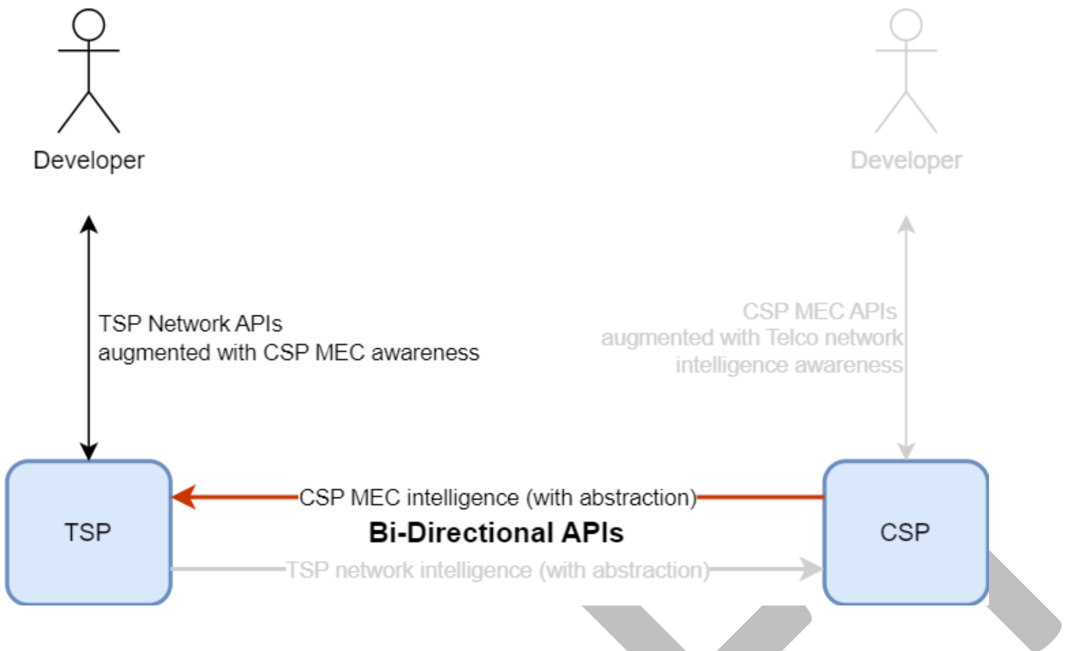


Figure 8: Information Exchange Through B-Directional APIs - CSP to TSP

5.2.1. Edge Discovery Augmented with CSP MEC information

Edge discovery is an intent-based service that a developer triggers to determine the optimal MEC location(s) to place their application. When edge discovery is performed, the response will be a MEC (or multiple MECs) that satisfies the developer intent, encompassing both network and compute requirements of the application.

The placement of a MEC application can depend on what is to be discovered about the edge ecosystem in near real-time. Equally, a placement may fail if the resources are unavailable at the time of attempted placement.

In order to offer an effective edge discovery service, TSPs need an understanding of the CSP Inventory of available compute and available capacity of other CSP-offered services. By combining its network intelligence with MEC inventory and capacity information from CSPs, TSPs will be able to offer developers an edge discovery service that ensures that the qualified MEC location to place the workload has sufficient capacity to accommodate the workload as well as sufficient network capacity and performance to meet the application performance requirements.

5.2.2. CSP Platform Analytics

CSPs generate domain-specific metadata during their operations and sharing this data with TSPs will enable TSPs to enrich their services and improve the user experience. Where it is not appropriate to share raw data (for example, data that is complex or privacy-sensitive), CSPs can expose the results of data analysis which simplifies, abstracts and anonymises the data.

Examples of platform analytics that a CSP may expose include:

- Compute-resource availability and capacity trends;
- Platform services trends (e.g., virtual machine (VM) instances, serverless functions, storage, etc.)

Real-time data analysis is critical for MEC applications, while relevant historical data and statistics that the CSP exposes can play an important role in building applications that adapt to user, infrastructure resource and network behavior based on historic trends and heuristic data models. CSP Infrastructure resource analytics can also assist in correlating network and device analytics to create a complete profile of application usage which helps in scope and capacity planning.

5.2.3. CSP Lifecycle Events

Certain CSP events may trigger a life cycle management action within the TSP domain, such as a CSP compute failure or compute capacity “threshold met” events. Receiving these events from the CSP will enable the TSP to effectively lifecycle manage workloads for which the TSP has taken end-to-end responsibility on-behalf of the developer/customer.

5.2.4. API Based Discovery for CSP Platform Services

As MEC becomes more distributed, there will be variation in the range of cloud services supported across MEC deployments. In order to perform an effective discovery and workload placement, it will be necessary for TSPs to programmatically retrieve information on supported services at a given CSP MEC location, e.g., geographical constraints (resource or service availability) per CSP region.

5.2.5. CSP Policy Discovery

CSP resource usage will likely be subject to usage policies. Since these will differ across CSPs, regions and countries, CSPs could enable the programmatic lookup of such policies including:

- Service quotas (resource constraints including uptime, memory usage, etc.)
- Any other terms of use

6. Use Case Examples

This section describes a series of example Bi-Directional API use cases together with potential interactions and information sharing between the TSP and CSP to enable each use case.

- Section 6.1 describes three use cases from a CSP go-to-market perspective, where the developer's primary relationship is with the CSP.
- Section 6.2 describes three use cases from a TSP go-to-market perspective. In this case, the developer's primary relationship is with the TSP and the TSP then uses CSP MEC infrastructure to deliver services to the developer.

Note that the information flows described in this section are for illustrative purposes only.

Additionally, some interactions are shown that are outside of the scope of Bi-Directional APIs for contextual reasons. For example, the API calls from an end-user's client application to the TSP or CSP to identify an optimal service endpoint for it to connect to at runtime (service discovery). For clarity, the scope of Bi-Directional APIs is indicated with a red shaded box in each diagram.

6.1. CSP Go-to-market Use Cases

This section identifies three potential Bi-Directional API use cases from the CSP go-to-market perspective. It then describes some potential interactions and information sharing between the CSP and TSP to enable each use case.

6.1.1. Use Case CSP-1: Edge Discovery

Description

A developer wishes to deliver a real-time AR/VR experience to its customers at a sports stadium. The application needs a low round-trip latency and a guaranteed bandwidth per end-user for optimal performance.

The developer has a business relationship with a CSP and wishes to leverage edge platform services delivered from the CSP's MEC platform to identify the optimal locations to deploy the application. The CSP's MEC platform leverages TSP network intelligence from one or more TSPs to assist in identifying the optimal MEC platforms for the application, taking into consideration compute and network requirements.

The developer initially goes through a registration process to register a Service Profile with the CSP. As part of this process, the developer defines the specific requirements that need to be met across both the CSP MEC infrastructure and the TSP network in order to provide an optimal experience to its customers. These requirements may include:

- Network performance (e.g., latency, jitter and bandwidth between the end-user and the edge application)
- Compute resources (e.g., CPU, memory, GPU)
- Storage resources
- Scaling headroom
- Client service area (i.e., the application may be offered to end-users only in specific geographies)
- Non-technical constraints (i.e., business or regulatory constraints around where an application can be deployed)
- Pricing preferences (e.g., potentially trading-off some performance for low spot-prices)

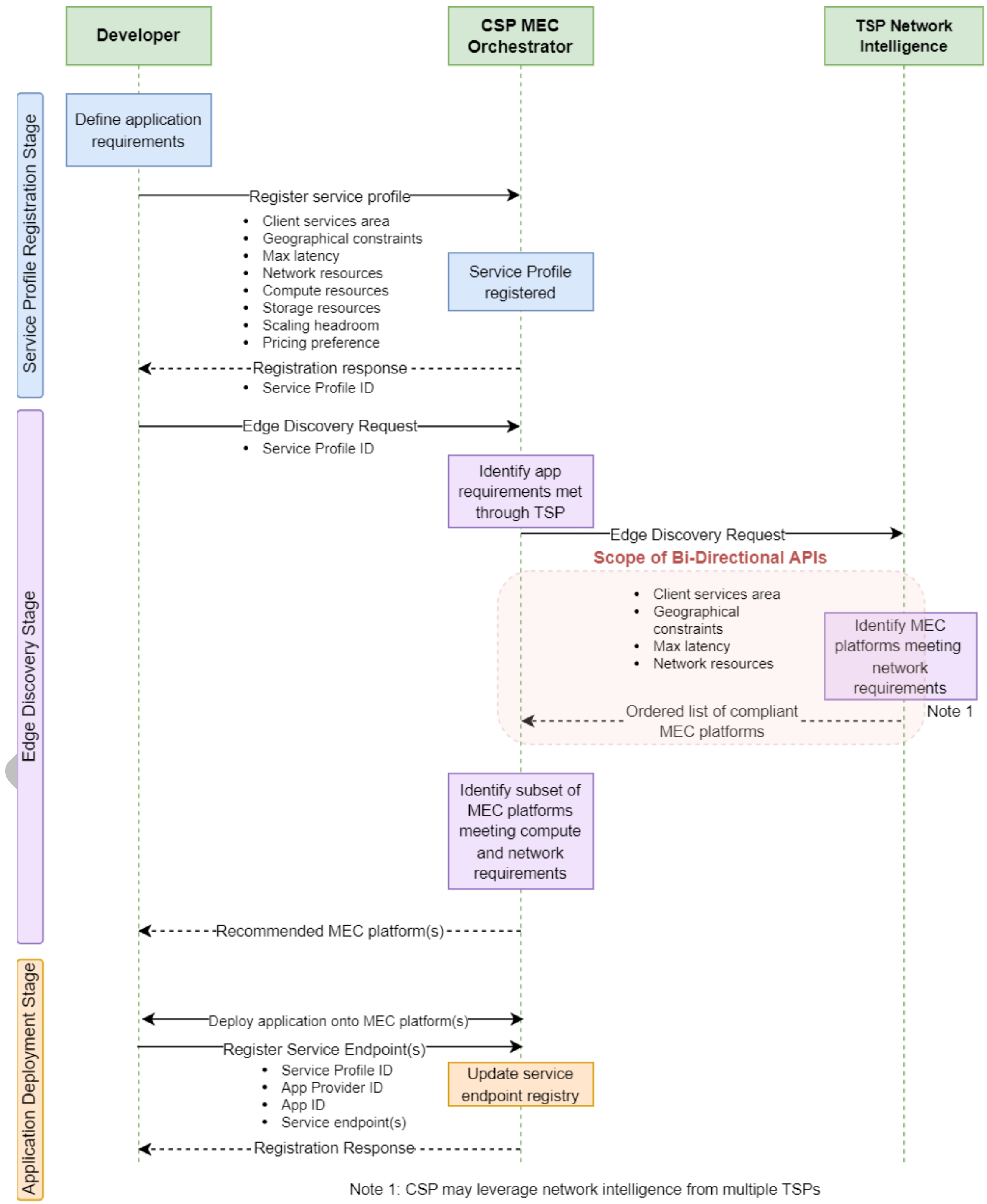
The developer then issues an edge discovery request referencing the Service Profile to the CSP. The CSP API service uses the Bi-Directional APIs with the TSP to discover candidate MEC platform(s) that meet the relevant network requirements and combines this information with its knowledge of its MEC platform capability and capacity (e.g., compute and storage resources) in order to identify the optimal MEC platform(s) on which to deploy the application.

The CSP then returns a list of the optimal MEC platform(s) to the developer.

The developer then deploys the application onto the CSP MEC infrastructure. As part of this process, the developer populates a service endpoint registry consisting of application metadata, including the Service Profile (which defines the application's compute and network requirements) and service endpoint information for each instance of the application (e.g., CSP MEC ID and corresponding app URL for each MEC site at which the app is deployed).

Potential Information Flows

Application Life-cycle Stage	Deployment
Information provided to CSP by TSP through Bi-Directional API	A list of CSP MEC platforms that meet the network requirements relevant to the application (e.g., maximum latency, bandwidth)
Actions taken by CSP	Identify a list of CSP MEC platform(s) that meet the full set of application requirements (including compute and network) and returns the list to the developer



6.1.2. Use Case CSP-2: Service Discovery

Description

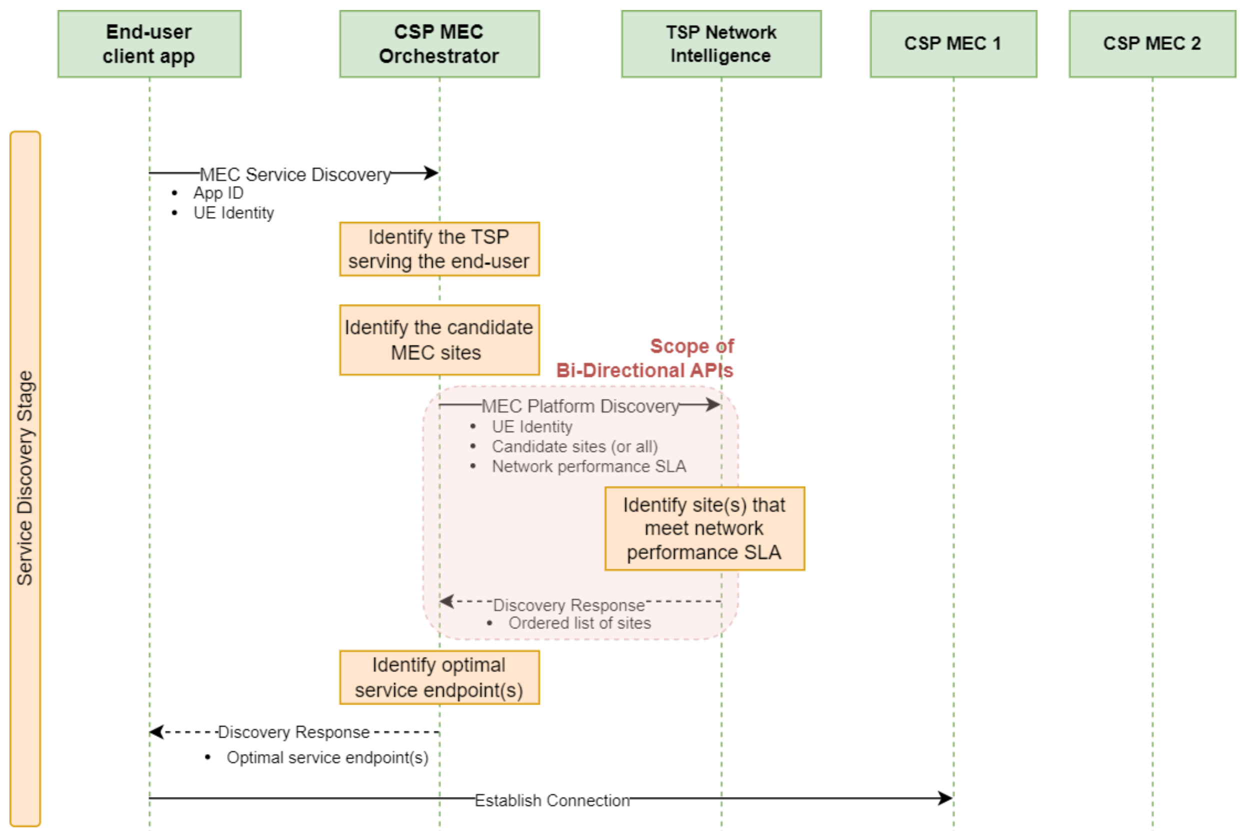
This use case follows on from Use Case CSP-1. It assumes that the developer has registered an application on a CSP MEC platform and deployed the application to a number of CSP MEC sites (CSP MEC 1 and CSP MEC 2 in the following diagram). The CSP maintains a service endpoint registry consisting of application metadata, including the Service Profile (which defines the application's compute and network requirements) and the CSP MEC locations at which the application is deployed.

The CSP provides an abstracted service discovery API to the developer that facilitates connecting its end-user application sessions to the optimal MEC service endpoint based on the application's requirements. In the back-end, the CSP leverages a Bi-Directional API to one or more TSPs to determine which of its MEC platforms can best satisfy the application run-time requirements from a network performance perspective. In the example below, it provides the TSP with an end-user device identifier, a candidate list of CSP MEC sites and the required network performance SLA.

The TSP then leverages its network intelligence to identify the optimal CSP MEC site(s) to serve the end-user from a network connectivity perspective. The CSP then notifies the end-user application of the optimal service endpoint (alternatively, it may provide an ordered list of optimal service endpoints) and the application establishes a connection.

Potential Information Flows

Application Life-cycle Stage	Production
Information provided to CSP by TSP through Bi-Directional API	Identify the CSP MEC sites that can meet the application performance SLA, taking into consideration network intelligence such as end-user location, network connectivity and QoS
Action taken by CSP	Provide the end-user application with the optimal service endpoint(s) to connect to



6.1.3. Use Case CSP-3: End-user Mobility

Description

This use case follows on from the previous Use Case CSP-2. It assumes that the developer has already deployed an application into several CSP MEC locations and one or more end-users have connected to the application at their optimal MEC location. The application has an ultra-low latency performance SLA.

As the application has an ultra-low latency SLA, the end-user client application wishes to be notified if it needs to reconnect to a different MEC site to achieve better performance. In the example below, this may be due to a wireless end device moving a significant distance away from the current serving MEC location, such that it would be better served by a different MEC. To achieve this, the end-user client application subscribes to a mobility event with the CSP.

The CSP then leverages Bi-Directional APIs to create a corresponding mobility event subscription with the appropriate TSP.

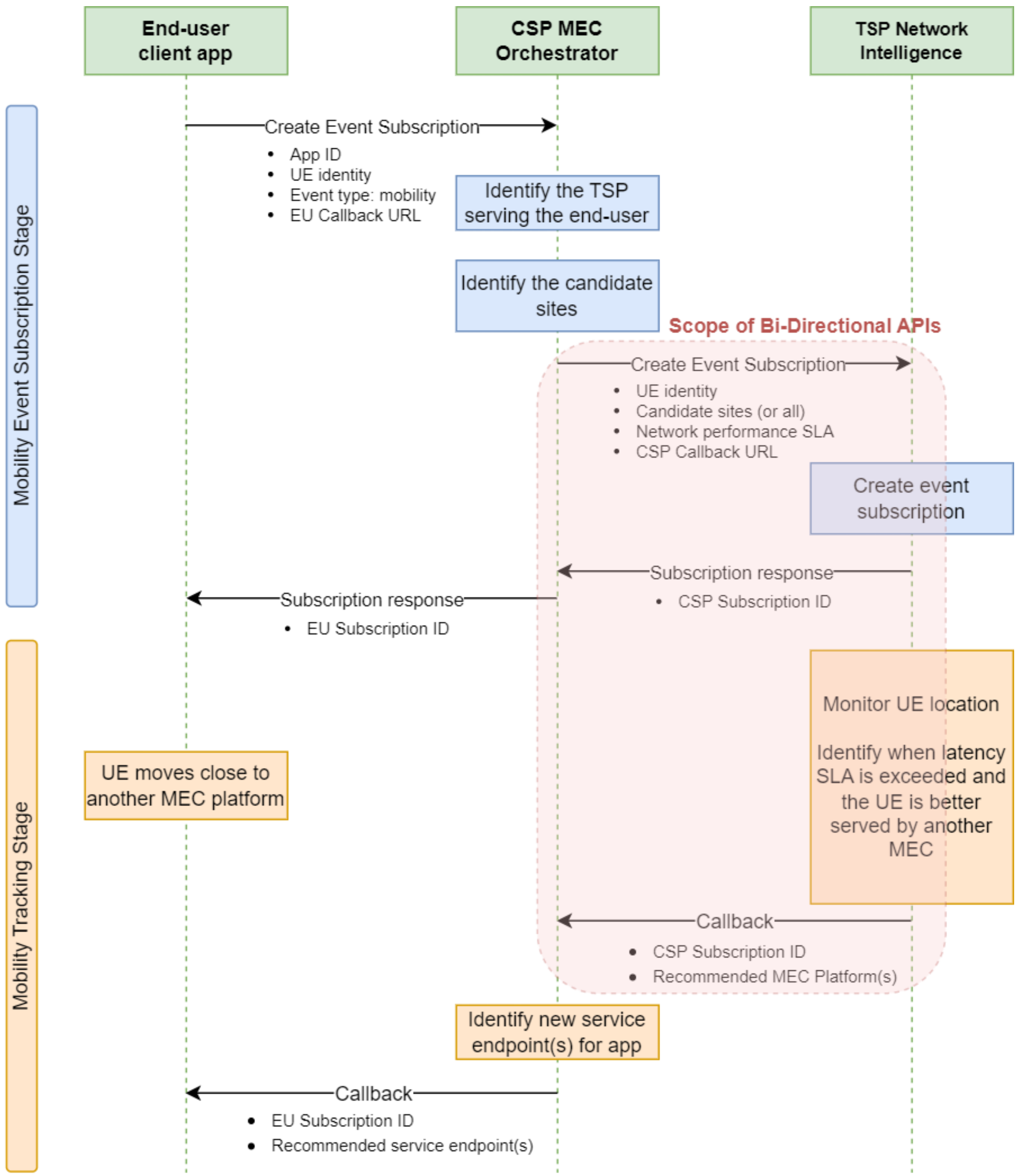
The TSP then monitors the end-user device location as it moves around. At some point, the end-user travels to a new location where the application SLA is no longer being met. The TSP then identifies one or more alternative MEC platforms that can satisfy the SLA and notifies the CSP, which in turn notifies the end-user application. This example assumes that the end-user application will then decide whether to re-home the session and if so, when to do so.

In this example, the developer has registered the MEC sites (or zones) in which the application is deployed with the CSP (see Use Case CSP-1). The CSP may then pass this information on to the TSP to limit the selection of sites to only those sites in which the application is deployed. An alternative approach could be for the TSP to simply advise the CSP when the network performance SLA is no longer being met and for the end-user client application to then make a fresh call to the CSP's service discovery API (as described in Use Case CSP-2).

Note that the interactions associated with moving the end-user network session to an alternative anchor point in the wireless network and migrating any application state to the new application instance is not shown in the diagram below.

Potential Information Flows

Application Life-cycle Stage	Production
Information provided to CSP by TSP through Bi-Directional API	1. Notification of end-user application mobility events that result in the application latency SLA being exceeded. 2. Recommended alternative MEC platform(s) for optimal performance
Action taken by CSP	Notify end-user application of new optimal service endpoint



6.2. TSP Go-to-market Use Cases

6.2.1. Use Case TSP-1: MEC Application Placement Optimisation

Description

Similar to use case CSP-1, a developer wishes to deliver a real-time AR/VR experience to its customers at a sports stadium. The application needs a low round-trip latency and a guaranteed bandwidth per end-user for optimal performance.

In this case, the developer has a business relationship with a TSP and wishes to leverage edge platform services delivered from the TSP's MEC platform to deploy and lifecycle manage its AR/VR application. The TSP's MEC platform leverages CSP MEC compute resources from one or more CSPs to provide its compute services.

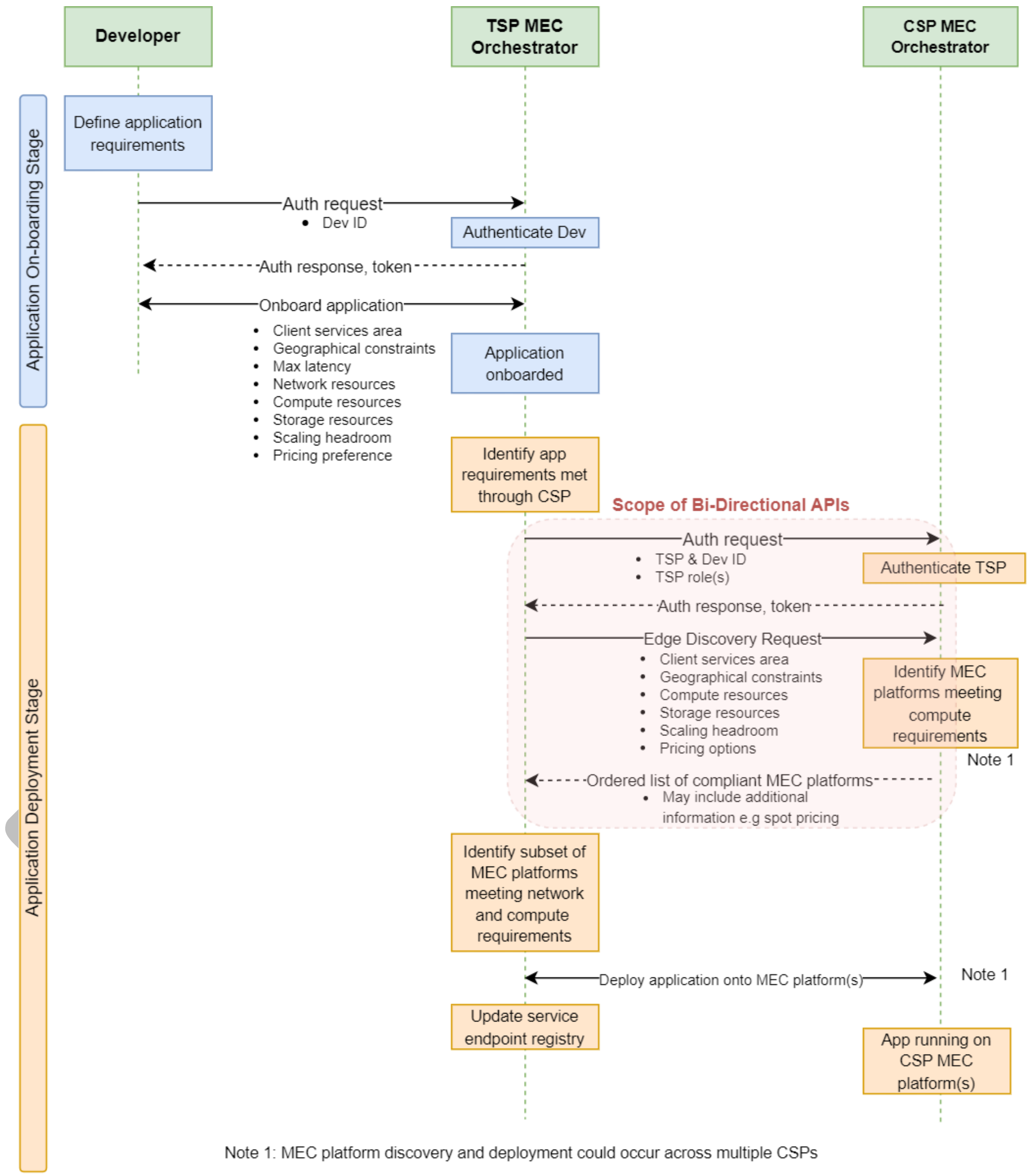
The developer initially goes through an on-boarding process to on-board its application onto the TSP's MEC platform. As part of this process, the developer defines the specific requirements that need to be met across both the TSP network and the CSP MEC infrastructure in order to provide an optimal experience to its customers. These requirements may include (but not exhaustive):

- Network performance (e.g., latency, jitter and bandwidth between the end-user and the edge application)
- Compute resources (e.g., CPU, memory, GPU)
- Storage resources
- Scaling headroom
- Client service area (i.e., the application may be offered to end-users only in specific geographies)
- Non-technical constraints (i.e., business or regulatory constraints around where an application can be deployed)
- Pricing preferences (e.g., potentially trading-off some performance for low spot-prices)

The TSP is then responsible for deploying the application onto the optimal set of CSP MEC infrastructure according to the requirements defined by the developer. It uses the Bi-Directional APIs to discover candidate CSP MEC platform(s) that meet the relevant compute infrastructure requirements (e.g., available compute and storage resources) and combines this information with its network intelligence (e.g., underlying network topology, network capacity, quality-of-service capability) in order to identify the optimal MEC platform(s) on which to deploy the application. After deploying the application onto the CSP MEC infrastructure, the TSP maintains a registry of available service end-points that the end-user client application can connect to.

Potential Information Flows

Application Life-cycle Stage	Deployment
Information provided to TSP by CSP through Bi-Directional API	A list of CSP MEC platforms that meet the subset of application requirements relevant to the CSP (e.g., compute and storage resources, geographical constraints)
Actions taken by TSP	. Identify the set of CSP MEC platform(s) that meet the full set of application requirements (including network-related requirements). . Deploy the application on the optimal CSP MEC platform(s).



Note 1: MEC platform discovery and deployment could occur across multiple CSPs

6.2.2. Use Case TSP-2: Service Discovery Optimisation

Description

This use case follows directly from Use Case TSP-1. It assumes that the developer has on-boarded an application onto a TSP MEC platform, which has subsequently deployed the application across multiple CSP MEC locations (CSP MEC 1 and CSP MEC 2 in the following diagram) and registered the corresponding service endpoints in its registry. In this example, end-users are located in an area that will receive optimal performance when connected to CSP MEC 1, with CSP MEC 2 being the next most optimal.

When an end-user opens the client application on their device (e.g., a smartphone), the client application first queries a service discovery API offered by the TSP in order to identify the optimal service endpoint to connect to. The TSP uses its knowledge of the service endpoints for that application (from the deployment stage in Use Case TSP-01), and combines this with its network intelligence (e.g., current location of the end-user and network topology) to identify which service endpoint will provide an optimal level of performance. It then notifies the client application of the optimal service endpoint (in this case CSP MEC 1), or alternatively, an ordered list of service endpoints. The client application then establishes a connection with the optimal service endpoint.

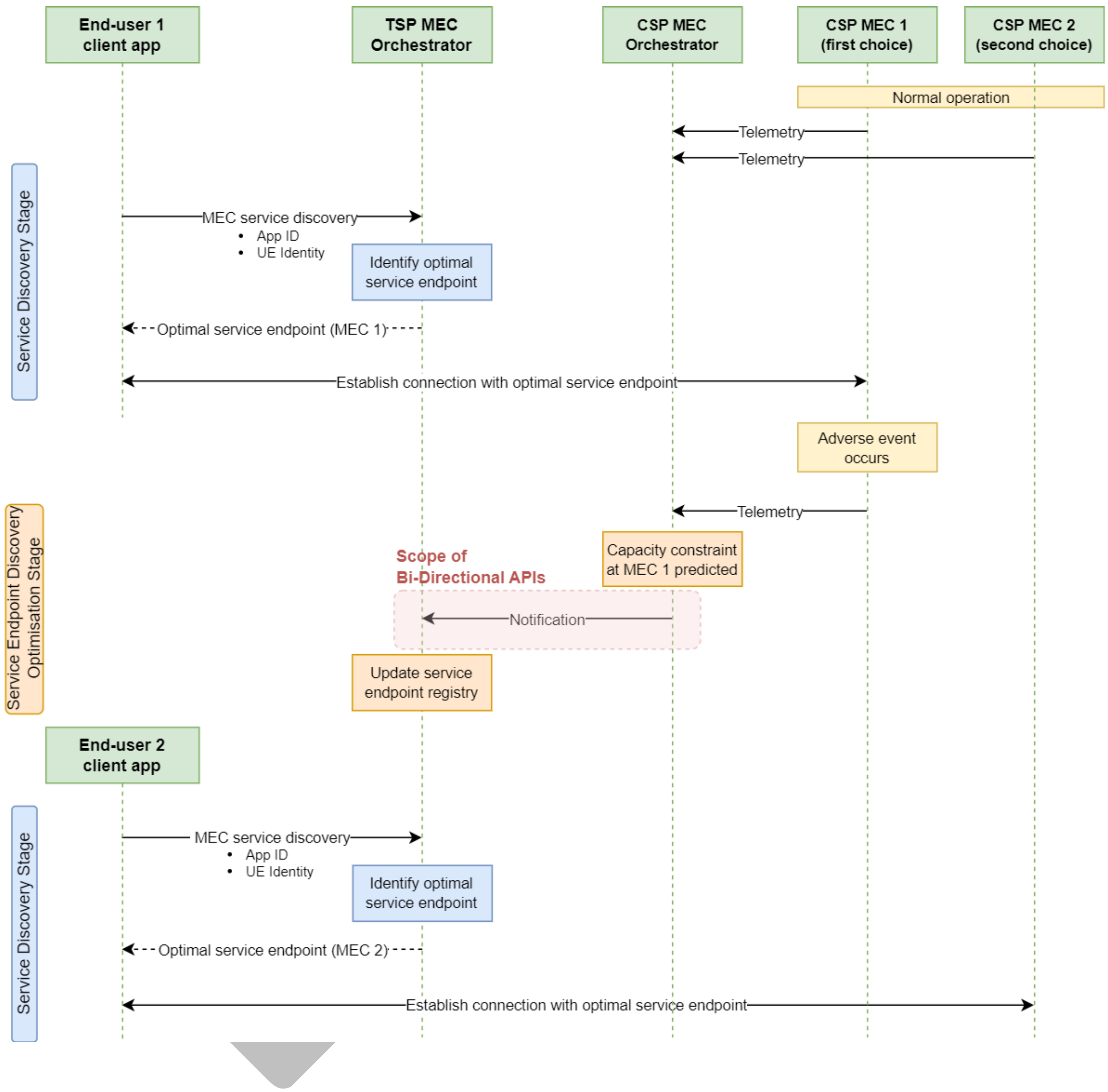
An event then occurs at CSP MEC 1 and some CSP platform analytics identifies the likelihood of impending capacity constraints. This may be due to a range of factors, such as unexpectedly high sustained demand or compute/storage failure reducing available resources. The CSP uses a Bi-Directional API to notify the TSP of this capacity constraint at CSP MEC 1 and the TSP updates its service endpoint registry accordingly.

When subsequent end-users open the client application on their device and the client application queries the service discovery API, the API will now return a new optimal service endpoint (CSP MEC 2 in the example).

Note that a similar flow can apply to other types of events, e.g., a pricing event such as a spot-discount at a nearby edge site.

Potential Interactions

Application Life-cycle Stage	Production
Information provided by CSP through Bi-Directional API	Notification of capacity-impacting adverse event at CSP MEC site
Action taken by TSP Orchestrator	Direct future end-user application service discovery requests to an alternative CSP MEC



6.2.3. Use Case TSP-3: Workload Management

Description

This use case also follows directly from Use Case TSP-1. It assumes that the developer has on-boarded an application onto a TSP MEC platform, which has subsequently deployed the application across a subset of CSP MEC locations (CSP MEC 1 only in the following diagram) and registered the corresponding service endpoints in its registry.

When an end-user opens the client application on their device (e.g., a mobile phone), the client application first queries a service discovery API offered by the TSP in order to identify the optimal service endpoint to connect to. The TSP uses its knowledge of the service endpoints for that application (from the deployment stage in Use Case TSP-01), and combines this with its network intelligence (e.g., current location of the end-user and network topology) to identify which service endpoint will provide an optimal level of performance. It then notifies the client application of the optimal service endpoint (CSP MEC 1 in this case), or alternatively, an ordered list of service endpoints. The client application then establishes a connection with the optimal service endpoint.

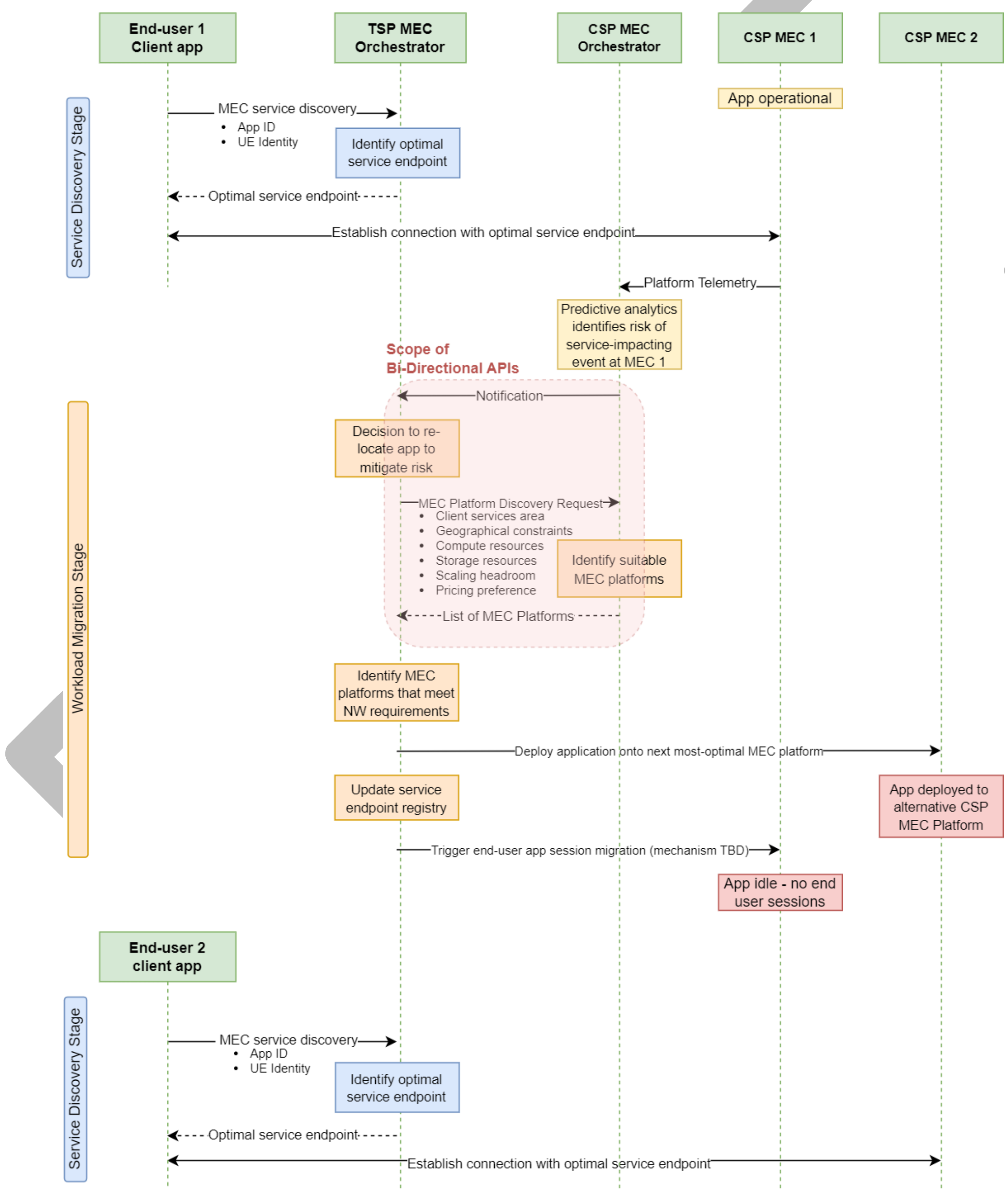
The CSP has implemented a predictive analytics capability, which indicates risk of a service-impacting event occurring at CSP MEC 1. This may be due to a range of factors such as a power outage with limited backup power available, or unusually high demand resulting in capacity limitations. The CSP then uses a Bi-Directional API to notify the TSP of the issue at CSP MEC 1.

The TSP decides to migrate the application to an alternative CSP MEC location to mitigate the risk. It uses a Bi-Directional API to discover candidate CSP MEC platform(s) that meet the relevant compute infrastructure requirements (e.g., available compute and storage resources) and combines this information with its network intelligence (e.g., underlying network topology, network capacity, quality-of-service capability) to identify the optimal alternative CSP MEC platform on which to deploy the application. It then deploys the application to the optimal alternative CSP MEC platform (CSP MEC 2 in the diagram below) and updates its service endpoint registry accordingly. The TSP then triggers end-user session migration to the new CSP MEC platform (note that the mechanism to achieve this is outside the scope of Bi-Directional APIs).

When subsequent end-users open the client application on their device and the client application queries the service discovery API, the API will now return the alternative optimal service endpoint (in the example, CSP MEC 2 will be returned in place of CSP MEC 1).

Potential Information Flows

Application Life-cycle Stage	Production
Information provided to TSP by CSP through Bi-Directional API	. Notification of impending service-impacting event . A list of alternative CSP MEC that meet the application requirements (e.g., compute/storage resources, geographical constraints, etc.)
Action taken by TSP	. Identify the optimal alternative CSP MEC platform (taking into consideration the network connectivity, network topology, QoS requirements, etc.) . Re-deploy application on the optimal alternative CSP MEC platform . Trigger migration of end-user application sessions to the alternative MEC platform



7. Conclusions

The 5GFF envisions an interconnected and interoperable global Mobile Edge Compute ecosystem, fostering widespread adoption and deployment across TSPs and CSPs. However, distributing compute services close to the end-user brings technical challenges in seamlessly weaving the TSP network and CSP compute together to bring MEC use cases to life.

This whitepaper aims to increase industry awareness of the value of bi-directional information exchange between TSPs and CSPs to address these challenges and empower both TSPs and CSPs to deliver innovative MEC services to their customers, at the right location and with the right compute power and network performance to fully take advantage of the capabilities that MEC has to offer.

The 5GFF is working with the Linux Foundation CAMARA initiative to define, develop and test APIs to expose network and MEC services to developers in a way that is easy to consume, even for developers with no telco experience. Bi-Directional APIs will enhance the utility of these developer-facing APIs by enabling TSPs and CSPs to share information about their respective network and compute services and resources, relieving developers of the burden of stitching the network and compute together.

Once industry consensus is reached on the value of Bi-Directional APIs, the 5GFF proposes that a new working group be established within CAMARA to drive Bi-Directional API development.

8. Authors

We would like to acknowledge and thank all of the authors that collaborated to write this paper. If there are any questions, comments, or desire for more information then please [contact the 5GFF](#).

We would especially like to thank select individuals for their contributions:

- **Telstra:** Stephen Hilton, Brett Hill, Senthoor Punniamoorthy
- **Amazon Web Services:** Jim Huang
- **Verizon:** Mahesh Chapalamadugu, Syed Rehman
- **Vodafone:** Kevin Smith
- **Bell Canada:** Luc-Philippe Bouchard

9. References

- 5GFF: https://www.5gff.org/
- CAMARA: https://camaraproject.org/
- GSMA Open Gateway: https://www.gsma.com/futurenetworks/gsma-open-gateway/
- TM Forum Open APIs: https://www.tmforum.org/oda/open-apis/