Introduction

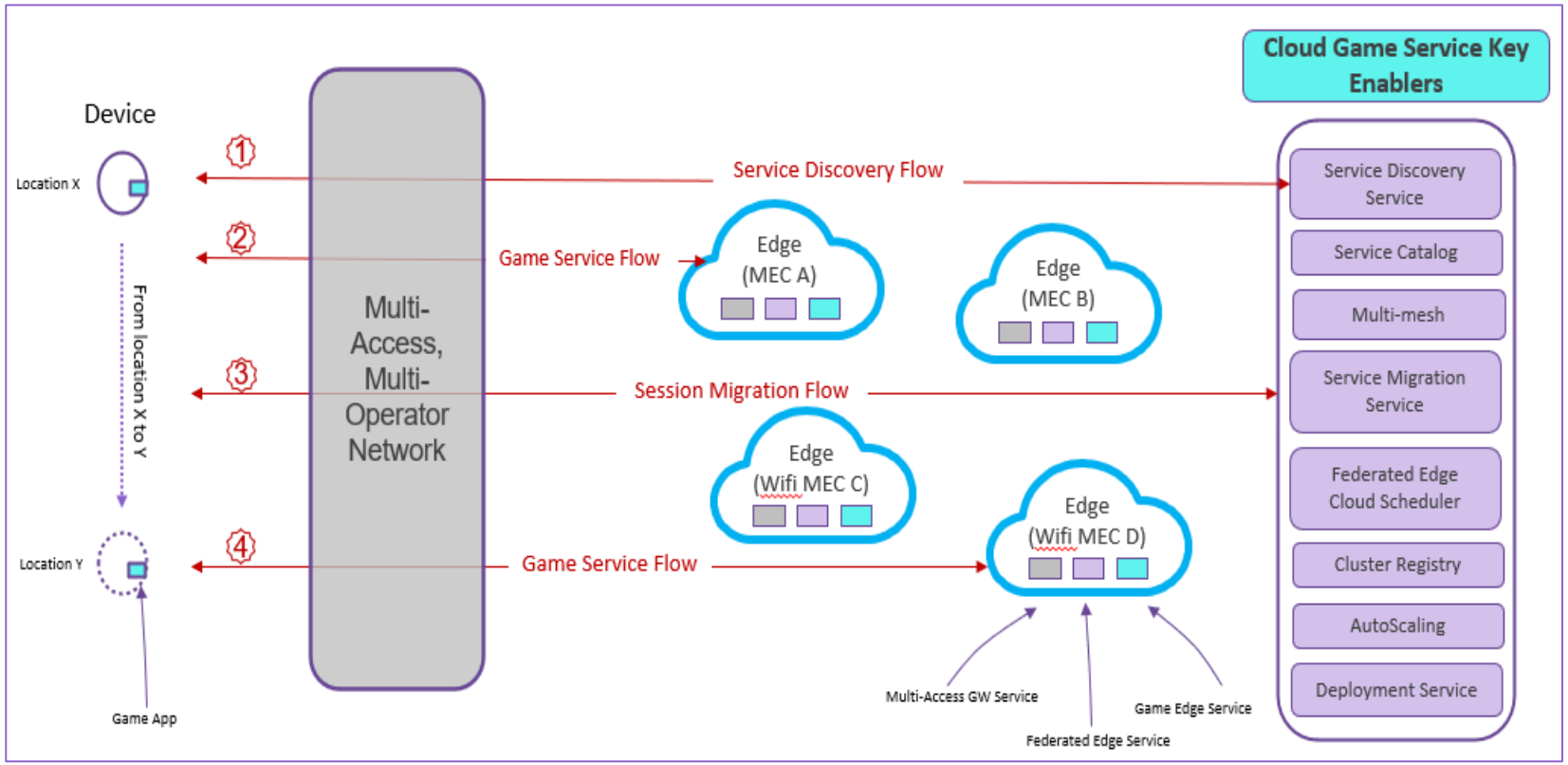
This document provides an overview of the “Federated Multi-Access Edge Cloud Platform” blueprint as part of the Public Cloud Edge Interface (PCEI) blueprint family. The document provides an overview of the key features and implemented components as part of Akraino Release-5.

Overview

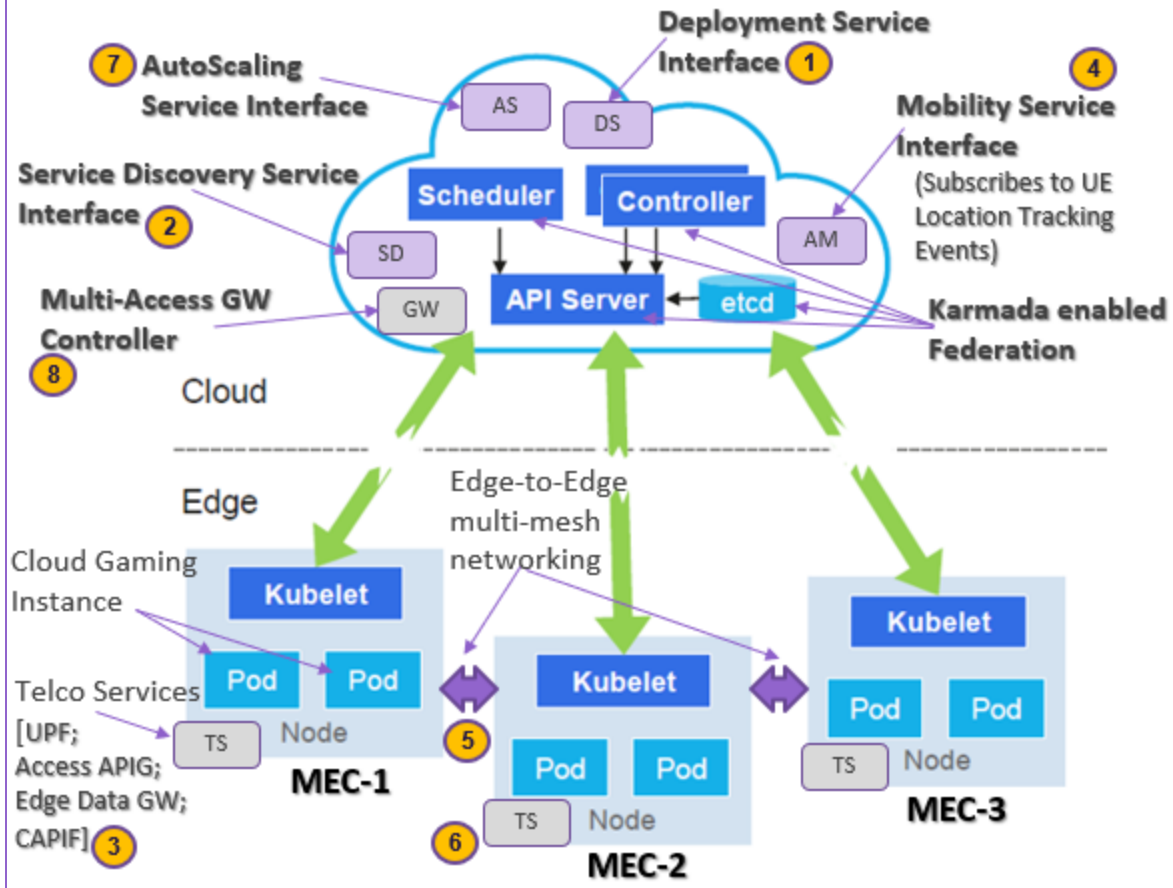
Akraino Blueprint: Federated Multi-Access Edge Cloud Platform

The purpose of this blueprint is an end-to-end solution for mobile game deployed across multiple heterogeneous edge nodes using various network access protocols such as mobile and WiFi and others. This blueprint demonstrates how an application leverages a distributed and multi access network edge environment in order to get all the benefits of edge computing.

Use Case Scenario



Detail Flow of various Architectural Components



- KubeEdge provides the logical MEC station abstraction by using K8S labels to group edge nodes into logical MEC stations. 1
- Operator deploys cloud gaming workload to the specified MEC station/s in accordance to the MEC application placement policies. 1
- UE retrieves the optimal location-aware endpoint address of the edge node (using cloud core side Service Discovery service interface). 2
- UE establishes session to the retrieved edge cloud telco UPF service (provides support for multi-access protocols). 3
- UE connects to the cloud gaming service instance on the edge node.
- Cloud Core side mobility service subscribes to UE location tracking events or resource rebalancing scenario. 4
- Upon UE mobility or resource rebalancing scenario, mobility service uses Cloud core side Service Discovery service interface to retrieve the address of new appropriate location-aware edge node. 2
- Cloud Core side mobility service initiates UE application state migration process between edge nodes.
  - Edge-to-Edge state migration (using east-west multi-mesh networking). 5
- UE connects to new edge telco UPF service. 6
- Redirect UE connection to the new cloud gaming service instance on the new edge node.

Key enabling architectural components and implementations in Akraino Release 5:

- **Federation Scheduler (Included in Release-5)**: As a “Global Scheduler”, responsible for application QoS oriented global scheduling in accordance to the placement policies. Essentially, it refers to a decision-making capability that can decide how workloads should be spread across different clusters similar to how a human operator would. It maintains the resource utilization information for all the MEC edge cloud sites.
- **Multi-Mesh (Included in Release-5)**: Provides support for service mesh capabilities for the edge clouds in support of microservice communication cross cloud and edges.
- **Service Discovery**: Retrieves the endpoint address of the edge cloud service instance depending on the UE location, network conditions, signal strength, delay, App QoS requirements etc.
- **Mobility Management**: Cloud Core side mobility service subscribes to UE location tracking events or resource rebalancing scenario. Upon UE mobility or resource rebalancing scenario, mobility service uses Cloud core side Service Discovery service interface to retrieve the address of new appropriate location-aware edge node. Cloud Core side mobility service subsequently initiates UE application state migration process between edge nodes. Simple CRIU container migration strategy may not be enough, much more complex than VM migration.
- **Multi-Access Gateway:** Multi access gateway controller manages Edge Data Gateway and Access APIG of edge nodes. Edge data gateway connects with edge gateway (UPF) of 5g network system, and routes traffic to containers on edge nodes. Access APIG connects with the management plane of 5g network system (such as CAPIF), and pulls QoS, RNIS, location and other capabilities into the edge platform.
- **AutoScaling**: Autoscaler provides capability to automatically scale the number of Pods (workloads) based on observed CPU utilization (or on some other application-provided metrics). Autoscaler also provides vertical Pod autoscaling capability by adjusting a container’s ”CPU limits” and ”memory limits” in accordance to the autoscaling policies.
- **Service Catalog**: It provides a way to list, provision, and bind with services without needing detailed knowledge about how those services are created or managed.