IPCEI-CIS Reference Architecture

Version 2.0

January 2026

Abstract

This document presents the second version of the IPCEI-CIS Reference Architecture, to be used as a framework for the IPCEI-CIS Integrated Project and guidance for describing, implementing, and running integration clusters and pilots.

TABLE OF CONTENTS

CONTRIBUTORS

IPCEI-CIS Partner	Co-author
SAP	Andreas Schlosser - *Coordinator*
	Florian Müller - *Editor*
TIM	Mauro Boldi - *Coordinator*
	Ignazio Di Natali
	Roberto Querio
	Roberto Procopio
Telefónica	Cristina Santana Casillas
	Antonio Medina Martín
	Alexandre Harmand
	Álvaro Fernández Garrido
REPLY	Giovanni Gregorelli
	Gabriele Lospoto
WestfalenWIND IT	Marco Plaß
E-Group	Antal Kuthy
	Ákos Tényi
	Marcell Zoltay
OpenNebula Systems	Jordi Guijarro Olivares
	Alberto Picón
	Alfonso Carrillo Aspiazu
	Constantino Vázquez
NBIP	Hans Punt
Tiscali Italia	Alessio Manca
	Fabio Catalano
BIT (ECOFED)	Cristiaan Brans - Jansen
	Stefan Kooman
Info Support (ECOFED)	Erwin Kersten
	Vincent van der Winkel
TNO (ECOFED/MISD)	Magiel Bruntink
	Edwin Harmsma
	Bart Kamphorst
	Erik Langius
	Rob Smets
Lindner SE (BiGreen)	Ahmed Chebaane

0. Executive Summary

This document presents an updated version of the Reference Architecture for the IPCEI-CIS integrated project, describing the functional components that it will deliver. In the short term, it is guiding the definition, development, and delivery of first pilot scenarios. These pilots are helping to validate and further improve this *IPCEI-CIS Reference Architecture* (ICRA), adding aspects that may be missing in the initial versions.

The model focuses on the integration and orchestration of infrastructure and workload lifecycle management across a hybrid edge-cloud environment, and includes other aspects like physical infrastructure management, connectivity management, or edge-cloud federation.

Section 1 defines the scope of this Reference Architecture. Section 2, "High-level Overview of the IPCEI-CIS Reference Architecture," describes the layers and domains. Section 3, "Component Description," provides detailed examination of each layer and domain, describing their functional components in depth. Section 4, "Federation," offers a high-level overview of federation across multiple providers. Section 5, "Functional Interfaces," covers the interfaces between layers and domains. Section 6, "Reference Architecture Roles," explains the main personas considered in the ICRA. Section 7 summarizes the document and outlines possible next steps for future versions and adoption of the ICRA.

The Reference Architecture outlined in this document is designed to be robust and scalable, providing a flexible framework that covers the needs of the different verticals and can evolve to meet future demands. It is adapted, usually by streamlining it, to more efficiently fit specific sector needs in the shape of *blueprints*, or instantiations of this ICRA.

1. Scope

This document outlines a Reference Architecture to serve as a foundational framework for the IPCEI-CIS Integrated Project and guide the design and delivery of the first Pilot Scenarios. The architecture aims to position partners involved in the project within a cohesive architecture, ensuring that their roles and contributions are clearly defined and aligned with the overall objectives of the IPCEI-CIS.

The objective of this document is to create an **IPCEI-CIS Reference Architecture** that:

1. integrates the main components with capabilities contributed by IPCEI-CIS partners into a unified structure, providing a high-level framework that effectively positions these components within the overall project, facilitating collaboration and maximizing the impact of each partner's contributions;
2. aligns with and endorses compliance with relevant EU regulations such as GDPR, Data Act and AI Act;
3. guides the positioning of IPCEI-CIS partners within a unified framework, driving the development and delivery of first Pilot Scenarios.

In support of the overall IPCEI objectives to "…perform research, development and innovation (R&D&I) and first industrial deployment (FID) of the software components necessary to establish and operate a distributed, openly accessible and interoperable EU Multi-Provider Cloud-Edge Continuum, thus supporting Europe's digital transformation."[1]

The Pilot Scenarios are proofs of concept to demonstrate the practical application of this Reference Architecture, showcasing the integrated management of infrastructure and workloads in a hybrid edge-cloud environment. They will help to validate the Reference Architecture and further improve it by identifying missing elements or aspects or optimizing others that are already considered.

In this version, the ICRA includes a description of functional components based on our current knowledge and status of the different workstreams. It will continuously be extended in future releases with more detailed descriptions of mechanisms and solutions for integration, exposure and federation, capturing the experience gained with the pilots and advancements within the individual Research & Development projects under the IPCEI-CIS.

[1] https://www.8ra.com/wp-content/uploads/decision-of-the-european-commission-regarding-the-ipcei-cis.pdf

2. High Level Overview of the IPCEI-CIS Reference Architecture

The ICRA aims at clarifying and organizing responsibilities between the different IPCEI-CIS partners. For this purpose, it organizes the functional components in layers, that provide different levels of abstraction and help to manage the complexity of a Multi-Provider Cloud-Edge Continuum, and domains that represent cross-cutting aspects applicable to all layers (e.g. management or security).

Figure 2.1 presents a high-level view of the different layers and domains that compose the ICRA. The architecture includes the following **core layers**:

1. **Application Layer**: It provides the functionality required to design and develop applications and functions and to expose them for use. It includes end-user applications and function catalogs.

2. **Data Layer**: It contains functionalities for data collection, processing and exchange, making them simple, scalable, sustainable, trustable, and distributed over the Cloud-Edge Continuum. These data artifacts can be applied both in end-user applications, by third parties, and internally to optimize the edge-cloud system at different levels.

3. **Artificial Intelligence (AI) Layer**: Its aim is to create a fundamental set of advanced functionalities to a hybrid approach: sets of data and processing resources distributed or centralized depending on the use cases. As with data artifacts, they are also applicable to end-user applications or to internal edge-cloud optimization.

4. **Service Orchestration Layer**: It addresses the integration and management of multiple cloud and edge services and applications in a unified and automated way across diverse multi-provider environments. In the context of multi-cloud and edge ecosystems, service orchestration is essential for handling the complexity of deploying and managing applications at scale, especially when dealing with distributed resources across cloud and edge infrastructures.

5. **Cloud-Edge Platform Layer**: It allows the allocation and lifecycle management of resources over the virtualized Cloud-Edge Continuum and the deployment and chaining of application components to deliver a service over a certain geographical area. It includes components that facilitate and manage the complexity of computing environments that require multiple cloud technologies and providers, enabling interoperability, compatibility, and portability across a Multi-Provider Cloud-Edge Continuum.

6. **Virtualization Layer**: It provides the necessary abstraction over the physical hardware resources (compute, storage, networking) to facilitate their dynamic allocation, usage, and sharing, hiding the heterogeneity of the hardware infrastructure. It provides a virtual runtime environment in which the applications are deployed and executed. This layer is linked to the SDN (Software-Defined Network) or other management systems of public and private networks.

7. **Network Systems, SDN Controllers**: They provide the capabilities to manage physical and virtualized/cloudified networking elements to build network services in the geographically distributed Cloud-Edge Continuum.

8. **Physical Layer**: It includes all the physical hardware resources required to implement the Cloud-Edge Continuum (compute, storage, and networking). It is closely connected to the physical network infrastructure that supports communication among the computing nodes in the continuum and the connectivity of users to that continuum.

In addition to these layers, the architecture includes several **cross-cutting domains** that apply to all of them:

1. **Federation Domain**: It contains the components that provide functionality and mechanism to interconnect different providers in the Cloud-Edge Continuum, in such a way that the customer may get access to and use the services from all the federated providers. Federations formalize the collaboration among multiple providers. Federations may happen at virtualized infrastructure, platform, service, data, AI, and application levels, or over multiple levels simultaneously.

   In each level, specific mechanisms and interfaces are defined to provide the means for collaboration. Federation will be a fundamental pillar for the European Multi-Provider Cloud-Edge Continuum, which will provide for interconnection and seamless operation of the cloud and edge components, even if they are operated by independent providers.

   The Multi-Provider Cloud-Edge Continuum is not static. Provider compositions for a service consumer may change during its life cycle. Customers and providers may switch between data processing services (including cloud and edge services), in line with applicable regulatory requirements (for example, the EU Data Act's rules on switching and portability). Such events require support for porting workloads, data and digital assets, and reduction of technical, contractual and economic obstacles. The Federation domain facilitates these kinds of processes and the assigned responsibilities.

2. **Management Domain**: Effective management of cloud infrastructures necessitates a comprehensive suite of capabilities to ensure the required quality and performance at the different layers.

   Logging and monitoring are foundational, providing detailed records and continuous observation of system activities, which support troubleshooting, security analysis, and performance optimization. Coupled with an inventory of resources and services, as well as alerting mechanisms and fault management, these capabilities enable real-time detection and swift resolution of anomalies and potential failures, to operate the different layers and ensure high availability and reliability.

   Metering, accounting and charging systems are essential for tracking resource consumption and generating accurate invoices, fostering transparency and accountability. Meanwhile, the

management of *Service Level Agreements* (SLAs) ensures that performance and availability commitments are consistently met, enhancing customer satisfaction and trust.

Additionally, fault management processes detect, diagnose, and resolve issues promptly, contributing to the resilience and robustness of the cloud environment. Automation for integration, delivery, verification, testing, and other processes in cloud-edge environments is crucial for efficient and reliable operations. Together, these management capabilities form an integrated framework that supports the scalability and efficiency of cloud services, driving both operational excellence and strategic value.

3. **Security and Compliance Domain**: It plays a crucial role in safeguarding data integrity, ensuring that access is strictly regulated, and maintaining adherence to industry standards and regulations. This domain is fundamental in designing a robust cloud infrastructure that not only protects sensitive information but also enforces policies that prevent unauthorized access and misuse.

   By integrating advanced security protocols, real-time monitoring systems, and compliance frameworks, this domain mitigates risks, responds swiftly to potential threats, and ensures that the cloud environment remains secure and compliant with legal and regulatory requirements. This proactive approach to security and compliance is essential for fostering trust and reliability in cloud services.

4. **Sustainability Domain:** Enhancing energy efficiency in cloud and edge infrastructures is crucial and could be achieved at the physical resource layer through energy-efficient cooling solutions, hardware optimization, and the integration of data processing with renewable energy sources. Smart placement of edge nodes ensures access to renewable energy and efficient cooling. As a side effect to efficient energy usage, smart workload placement may reduce network congestion during times of an excess supply of green energy and times where demand of grey energy is high.

   Additionally, energy-efficient technologies are being developed to optimize virtualization, application placement, and lifecycle management, ultimately reducing the carbon footprint of distributed cloud-edge environments.

   To further environmental sustainability, efforts are being focused on minimizing redundant data exchanges, leveraging AI for energy optimization, and monitoring environmental targets while balancing multi-cloud resource usage. These efforts also include reusing software components, both for managing the overall ecosystem and for developing end-user applications. Efficient data interchange formats are being implemented to reduce resource consumption and mitigate environmental risks. Energy efficiency is enhanced across a Multi-Provider Cloud-Edge Continuum by integrating energy optimization, waste reduction, and autonomous systems for environmental monitoring and protection into the lifecycle management of cyber-physical systems.

These cross-cutting domains may use the capabilities provided by Data and AI layers to optimize the different aspects they deal with.

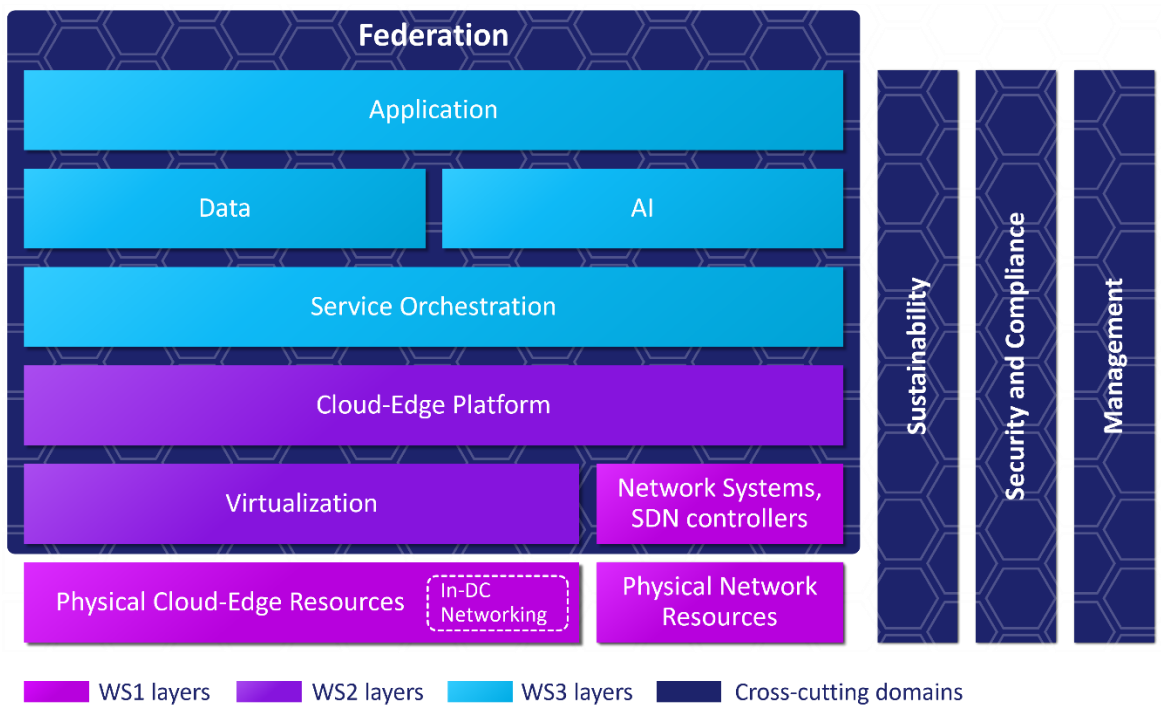


Figure 2.1: Layers and Domains in the ICRA

The layers and components of the ICRA manage different kinds of resources (physical, virtual, platform functions, and services and applications) and provide different kinds of cloud computing services (*Infrastructure as a Service* – IaaS, *Container as a Service* – CaaS, *Platform as a Service* – PaaS, *Software as a Service* – SaaS), as depicted in Figure 2.2. The combination of components providing different services is fundamental in the Cloud-Edge Continuum solutions that aim at being multi-provider, open, and disaggregated.

It has to be noted that the Reference Architecture is modular and that the different components and layers can be merged or segregated based on development approach and commercial need. As an example, not all components may be required for the commercial launch of an edge service, and several functional capabilities/layers may be consolidated in a single commercial product.

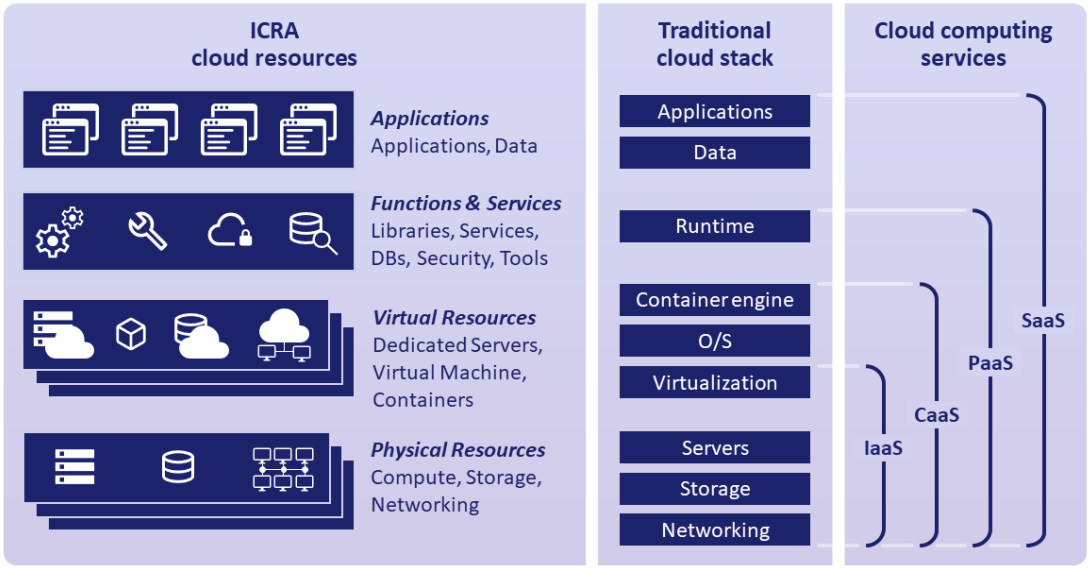


Figure 2.2: ICRA Resources managed by every type of cloud computing service.

3. Component Description

Each layer represents a certain level of abstraction and provides a set of functionalities that can be split into smaller components. The domains represent functional aspects that span across all layers, i.e., the domain components can be found in any of the architecture layers.

This section covers the different domains and layers, describing the components that implement them. In each layer description (subsections 3.5 to 3.11), some examples of domain components are given. The list of domain components in each layer will be more detailed and exhaustive in further releases of this architecture.

It must be noted that this architecture does not limit the possibility that the implementation of a certain layer groups certain functional components into bigger blocks or excludes some of them that may not be needed for a specific scenario. In some scenarios, even complete layers may be merged or skipped in the actual implementation.

The decomposition into components is intended to facilitate the description of the whole system functionality and the participation of specialized providers with innovative solutions in certain functional components of the cloud computing value chain.

It represents neither a reference implementation nor an implementation guideline, but more of a conceptual framework for understanding the integration and interaction of components within the Cloud-Edge Continuum. Industrial implementations may group components in a different way or exclude some of the components in case they are not required for the scenarios they address. For their interoperability, portability and compatibility within the Cloud-Edge Continuum, they will support the relevant interfaces to facilitate integration with other systems (via APIs, manifests, resource descriptors, or other integration artifacts) and the consumption of services (via open standard service interfaces to customers).

3.1. Management Domain

Effective management of cloud infrastructure necessitates a comprehensive suite of capabilities to ensure optimal performance, security, and compliance. Together, these management capabilities form an integrated framework that supports the scalability, efficiency, and security of cloud services, driving both operational excellence and strategic value.

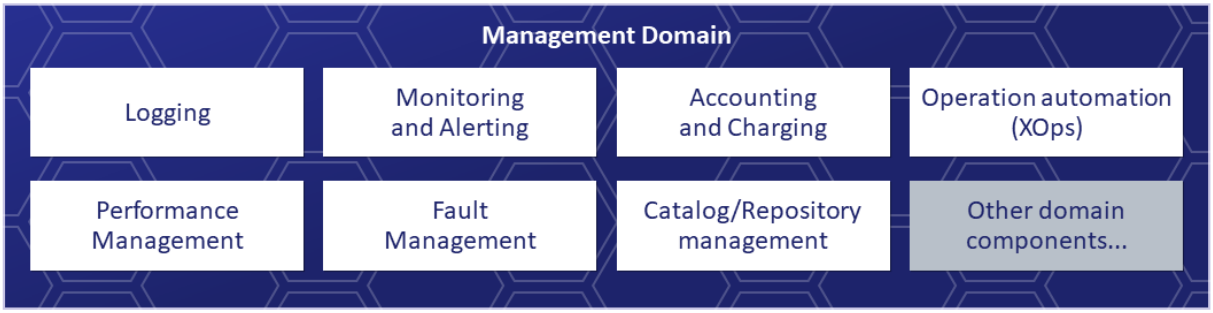


Figure 3.1.1: Components in the Management domain.

Logging

Logging in cloud infrastructures refers to the systematic recording of events, transactions, and activities within the cloud environment. This process captures detailed logs of user actions, system operations, and data interactions, creating a repository of information that supports monitoring, auditing, troubleshooting, and security analysis.

By maintaining comprehensive and accurate logs, cloud providers and users can trace system behaviors, detect anomalies, and swiftly respond to incidents. Effective logging not only aids in compliance with regulatory requirements but also enhances the overall transparency, reliability, and resilience of the cloud infrastructure.

Monitoring and Alerting

Monitoring and Alerting in cloud infrastructures involves continuous observation and real-time analysis of system performance, application behavior, and resource utilization. This process employs various tools and techniques to collect and analyze data from different components of the cloud environment, such as servers, networks, and applications (running on cloud, edge, or far edge devices).

By setting static or AI-supported thresholds and rules, monitoring systems can detect anomalies, performance bottlenecks, and potential failures. When these conditions are met, alerting mechanisms are triggered to notify administrators and stakeholders promptly, enabling swift resolution and minimizing downtime. Effective monitoring and alerting are essential for maintaining the reliability, availability, and overall health of cloud and edge services, ensuring optimal user experiences and adherence to SLAs.

Logging, monitoring, and alerting are foundational, providing detailed records and continuous observation of system activities, which support troubleshooting, security analysis, performance optimization, and pro-active intervention. Coupled with alerting mechanisms, they enable real-time detection and swift resolution of anomalies and potential failures, ensuring high availability and reliability.

Logging, monitoring, and alerting could use artifacts from the Data layer to implement its functionality.

Accounting and Charging

Accounting and Charging in cloud infrastructure and services refers to the systematic process of tracking and invoicing the usage of cloud services by users and applications. Accounting involves the continuous collection of data on resource consumption, such as CPU usage, memory allocation, storage, and network bandwidth. This data is then analyzed to generate detailed usage reports, which serve as the basis for customer billing. Charging systems apply predefined pricing models and rates to the metered data, ensuring accurate and transparent charges based on actual usage. This process not only provides customers with clear insights into their cloud expenditures but also enables providers to manage resources efficiently and optimize their service offerings.

Effective accounting and charging are essential for tracking resource consumption and generating accurate accounting, fostering transparency. This is critical for maintaining financial accountability, fostering trust, and supporting the scalable and on-demand nature of cloud services.

Performance Management

Performance Management in cloud infrastructures refers to the comprehensive process of defining, monitoring, and enforcing SLAs between cloud service providers and their customers. This applies also to internal performance goals the providers may set for their services.

This involves setting clear expectations for service performance, availability, and support, and ensuring that these commitments are met consistently. SLA management includes tracking key performance indicators (KPIs), generating compliance reports, and addressing any deviations through corrective actions. Effective performance management not only enhances customer satisfaction and trust but also enables providers to maintain high standards of service quality and reliability, thereby fostering long-term business relationships and competitive advantage.

The management of SLAs ensures that performance and availability commitments are consistently met, enhancing customer satisfaction and trust.

Fault Management

Fault Management in cloud infrastructures involves systematic detection, isolation, and resolution of faults or issues within the cloud environment. This process includes identifying potential failures, diagnosing their root causes, and implementing corrective actions to restore normal operations. Fault management leverages workflow and ticketing systems combined with automated tools and techniques to monitor system components, analyze error logs, and trigger alerts for anomalies. By proactively managing faults, cloud providers can minimize downtime, enhance system reliability, and ensure continuous service delivery, ultimately contributing to the overall robustness and resilience of the cloud infrastructure.

Fault management processes detect, diagnose, and resolve issues promptly, contributing to the resilience and robustness of the cloud environment.

Catalog/Repository Management

It provides inventory information, in different layers, about resources used and available and services available and deployed to make decisions. For instance, list of *Kubernetes* (k8s) clusters available (with characteristics) and list of applications deployed (in which cluster). In addition, catalogs enable the implementation of a model-driven approach for services and resources.

Operation Automation

This component provides automation for integration, delivery, verification, testing, optimization, and other processes in cloud-edge environments. It allows us to manage distributed software and infrastructure in an automated way (by code, by software, scripting, declaratively) reducing human errors, making implementations more uniform and predictable and facilitating the reconciliation and recovery to a working configuration after a misconfiguration, disaster, or failure. It is usually referred to as XOps (DevOps, MLOps, AIOps, FinOps...).

3.2. Security and Compliance Domain

The Security and Compliance Domain in a cloud environment plays a crucial role in safeguarding data integrity, ensuring that access is strictly regulated, and maintaining adherence to regulations.

In this document, the security and compliance domain is based on the soon to be released *Collaborative Security Framework* (CSF), developed by the IPCEI-CIS Security Workstream. The CSF aims to establish a unified approach to security across the IPCEI-CIS project by fostering collaboration among partners. This framework identifies and standardizes security components, promotes interoperability, and ensures baseline security compliance across all identified 8ra security topics.

This proactive approach to security and compliance is essential for fostering trust and reliability in cloud services.

Since the security and compliance domain requires an all-encompassing view, the CSF is based on 10 major topics listed below (Figure 3.2.1). When applicable, subtopics are already listed.

Identity & Access Management (IAM)

- Decentralized Identity (DID)
- Role-Based Access Control (RBAC)
- Zero Trust Architecture (ZTA)
- Multi-Factor Authentication (MFA) and biometrics
- Identity federation (OAuth 2.0, OpenID Connect)

This domain focuses on controlling and verifying identities and their access to resources across the Cloud-Edge Continuum. It encompasses capabilities like decentralized identity, role-based access control, multi-factor authentication, and identity federation. The CSF will deliver a unified approach for IAM by classifying all partner-contributed identity solutions and mapping them into the Reference Architecture. This ensures consistent application of Zero Trust principles for access, clear alignment of each IAM component to the overall security architecture, and a baseline set of identity services that all partners can leverage and trust.

Network Security

- Firewalls and Segmentation
- VPN and IPsec tunnels
- Secure DNS (DoH/DoT)
- Intrusion Detection/Prevention (IDS/IPS)
- Secure routing and path validation

Network Security covers the protection of data in transit and the defense of network infrastructure. This includes DDoS protection, firewalls, secure network segmentation, encrypted communications (e.g., VPN/IPsec tunnels), intrusion detection/prevention systems, and secure routing protocols. The CSF consolidates and classifies network security components provided by different partners, integrating them into a cohesive multi-provider network defense strategy. By mapping each solution from secure DNS to DDoS mitigation into the framework, the CSF ensures that the entire Cloud-Edge Continuum benefits from a coordinated network security posture, and it identifies any gaps where additional controls or partner contributions may be needed.

Data Security

- End-to-End Encryption (E2EE)
- At-rest and in-transit encryption

- Data loss prevention (DLP)
- Key management systems
- Post-quantum cryptography

Data Security is concerned with protecting data at rest, in transit, and in use. It spans end-to-end encryption, DLP measures, robust key management systems, and emerging techniques like post-quantum cryptography. The CSF delivers classifications of all data protection mechanisms across partners, establishing common standards for encryption and data handling throughout the platform. By mapping partner contributions (e.g., encryption services or key vaults) to this domain, the framework ensures consistent data confidentiality and integrity measures are applied project-wide. It also helps in verifying that each partner's solutions meet compliance requirements for data privacy and that any critical gaps (such as missing encryption capabilities) are addressed collaboratively.

Application Security

- Secure coding practices
- Static and dynamic analysis
- Web Application Firewall (WAF)
- API security
- Software Bill of Materials (SBOM)

This domain addresses the security of software applications and services, including their development and runtime protection. It involves secure coding practices, code analysis (static and dynamic testing), application firewalls like WAF, API security, and the use of SBOM to track components. The CSF maps and classifies all application security tools and practices contributed by partners for example, cataloging code scanning tools or runtime protection modules to ensure that they align with the Reference Architecture's application layer. By doing so, the CSF facilitates a baseline for secure software development and deployment across the project. Partner contributions are integrated so that vulnerabilities are identified early, common security standards like OWASP Top 10 mitigations are followed, and all applications in the IPCEI-CIS ecosystem benefit from a robust and consistent security posture.

Endpoint & Device Security

- Secure OS and firmware
- Device identity and attestation
- Mobile device management (MDM)
- IoT security frameworks

Endpoint and Device Security focuses on protecting the multitude of devices and nodes from user devices to edge hardware and IoT sensors that connect to the cloud-edge environment. This includes secure operating systems and firmware, device identity and attestation mechanisms such as using TPM/HSM hardware roots of trust, MDM, and IoT security frameworks. The CSF classifies partner contributions in this area to build a comprehensive device security layer within the framework. It will deliver an integrated approach where solutions like secure boot processes, remote attestation services, and device management tools are mapped to the appropriate architecture layers. By aligning these contributions, the CSF ensures that all endpoints in the federation, regardless of

provider, meet a common baseline of trustworthiness and that device-level threats are mitigated through coordinated controls and monitoring.

Security Operations (SecOps)

- Continuous monitoring and logging
- SIEM systems – threat detection and response
- Automated incident response
- Red team/blue team exercises

This domain covers the operational aspects of security, including continuous monitoring, threat detection, incident response, and security assurance activities. It involves SIEM systems for centralized logging, alerting, and analytics, automated threat response or SOAR playbooks, and regular exercises (red team/blue team) to test defenses. Through the CSF, partners' SecOps tools and services will be identified and integrated into a unified operational security framework. The CSF delivers a coordinated approach to monitoring and incident handling. Examples include ensuring that logs and alerts from all components feed into a common analysis platform and establishing joint incident response procedures. By mapping each partner's monitoring and response capabilities, the framework improves overall situational awareness and ensures that security events anywhere in the continuum can be detected and addressed quickly through collaborative efforts.

Governance, Risk & Compliance

- Security policy management
- NIST/ISO compliance
- Risk assessments
- Privacy regulations (e.g., GDPR, CCPA)
- Audit logging and forensic readiness

Governance, Risk and Compliance (GRC) encompasses the policies, processes, and oversight needed to manage security risks and ensure compliance with standards and regulations. This includes security policy management, risk assessment methodologies, audits and reporting, and adherence to frameworks like ISO 27001 or industry-specific standards. Within the CSF, all partner contributions related to governance and compliance, such as policy frameworks, audit tools, or compliance checkers, are cataloged and harmonized. The CSF delivers a baseline security policy framework developed in collaboration with all partners, aligning everyone with common compliance requirements, for example, GDPR for privacy or national cloud security guidelines. By classifying each component's compliance posture and mapping it to the relevant controls, the framework helps identify any regulatory gaps and ensures that risk management is a shared responsibility. This unified approach to GRC builds trust among partners and stakeholders by demonstrating that the entire platform adheres to high security standards and documented best practices.

Resilience & Availability

- DDoS mitigation
- Redundancy and failover
- Disaster recovery planning
- Chaos engineering
- SLA for uptime

This domain focuses on keeping services reliable and available even under adverse conditions. Key aspects include DDoS mitigation strategies, redundancy and failover mechanisms, disaster recovery planning, and even techniques like chaos engineering to test system robustness. The CSF integrates and classifies all measures related to resilience contributed by partners, delivering a comprehensive continuity strategy for the project. This involves mapping out how each partner's components achieve high availability, for example, identifying which services have built-in failover or backup, and how they interconnect across providers. By coordinating these strategies, the CSF ensures that critical cloud-edge services maintain agreed SLAs for uptime and that recovery procedures are in place. The framework will highlight any weaknesses in continuity, such as single points of failure and facilitate joint improvements, thereby enhancing the overall reliability of the multi-provider environment.

Privacy & User Control

- Privacy-by-design principles
- Consent management
- Differential privacy
- Data minimization practices

Privacy and User Control is dedicated to protecting personal data and upholding user rights across the system. It includes enforcing privacy-by-design principles, managing user consent and data preferences, implementing data minimization, and techniques like differential privacy for data analysis. The CSF delivers a consolidated view of how each partner's solutions address privacy concerns, classifying these measures and ensuring they are embedded into the architecture from end to end. All partner components will be mapped against privacy requirements, for instance, whether they properly handle consent or anonymize user data, and the CSF identifies any gaps where additional controls are needed. By collaboratively developing a privacy questionnaire and a baseline aligned with regulations like GDPR, the framework guarantees that user data is handled transparently and that users maintain control over their information. This unified stance on privacy fosters user trust and compliance with legal mandates throughout the IPCEI-CIS platform.

Emerging & Future Threats

- Quantum-resistant encryption
- AI-driven threats
- Autonomous agents and smart contracts
- Satellite and mesh network security
- Biological and neuro-network interfaces

This forward-looking domain addresses the need to anticipate and guard against evolving security challenges. It covers preparation for threats on the horizon such as quantum-computing attacks and the need for quantum-resistant encryption, AI-driven or autonomous attacks, vulnerabilities in novel technologies like smart contracts or satellite/mesh networks, and even potential bio/neuro-technology risks. In the Emerging and Future Threats domain, priority should be given to building federated threat intelligence capabilities that empower every IPCEI-CIS participant to proactively recognize and respond to existing and novel risks. Secure data sharing agreements and trust frameworks must enable partners to exchange timely threat information without compromising sovereignty. This approach will turn isolated insights into coordinated threat defense and establish a

unified strategy to anticipate and counter new cyber threats across the European cloud-edge ecosystem. The CSF coordinates research and development efforts among partners to identify and classify solutions for these emerging threats, ensuring that the security architecture remains adaptive. Concretely, the CSF maps partner contributions, such as experimental quantum-resistant cryptography implementations or AI-based threat detection tools into the framework. By doing so, it delivers a strategy for continuous innovation in security: as new threats are identified, the framework can evolve by incorporating new controls or best practices. This collaborative approach means that the project is not just reacting to current threats but actively preparing for future challenges as a unified front.

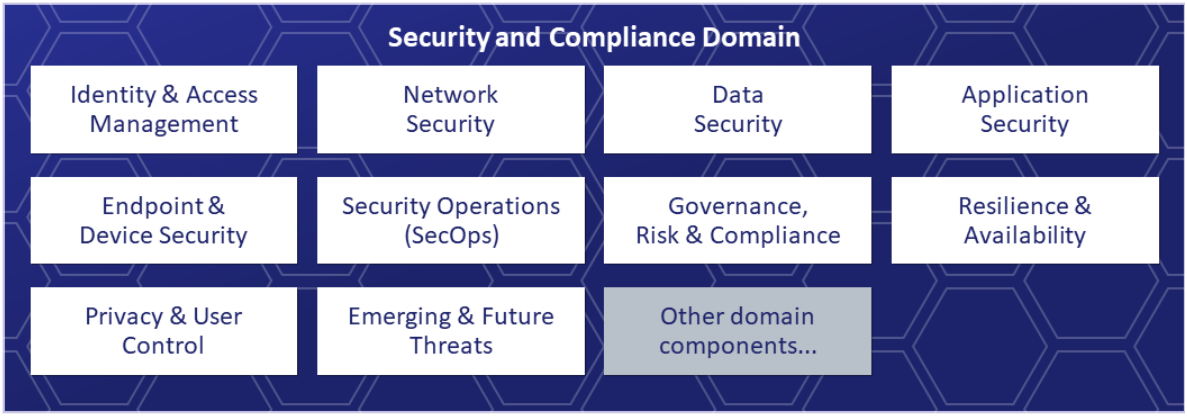


Figure 3.2.1: Components in the Security and Compliance domain.

3.3. Sustainability Domain

The sustainability components in the ICRA enable significant gains in energy efficiency of the Cloud-Edge Continuum. In general, the promise of edge computing is to reduce the amount of data transferred between edge and cloud data facilities and thus reduce energy consumption stemming from data transmission. However, each layer incorporates specific components that enable the sustainability improvements of the next-generation of advanced cloud infrastructure and services, as shown in Figure 3.3.1.

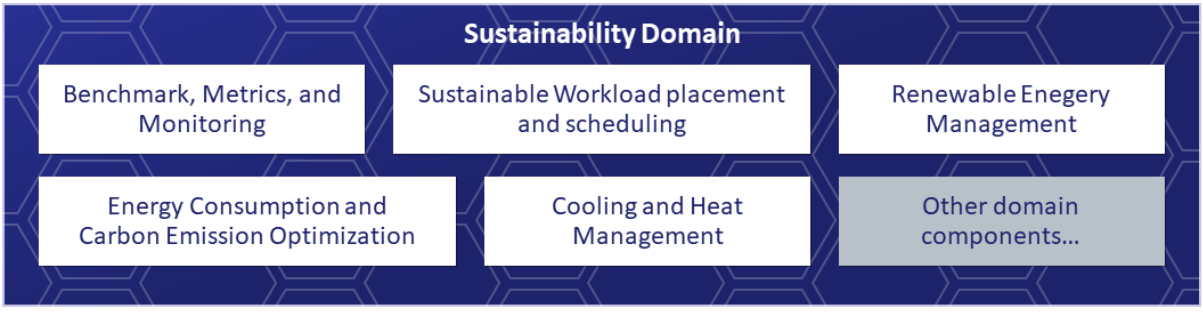


Figure 3.3.1: Components in the Sustainability domain.

Benchmark, Metrics & Monitoring

It provides comparisons of energy consumption among application instances, services, sites, or hardware elements to identify bad or good performers and build categories of them.

It also compares energy consumption needs of applications/workloads with low-cost/renewable energy availability to help in identifying potential energy optimizations based on moving workload to places where energy is free or low-cost.

Both energy comparisons lead to a choice of metrics. They facilitate identifying impact areas.

It tracks in real-time the energy consumption of workloads, hardware elements (servers, storage, networking), sites, and services, and the energy generation of associated renewable energy sources. It also monitors the resource consumption per service or application, the waste and heat generation, and other potential environmental protection measures.

This component sets the standards for monitoring services required to support environmental targets and energy cost reduction as well as financial optimization of 8ra services and infrastructure

These data are intended for use by the optimization component for forecasting, planning and decision making around energy and resource consumption.

It identifies when applications, hardware elements, nodes or facilities exceed certain thresholds in terms of energy or resource consumption, triggering alerts.

The monitoring component also tracks the availability and cost of energy in energy generation locations, identifying those where free or low-cost energy is available. These are energy producers or consumers with excess energy capacity that otherwise would be wasted if not consumed, like wind turbines, solar power plants, and data centers.

The component also gives recommendations for data collection frameworks and tools.

Energy Consumption and Carbon Emission Optimization

This component implements mechanisms to optimize resource and energy consumption and carbon emission of infrastructure or hardware, for instance, scaling down the resources allocated to a certain application when they are not required, and it is not foreseen that they will be needed in the short term. It is in charge of implementing a “green” operational mode for cloud-edge nodes and platforms, down to managing and exchanging product carbon footprint data across supply chains.

It provides recommendations for optimization of application and infrastructure setup based on data from benchmark and monitoring.

This component may also provide recommendations with respect to application placement to reduce data transfer, optimize energy consumption, or balance occupation among sites (reduce risk of congestion).

At the data layer, it advises on data fusion and data reuse options to minimize redundant data exchange.

Sustainable Workload placement and scheduling

It generates recommendations (conventional or AI based) of temporal and spatial workload movement to balance energy costs, performance, and/or carbon emissions. Recommendations are e.g. oriented towards moving workloads temporarily to energy generation sites where energy is going to be wasted (free or low-cost). For this, it forecasts and matches workload requirements with server capacity at these sites with excessive energy.

It leverages datacenter properties (e.g. PUE) algorithmically to maximize energy efficiency and resource optimization and minimize environmental impact.

Workload placement considers also the amount of data communication and latency and cost of migration when selecting the optimum location.

The scheduler plans and implements the recommendations of the resource optimization component, moving workloads e.g. ai training, fine-tuning, or inferencing temporarily to low-cost energy generation locations, and providing workload execution assurance. This component should balance the cost of moving workloads (e.g. data transfer or added latency) with the sustainability benefits.

The implementation is integrated into or used by a (de-)central *Multi-Cloud Orchestrator* (MCO), that is the component able to move workload between sites.

Renewable Energy Management

In general, the purpose of this component is to ensure that the optimal amount of renewable (green) energy is used for powering datacenters. It gives recommendations for local energy production (e.g. wind, sun, as well as traditional energy sources), placement of datacenters, and integration of battery storage systems to balance grid capacity.

Service providers e.g. for training and fine-tuning compile their own energy mix of those classes to use or purchase resources, location, and report on their choice.

Cooling and Heat Management

The cooling and heat management component enhances energy and operational efficiency in many ways, by 1. integrating next-generation cooling technology and real-time optimization of cooling and workloads, 2. enabling heat reuse, and 3. leveraging local environmental opportunities for data center cooling.

1. Efficient next-generation (immersion and liquid-based) cooling technologies to increase cooling efficiency, to reduce energy consumption, and to support high-density computing.
2. Functionality to enable recovery, upgrade, and reuse of heat energy from cooling systems within data centers, and the subsequent storage and exploitation of the recovered heat.
3. Functionality to support the selection of data center locations to benefit from natural environmental factors such as cool climates and other natural cooling opportunities.

3.4. Federation Domain

Federation in cloud infrastructures refers to the collaboration and seamless integration between cloud environments belonging to different organizations (private cloud owners or public cloud

providers), implementing, e.g. what ISO defines as “inter-cloud computing”[2], but there will also be different notions and implementations, as reported in Section 5.

This domain enables interoperability and resource sharing while maintaining autonomy and security across each participating entity. Through standardized protocols and interfaces, federation allows the pooling of resources, data, and services, facilitating enhanced scalability, efficiency, and innovation.

This approach addresses concerns regarding data sovereignty, compliance, and governance, ensuring that each entity can leverage combined cloud capabilities without compromising their individual policies and control.

Federation supports diverse use cases such as cross-cloud data analysis, collaborative applications, and distributed AI scenarios, driving forward the capabilities of modern cloud ecosystems.

Figure 3.4.1: Components in the Federation domain.

The Federation Manager, Federation Broker, and details on the federation concept are in Section 5.

3.5. Application Layer

It provides the functionality required to design and develop applications and functions and to expose them for use. It includes end-user applications and function catalogs.

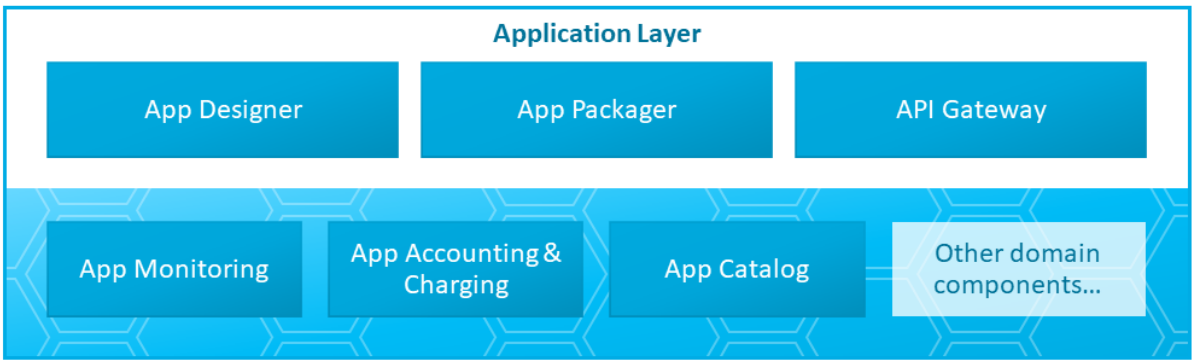


Figure 3.5.1: Components in the Application Layer. In blue background, examples of components from management domains associated to that layer.

[2] ISO/IEC22123-2: “*A CSP [cloud service provider] that uses one or more cloud services provided by other CSPs is said to be in an **inter-cloud** relationship. The CSP using the cloud services is referred to as the primary CSP while a CSP whose cloud services are being used is referred to as a secondary CSP. The purpose of the relationship is to jointly provide cloud services to a CSC [customer]. The use of inter-cloud computing can be invisible to the CSC who thinks he is only using the cloud services from the primary CSP even though some of those cloud services are provided by a secondary CSP*”

Application Designer

This component enables developers to design, create, and customize applications using intuitive interfaces or predefined templates. It facilitates rapid development, integration, and delivery of tailored applications using automated CI/CD practices (DevOps).

The Application Designer facilitates the description of the application in terms of: set of application components it is made of and how they are connected (service function chain), runtime environment that each of the application components will require, including the set of functions/services to support its execution, the attributes that may allow the selection of the computing node to host it (hardware requirements, latency, privacy, etc.).

It also provides tools for automatic verification and validation (CV/CT) of the application and its supply chain before its final packaging.

Application Packager

The Application Packager supports the packaging of applications for their deployment in the Cloud-Edge Continuum.

It facilitates the automation of application deployment and update (DevOps, both traditional and AI-assisted), providing an integrated toolkit that enables quick, secure and innovative ways to deploy cloud-aware applications.

API Gateway

This component provides the interface to invoke and use the applications contained in the catalog. It checks the identity and authenticates the user and checks his authorization to use the application before providing access to it.

Application Monitoring

It tracks application usage and execution, monitors the performance and identifies abnormal behavior and suboptimal use of resources.

Application Catalog

It implements a directory of applications and functions that the providers have made available. contain the characteristics of the application and the environment it requires for its execution (runtime, services, hardware characteristics).

Application Accounting and Charging

This component implements the accounting of application usage and provides online charging information for the customer to track application expenditure in real-time.

3.6. Data Layer

This layer offers simple, scalable, sustainable, and trustable mechanisms for collection, processing, and exchange of data distributed over the Cloud-Edge Continuum.

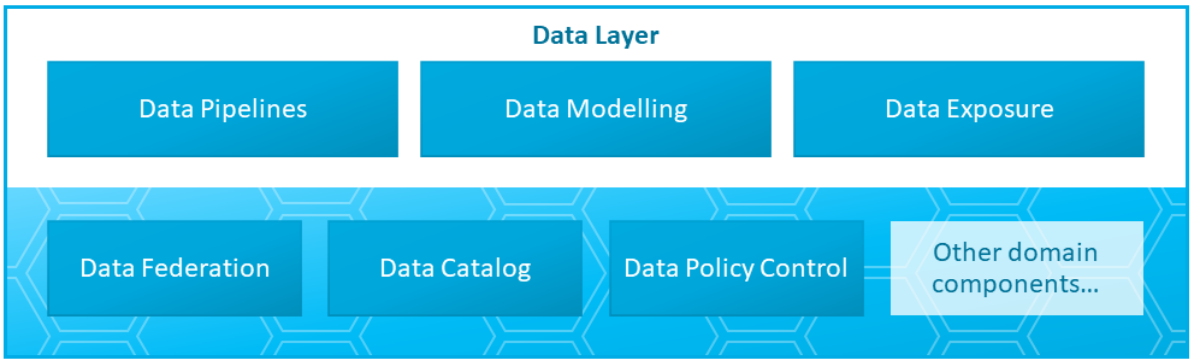


Figure 3.6.1: Components in the Data Layer. In blue background, examples of components from federation and management domains associated to that layer.

Data Pipelines

This component provides the functionality for data collection, including the connectors to integrate with the data sources and the capabilities for data curation and pre-processing that ensure its quality and readiness for analytics, insight generation, training, modelling, or inferencing phases.

Data Modelling

This component enables data cataloguing to enable exposure and discovery at scale to easily search, find, and browse data, over a distributed environment.

Data Exposure

The *Data Exposure* component provides customers with standard mechanisms and interfaces for safe and controlled access to data. It includes capabilities for making data offers and contracting data acquisition, identity checking, and data access authentication and authorization.

Data Policy Control

Data Policy Control sets the required policies for data sharing, providing a safe, controlled and regulation-compliant environment for data exchange. It allows the data owner to manage the permissions to access its data: who can make it, at which conditions and for which purposes.

Data Catalog

Data Catalog provides efficient storage and indexing of data to facilitate browsing, searching and finding data over a distributed environment.

Data Federation

Data Federation enables standard mechanisms and interfaces (connectors) for partnering in the provision of datasets, providing a unified view of data catalogs and databases from multiple data providers.

This component enables real-time data exchange across companies using data mesh principles, connecting distributed and heterogeneous actors over the Cloud-Edge Continuum, keeping data owners in full control of their data.

In order to create and maintain a coherent federated Multi-Provider Cloud-Edge Continuum, data federation capabilities should be designed consistently with the other federation capabilities described in this document.

3.7. AI Layer

The AI layer provides a set of advanced functionalities to apply *Artificial Intelligence* (AI) on distributed sets of data and processing resources complementing the centralized cloud-based solutions.

This layer facilitates the seamless integration of AI model lifecycle management into the Cloud-Edge Continuum, enabling efficient operations across distributed environments. Its capabilities are tailored to the specific characteristics of the Cloud-Edge Continuum, ensuring optimal performance and resource utilization. It provides:

- end-to-end solutions for the AI lifecycle, including preprocessing, model training, and evaluation
- inference with deployment, monitoring, operations across cloud and edge, and
- agents with tools and workflows for coordinated AI execution.

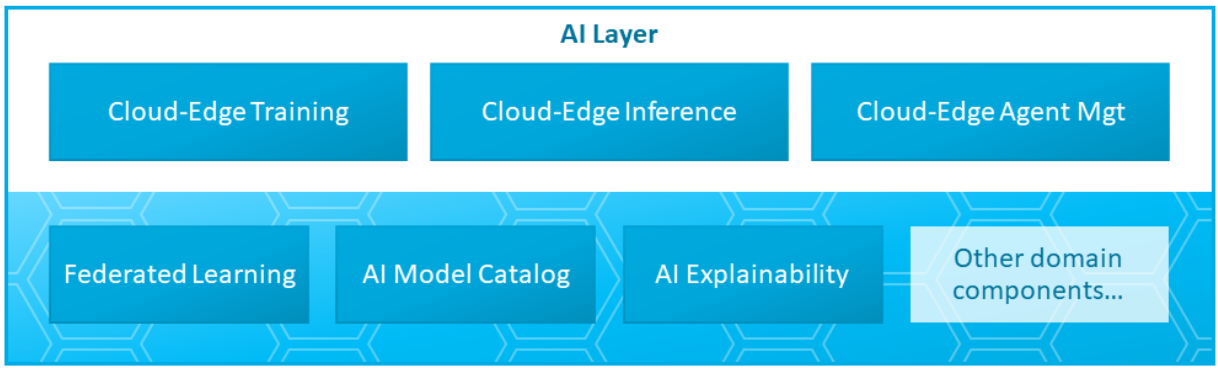


Figure 3.7.1: Components in the Intelligence Layer. In blue background, examples of components from federation, management and security domains associated to that layer.

Cloud-Edge Training

This component facilitates the dynamic and adjustable training of AI models across cloud and edge environments, ensuring scalability, reduced latency, and optimized resource utilization.

Cloud-Edge Inference

The inference components facilitate real-time deployment and execution of trained AI models on edge devices with efficient synchronization with the cloud for updates, monitoring, and enhancements.

Cloud-Edge Agent Manager

The Cloud-Edge Agent Manager enables the deployment and management of agents and agentic workflows on edge and hybrid edge-cloud deployments creating an agentic mesh.

AI Model Catalog

This component contains trained foundational models: LLMs, SLMs, multimodal LLMs, in multiple languages and managing multiple data types: text, images, video, code, etc.

These models provide support for *Natural Language Processing* (NLP), *Machine Translation* (MT), speech processing, text analysis, information extraction, summarization, or text and speech

generation. They can be fine-tuned and adapted to specific use cases, using techniques like RAG, model quantization, pruning, or distillation.

The catalog contains multilingual and multimodal LLMs tailored to diverse EU languages, capable of understanding and processing diverse data types, including text, images, and multimedia. These models address the scarcity of generative AI solutions in non-English languages, ensuring semantic precision, completeness, and compliance with the AI Act.

AI Training Federation

AI workloads can be split across multiple nodes with central orchestration for scalability and efficiency (distributed AI). AI federation enables autonomous nodes to collaborate securely, ensuring privacy and sovereignty. Together, they balance task-sharing efficiency with autonomy.

In distributed AI training, the AI model is generated at a central point based on the combination of models produced by different training agents distributed across an ecosystem of federated AI service providers or owners. The distributed training agents work locally on local datasets, reducing the need to transfer data to a central location for training.

This component allows to use and orchestrate AI resources across multiple providers to collaboratively perform a specific machine learning training task. It leverages a federated network of AI capabilities geographically distributed across the Multi-Provider Cloud-Edge Continuum, enabling seamless resource sharing and scaling while maintaining sovereignty and compliance. It ensures efficient distribution of AI computational workloads, minimizes data movement, and facilitates parallel model training without requiring centralized data aggregation, thus preserving data privacy and autonomy while enhancing overall system performance.

In order to create and maintain a coherent federated Multi-Provider Cloud-Edge Continuum, federated learning capabilities should be designed consistently with the other federation capabilities described in this document.

AI Explainability

This explainable AI component ensures transparency by providing interpretable insights into AI decision-making processes. It supports compliance, accountability, and trust by enabling users and regulators to understand, audit, and validate AI models while respecting privacy and data sovereignty.

3.8. Service Orchestration Layer

The service orchestration refers to the integration and management of multiple cloud and edge services and applications in a unified and automated way across diverse multi-provider environments. In the context of multi-cloud and edge ecosystems, service orchestration is essential for handling the complexities of deploying and managing applications at scale, especially when dealing with distributed resources across cloud and edge infrastructures.

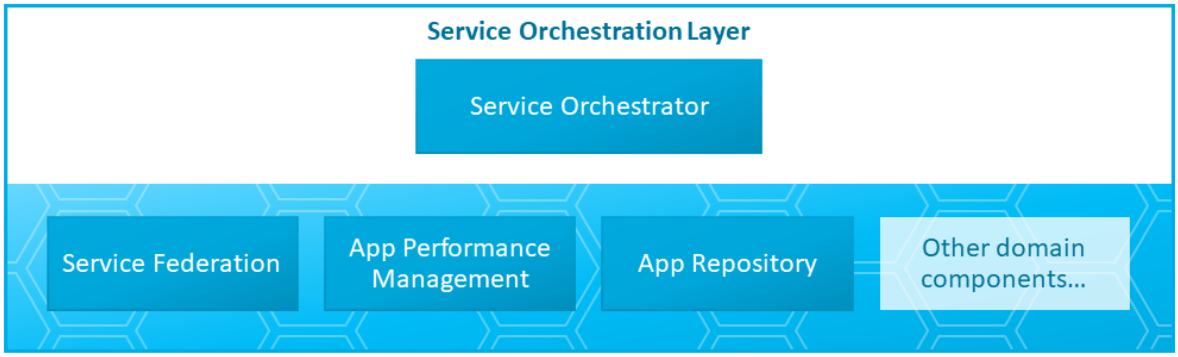


Figure 3.8.1: Components in the Service Orchestration Layer. In blue background, examples of components from federation, management and security domains associated to that layer.

Service Orchestrator

Service orchestration assures efficient tasks execution, load balancing, and real-time operations. For example, it could communicate with the Multi-Cloud Orchestrator that manages the virtualized infrastructure layer offering a single unified environment for application development and monitoring. This allows applications and services to be deployed seamlessly across multiple platforms, optimizing resource allocation and reducing operational complexity. Alternatively, the service orchestrator may directly or indirectly interact with the underlying capabilities of the cloud platform or virtualization management layer to orchestrate workload execution.

The Service Orchestrator automates application and tenant deployment, and lifecycle management processes. By automating workflows (or service function chains), orchestration ensures that services communicate efficiently across the Cloud-Edge Continuum.

Application Performance Management

It monitors the performance and resource consumption of the application or service and communicates deviations from set thresholds or SLAs to the service orchestrator for this to take actions to recover a state that meets application requirements.

It provides a unified view of states, including logging, monitoring, and alerting, for effective real-time application management and validation at runtime.

Application Repository

This component tracks the applications and services that have been deployed and their configuration, the locations where the application and service components are installed, and the resources they are consuming.

Service Federation

This component interconnects the Service Orchestrator with those of other federated providers, enabling the deployment and execution of applications (service function chains) across multiple providers in a seamless way, interacting with a single provider.

In order to create and maintain a coherent federated Multi-Provider Cloud-Edge Continuum, Service Federation capabilities should be designed consistently with the other federation capabilities described in this document.

3.9. Cloud-Edge Platform Layer

The Cloud-Edge Platform Layer hosts components that manage and orchestrate runtime environments (virtual machines, containers, serverless) and platform resources (libraries, functions, services, security, databases, tools).

The platform services (PaaS) it provides support the deployment of applications and the corresponding runtime environments and platform resources.

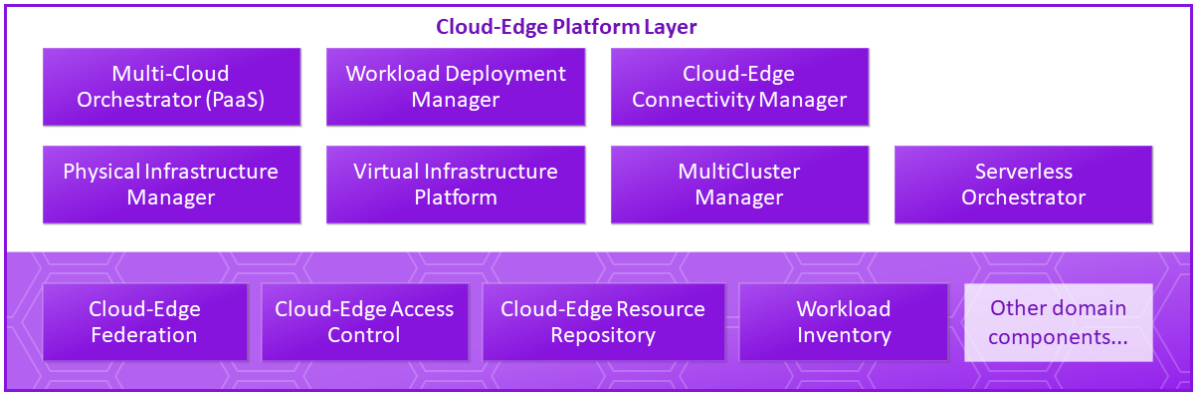


Figure 3.9.1: Components in Cloud-Edge Platform Layer. In purple background, examples of components from federation, management and security domains associated to that layer.

Multi-Cloud Orchestrator

MCO delivers a PaaS service. A PaaS provides a complete application development and deployment environment in the cloud. With PaaS, customers can build, test, deploy, manage, and update applications quickly and efficiently, without worrying about the underlying infrastructure.

It receives from the Service Orchestrator a request to deploy (or manage the lifecycle of) a certain application together with a descriptor (resource model) that defines the state the application needs for its execution (including runtime environment, services, data, application image and other attributes like area of service, performance...). The MCO processes the state and takes actions to set it up and preserve it, by updating, upgrading, or removing workloads and services, or rescaling or releasing resources.

MCO works in close relationship with other components (*Physical Infrastructure Manager* – PIM, *Virtual Infrastructure Platform Manager* – VIP, *Multi-Cluster Manager* – MCM, Serverless orchestrator) to provide the virtual runtime environment defined for the application, the specific combination of bare metal, virtual machine, containers, and serverless mechanisms, it has been developed to run on, using the technologies over which it has been tested and certified.

MCO also deploys and manages the lifecycle of essential tools and services such as middleware, development frameworks, databases, and business analytics, enabling organizations to streamline application development and drive innovation.

A PaaS, managed by the MCO, offers scalability, high availability, and reduced time-to-market, allowing developers to focus on coding and application functionality while the MCO supports with infrastructure, security, and operational aspects.

Based on certain attributes, like area of service and performance, the MCO may select the location(s) where to deploy the workload and the resources (physical and virtual) required at those location(s) to meet the desired state. This decision on application placement can also follow sustainability and privacy requirements.

The MCO deploys the workload once the necessary resources are available, using the *Workload Deployment Manager* (WDM). The MCO also updates and removes workloads, rescaling or releasing the corresponding resources.

This MCO description shows a decomposition of the functionality of a Cloud-Edge Continuum workload management solution that may be implemented in many ways, combining or excluding some of its components in order to fit specific sector needs.

Cloud-Edge Connectivity Manager

The *Cloud-Edge Connectivity Manager* (CEC) implements and modifies the service function chain, or removes it, totally or partially, following the requests from a Service Orchestrator, to guarantee the connectivity between workloads that will enable the service delivery and the connectivity from the service user to the workloads implementing the service front-end.

Connectivity is usually based on overlay and underlay components in each domain crossed by the traffic (e.g. WAN, data centers, etc.). The CEC manages the networking in the data center domain through the virtualization managers (*Virtual Infrastructure Manager* – VIM, *Container Infrastructure Service Manager* – CISM) or via specific *Network as a Service* (NaaS) interfaces. It manages the WAN connectivity using cloud networking services (via transport SDN Controllers) for the connection of different computing nodes.

In addition, the CEC manages the complexity deriving from the need to ensure consistency between overlay and underlay networking solutions (for example adapting the networking between the data center fabric and the WAN connectivity).

Physical Infrastructure Manager

The PIM monitors and manages a pool of physical resources (CPUs, storage, networking), and selects and prepares them (with the corresponding OS and necessary software) to allocate these resources to a virtual machine or container cluster.

The PIM provides multiple physical infrastructure management functions, including physical resource provisioning and lifecycle management, physical resource inventory management, or physical resource performance management.

Multi-Cluster Manager

The MCM creates and configures container clusters both over bare metal and over virtual machines upon request from the MCO, offering a single interface to manage infrastructure from multiple providers and with multiple K8s distributions.

The MCM provides open connectors/APIs to interact with the resources and k8s distributions offered by different providers (private and public) for cluster creation, configuration, and monitoring, and keeps track of their evolution.

The MCM may create a K8s cluster on bare metal (cluster nodes are servers) or on the virtualization stack (cluster nodes are VMs), interacting with PIM or VIP respectively.

Virtual Infrastructure Platform Manager

The *Virtual Infrastructure Platform Manager* (VIP) creates virtual machine clusters across several locations using the resources allocated by the PIM.

The VIP is required when the service component to be deployed is a virtualized application or a containerized application, which runs over container clusters that make use of VMs (virtual machines).

This component works on infrastructure and technology from different providers, enabling the Cloud-Edge Continuum to run on a diverse set of different virtualization solutions (VIMs, CISMs, or any other future virtualization technology).

Workload Deployment Manager

The WDM deploys software package(s) on top of an existing cluster, following the request of the MCO. It exposes a single interface to deploy software packages (i.e., via a helm chart or resource model declaration) on any K8s cluster (or alike) based on any distribution.

The WDM provides the connectors/APIs to interact with existing clusters in different locations and technologies (K8s distributions) for application deployment and lifecycle management.

This component can also deploy software packages directly on virtual machines (IaaS).

Cloud-Edge Federation

This component interconnects the MCO with the ones of other federated providers, enabling the customer to use cloud-edge computing services (IaaS, CaaS, PaaS, Serverless, NaaS, ...) across multiple providers seamlessly, interacting with a single provider.

This platform federation provides seamless integration and collaboration between multiple cloud platform providers, enabling interoperability, resource sharing, and unified lifecycle management. Shared resources may exist on all layers of cloud architecture.

By adopting standardized protocols and interfaces, platform federation facilitates enhanced scalability, efficiency, and innovation across different cloud environments while maintaining autonomy and security for each participating entity.

In order to create and maintain a coherent federated Multi-Provider Cloud-Edge Continuum, cloud-edge federation capabilities should be designed consistently with the other federation capabilities described in this document.

Cloud-Edge Access Control

This component implements a key aspect in terms of security in the management of cloud-edge infrastructure, a role-based access control that ensures proper access rights and security across the infrastructure.

Cloud-Edge Resource Repository

This component keeps a record of the resources available in each of the edge locations, the virtualization platforms available, and the configuration. The information in this repository helps the MCO to select the right location(s) to deploy workloads.

Workload Inventory

This component keeps record of the workloads that have been deployed and their configuration, as well as information about the location and cluster where they have been deployed and the resources they are consuming.

Serverless Orchestrator

The Serverless Orchestrator provides serverless capabilities, also known as *Function as a Service* (FaaS), which is a cloud computing model that allows developers to build and deploy applications in the form of individual functions, that are executed in response to specific events or triggers. This model eliminates the need to manage server infrastructure, enabling developers to focus solely on writing code. Each function runs in a stateless container, automatically scaling with demand and only consuming resources when invoked, leading to cost savings and efficient resource utilization.

3.10. Virtualization Management Layer

The Virtualization Layer in cloud infrastructures acts as a crucial abstraction layer that enables the efficient utilization of physical resources by creating multiple virtual instances of servers, storage, and network resources. This layer leverages hypervisors or related technologies to decouple hardware from the operating system, allowing for the dynamic allocation and scaling of resources based on demand. By providing a flexible and scalable environment, the virtualization layer enhances resource optimization, simplifies management, and supports the seamless deployment of various cloud services and applications. This foundational component is essential for delivering IaaS and other cloud service models, ensuring agility, cost-effectiveness, and high availability.

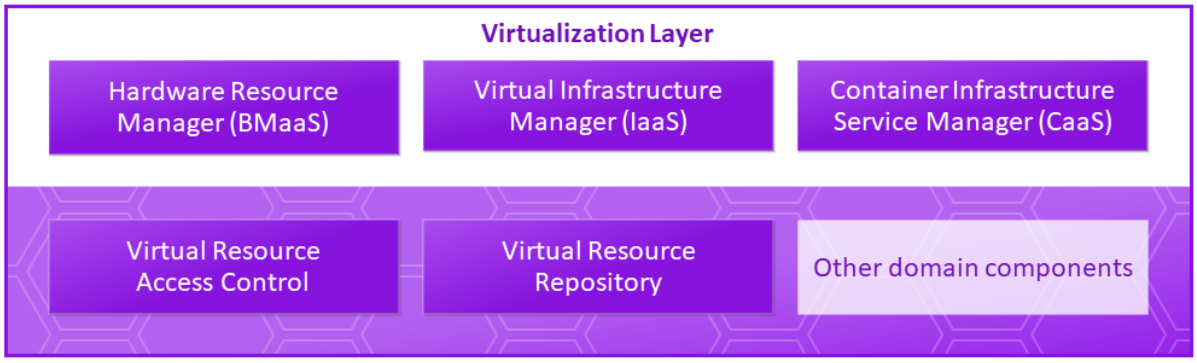


Figure 3.10.1: Components in the Virtualization Layer. In purple background, examples of components from management and security domains associated to that layer.

Hardware Resource Manager

The Hardware Resource Manager component delivers a *Bare Metal as a Service* (BMaaS) service. BMaaS is an abstraction that provides physical, non-virtualized hardware resources directly to users, offering dedicated servers, storage, and networking components without any virtualization layer.

This service allows users to harness the full power of the hardware for applications, resulting in higher performance, predictable latency, and complete control over the environment. BMaaS is particularly beneficial for workloads that require intensive computation, low-latency networking, or compliance with specific hardware configurations.

Virtual Infrastructure Manager

The VIM component provides IaaS service. IaaS is a cloud computing model that provides virtualized computing resources. IaaS delivers essential services such as virtual machines, storage, and networks. Users can provision, scale, and manage the resources dynamically according to their needs, while the cloud provider takes care of maintaining the underlying hardware, networking, and security. This model offers high flexibility, enabling organizations to quickly deploy and run applications and services, test new solutions, and handle varying workloads with ease, ultimately driving innovation and operational efficiency.

Container Infrastructure Service Manager

The CISM component provides a CaaS service. CaaS is a cloud service model that provides a platform allowing users to manage and deploy containerized applications and workloads. By leveraging container orchestration tools such as Kubernetes, CaaS facilitates the automation of container deployment, scaling, and operations, ensuring high availability and performance. This model abstracts the underlying infrastructure complexities, enabling developers and IT teams to focus on application and service development and deployment without worrying about the maintenance of the physical or virtual infrastructure.

Virtual Resource Access Control

As in the cloud-edge platform layer, this access control component implements virtual infrastructure management security, a role-based access control that ensures proper access rights and security for virtual resource management.

Virtual Resource Repository

This component keeps records of cloud-edge sites and the configuration and availability of virtual resources in each one of them (for instance, number of k8s clusters available per site, CPU/memory available per k8s cluster, number of virtual CPUs that are available to setup new k8s clusters, ...) in order to help make decisions on workload placement.

3.11. Physical Cloud-Edge Resource Layer

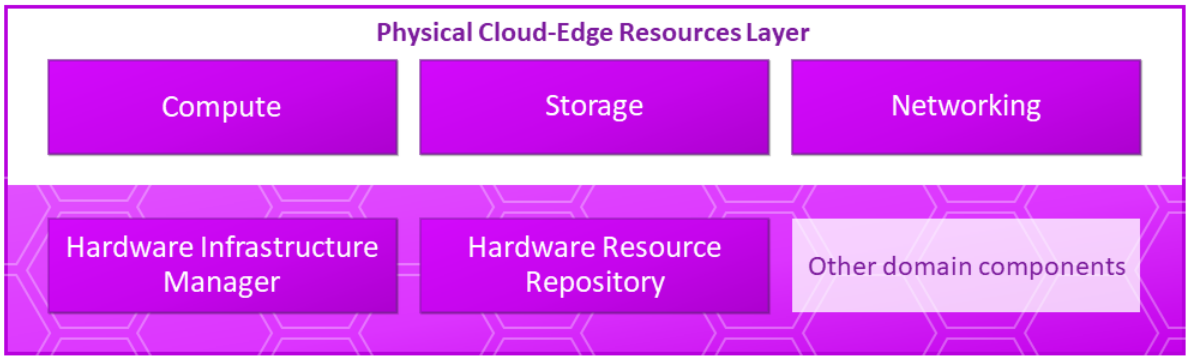


Figure 3.11.1: Components in the Physical Cloud-Edge Resource Layer. In pink background, examples of components from management domain associated to that layer.

Compute

Compute resources are fundamental to cloud infrastructure, delivering the computational power required for running applications and services. They facilitate scalable and efficient environments that dynamically adjust to varying workloads, thus enhancing resource utilization and performance while minimizing costs.

Storage

Storage is essential in cloud infrastructure, providing data persistence, management, and accessibility. It includes block storage for databases, object storage for unstructured data, and file storage for shared access applications. Advanced technologies like SSDs and distributed file systems ensure scalability, reliability, and performance.

Networking

Hardware networking resources in a cloud-edge location include routers, switches, load balancers, and firewalls. These components form the backbone of data center connectivity and inter-server communication. *Network Interface Cards* (NIC) in servers enable high-throughput connections to the virtual network. WAN gateways and edge routers extend connectivity to external networks, supporting hybrid cloud and remote access scenarios. All hardware is managed centrally through SDN controllers and scaled dynamically to support edge-cloud service demands.

Hardware Infrastructure Manager

A *Hardware Infrastructure Manager* (also known as *Data Center Infrastructure Management* system, DCIM[3]) is a management component designed to monitor, measure, and manage the IT equipment and infrastructure within a cloud-edge data center. It encompasses the following key aspects:

- Monitoring and Management: It provides real-time monitoring of data center operations, including power usage, cooling efficiency, and physical security. This helps in optimizing the performance and efficiency of the data center.
- Documentation and Planning: It maintains detailed documentation of the data center's physical and virtual assets. This includes layout planning, capacity management, and future expansion plans.
- Risk Management: By continuously monitoring environmental conditions and equipment status, it helps in identifying potential risks and mitigating them before they lead to failures.
- Integration with IT Systems: It integrates with other IT management systems to provide a holistic view of the data center's operations, facilitating better decision-making and resource allocation.
- Sustainability and Compliance: It supports sustainability goals by optimizing energy usage and ensuring compliance with industry standards and regulations.

Hardware Resource Repository

This component keeps records of cloud-edge locations and the configuration and availability of physical hardware resources in each one of them (for instance, number of servers per location, type

[3] Relevant standards: 1) ISO/IEC 18598, which focuses on automated infrastructure management (AIM) and includes aspects relevant to DCIM, 2) ITU-T L.1305 outlines technical specifications for DCIM systems, including principles, management objects, and operational function requirements.

of servers, type of NIC cards available per location, cost of resources, energy consumption of resources, ...) in order to help make decisions on workload placement and resource lifecycle management.

4. Federation

In cloud-edge infrastructures, federation refers to the collaborative integration and seamless interoperability among cloud environments belonging to distinct domains, organizations, or providers. Federation is based on standard APIs and data models that enable consistent communication and integration across different platforms. Federation preserves the security, autonomy, and governance of each participating entity. At the same time, it enables interoperability and resource sharing through standardized protocols and interfaces. Examples of these interfaces can be found in the next section, “Functional Interfaces”. These protocols facilitate the aggregation of services, computational resources, and data across distributed systems. This approach can enhance both scalability and operational efficiency.

Federation may ensure quality of service guarantees for end users across applications, such as specific bandwidth and latency requirements, and underpins advanced use cases including cross-cloud data analytics, collaborative software applications, and distributed artificial intelligence workflows. Through federation, cloud service providers can offer their customers access to services and resources from partner providers, creating a seamless multi-provider experience within a unified cloud continuum. Similarly, customers can compose and use services from multiple cloud service providers in a unified fashion.

Customers can leverage resources from multiple service domains through two primary approaches:

- Service Integration: Consuming external provider services and integrating them with existing distributed workloads.
- Workload Migration: Migrating workloads to provider platforms to achieve optimized execution and management while maintaining functional and quality requirements.

Crucially, these resource allocations maintain flexibility throughout the entire workload lifecycle. Both the services consumed by a workload and the workload’s execution location can be dynamically reassigned or relocated in response to evolving business requirements, performance demands, cost optimization objectives, or sustainability imperatives. Depending on the specific service characteristics and operational scenario, this reallocation may be customer-initiated or executed directly through inter-provider coordination.

Federation Principles

The core principles described here are designed to be universally applicable across diverse deployment scenarios: whether working with many providers, a few providers, or even a single provider with multiple locations. These principles should seamlessly support all connectivity patterns, including data center to data center, data center to edge, and edge-to-edge deployments.

A fundamental principle is to avoid technological lock-in, meaning that any federation should rely on open, well-specified, and technology-agnostic intermediate abstractions (e.g., interfaces and data

models). The technologies used by the providers should be mediated through those abstractions. The various abstractions expose capabilities and policies in a technology-neutral way while allowing concrete providers to implement them within their preferred stacks. The Reference Architecture defines the responsibilities and interaction patterns that implementations must support. Different federation frameworks (for example, those developed in specific IPCEI-CIS projects) can coexist as long as they adhere to those responsibilities. This ensures flexibility and interoperability across different vendor solutions.

Similarly, a lock-in to a single federation participant is generally undesirable. The architecture encourages flexible federation models where federation roles and functions (for example, discovery, brokerage, orchestration, trust, and policy evaluation) may be operated by different participants, including providers, customers, neutral intermediaries, or consortia. The architecture should not rely on any single privileged operator or specific technology. It strongly stimulates multiple, independently operated implementations to coexist and to be plug-replaceable, so that participants can change providers or implementations without disrupting the overall federation.

The architecture supports collaboration between many independent participants at the same time, for example, multiple cloud and edge providers, federation middleware operators, marketplaces, trust/credential authorities, and customer-operated components. Several federations may coexist and interconnect. No single participant is assumed or required to act as a global or central federation operator.

To work effectively across various cloud-edge providers, network interconnection and transport providers, service marketplaces, and customer organizations, a common understanding of the objects of interest is essential for consistent discovery, comparison, and composition across providers. As such, it is strongly recommended to describe services, their compositions, policies, and SLAs in a machine-readable, shared information model (e.g., an ontology-based model using semantic web standards). This standardized approach should be applied to all layers within the cloud-edge federation.

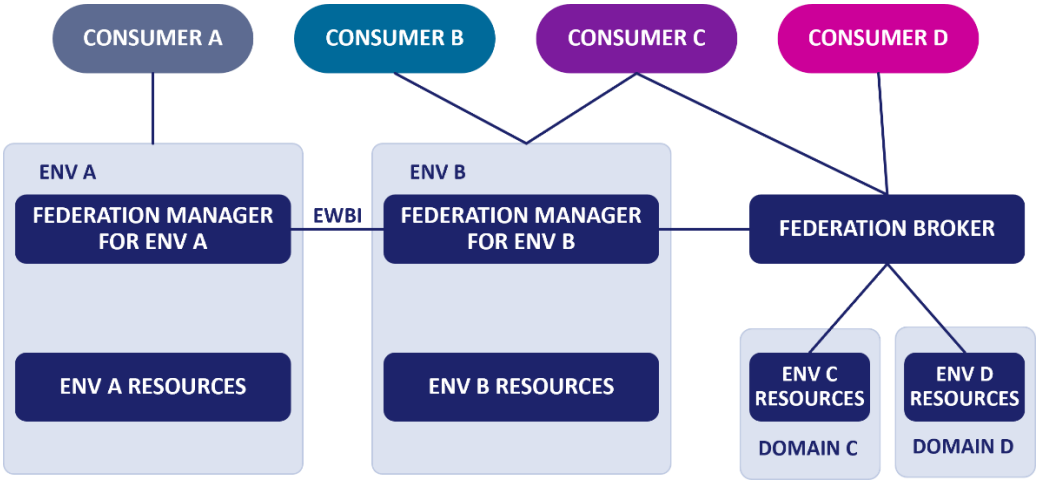
Trust, governance, and compliance are foundational elements in federations. Federations must respect regulatory, contractual, and sector-specific governance requirements. This entails implementing cross-provider identity and access management, using machine-readable, and verifiable representations of organizational attributes (such as provider’s headquarters location, certifications (e.g., ISO), and regulatory compliance) and establishing mechanisms for expressing and enforcing policies across domains. Trust models may also incorporate external trust frameworks and credential systems to deliver cryptographically verifiable evidence of identity, attributes, and compliance.

Finally, in line with the regulations on switching Data Processing Services in the Data Act (cf. Chapter 2), federation components and mechanisms should not introduce additional switching barriers beyond those of the underlying services and should, where possible, help to remove existing obstacles rather than create new forms of lock-in.

Core Components

The figure shows a federated cloud architecture in which multiple providers collaborate while maintaining independent environments. In the figure, a provider is the entity that owns and

manages an environment, i.e., its own operating environment with local infrastructure, services, and policies. Each provider can offer both its own services and those from federated partners.



Within each environment, there is a Federation Manager, the standard module that enables communication with other environments. Externally, it exposes the EWBI (*East-West-Bound Interface*), an interface to ensure interoperability between environments. Internally, however, it communicates natively with the systems of its own environment. A potential implementation of the EWBI can follow GSMA specifications.

Some environments (such as C and D) can offer resources through a Federation Broker (a module similar to the Federation Manager, which exposes the same interfaces), which serves as a unified access point to multiple environments, while others (such as A and B) communicate directly between Federation Managers.

Consumers are organizations or users that consume the services offered by providers. Consumers can use both provider and partner services through interfaces provided by the provider, which hide complexity.
Consumers always access everything through a single access point provided by the provider. Inside this access point, a consumer only sees what they need and only the resources that belong to them. If a partner offers additional services, consumers do not deal with any federation logic. Consumers never interact directly with the federation manager. The provider handles everything behind the scenes:

- Sets up the federation with the partner.
- Maps and standardizes the partner's services into its own model.
- Shows the consumer only the services they are allowed to use.

For consumers, everything appears to come from one place. In reality, the provider integrates and filters resources from multiple partners to provide a single, simple access point.

The two main components of federation are the Federation Manager and the Federation Broker. This document provides only an abstract description of these components. The next version will dive deeper into real-world implementations and practical use cases.

As a general principle, Federation Manager and Federation Broker shall be compatible implementations, allowing federated partners to decide which approach to be used based on a specific use case.

Federation Manager

The Federation Manager allows customers to use resources and services not only with their current provider, but also with other providers that are federated with it and handles the technical and contractual onboarding as well as runtime aspects such as accounting.

It enables federated partners to:

1. Establish and negotiate the federation.
2. Negotiate access to federated resources and services.
3. Allow access to deployed resources and services.

Interconnection between Federation Managers allows customers to allocate, scale, and release capabilities and resources (services, applications, bare metal, virtual machines, containers, serverless) across these providers using compatible APIs. The layer at which customers use federated services depends on the application type and its requirements.

Federation Broker

The Federation Broker enables customers and providers to discover services and platform offerings from other providers through Service Marketplace Approach and On-Demand Bundle Approach.

Service Marketplace Approach: A Federation Broker may maintain a comprehensive catalog of all offerings that providers have registered. This approach is commonly known as a service marketplace.

On-Demand Bundle Approach: Customers request specific capabilities and service bundles, and providers respond with tailored offers for the requested configuration. This approach offers two key advantages: First, customers receive a guarantee that bundled services technically work together seamlessly, with transparent pricing that includes all costs, such as inbound and outbound traffic between services. Second, providers gain greater flexibility in contract structuring. For example, bundle pricing may offer discounts compared to individual service pricing, potentially providing a competitive advantage.

Advanced Integration

Multiple Federation Brokers can be interconnected in various ways, and when Federation Managers and Federation Brokers are integrated, onboarding may be initiated directly from the Federation Broker. However, these advanced topics are beyond the scope of this document.

5. Functional Interfaces

The functional interfaces used in actual implementations are presented in the Figure 5.1. These are high-level abstractions and represent essential interfaces in all the possible architecture implementations. Lower-level interfaces, despite being present, are not reported in this view.

The interfaces in this view could/should be labeled as interfaces increasing European independence to different degrees. The potential identification of the interfaces that will enforce European independence could be a matter for further studies and improvements.

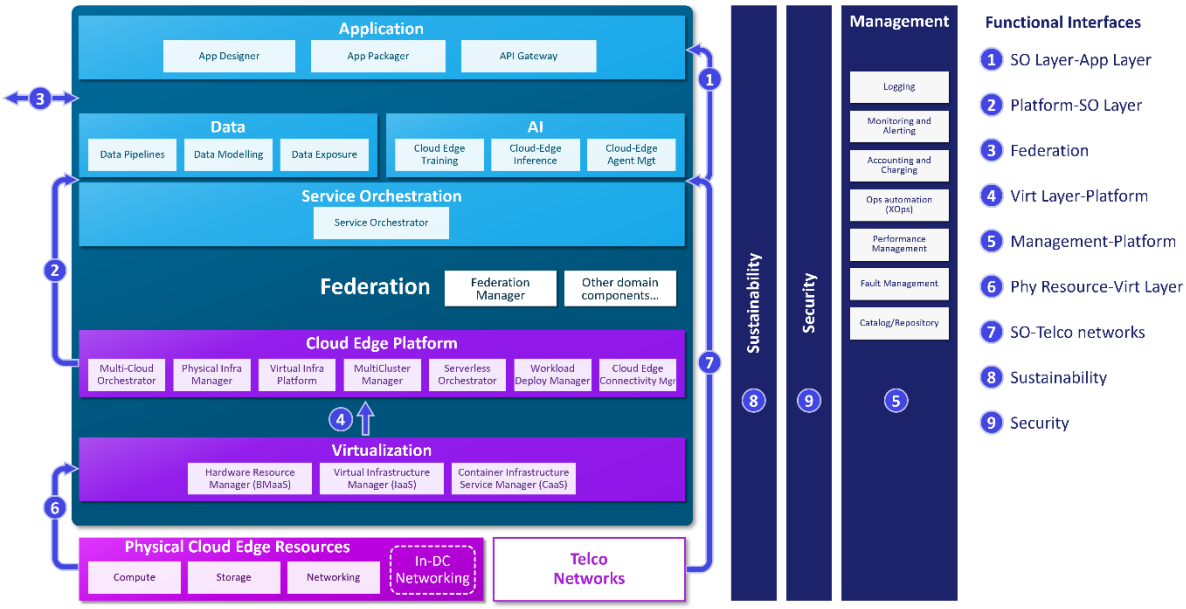


Figure 5.1: Functional Interfaces.

Based on the comprehensive analysis of partner responses to the functional interface definitions for the Reference Architecture, here is the synthesis of the nine interfaces that were selected among all possible interfaces in the Reference Architecture. For each of the interfaces, the key technologies on which they are based in current implementations are shown, with a synthesis to describe which are used in the projects, and a preliminary assessment of the strong points, where an alignment with state-of-the-art implementations is reached, and areas where further enhancements could be important in the near future.

1. Service Orchestration to Application Layer

Key Technologies:

- Kubernetes API (https://kubernetes.io/docs/reference/)
- Kubernetes Resource Model as used in NeoNephos Platform Mesh (https://platform-mesh.io/)
- REST APIs (https://restfulapi.net/)
- JSON(-LD) (https://www.json.org/json-en.html)
- Model Context Protocol (MCP, https://modelcontextprotocol.io/)

Synthesis: This interface enables the orchestration layer to deploy and manage application components across the Cloud-Edge Continuum. It ensures lifecycle control, connectivity, and policy enforcement for applications, allowing seamless integration of workloads with orchestration workflows. The industry converges on the Kubernetes API as the dominant technology. Solutions promoted or developed as part of the IPCEI-CIS include NeoNephos Platform Mesh for service instance management, the CAMARA EAM API (https://github.com/camaraproject) for application lifecycle management, and TMForum Business API Ecosystem (https://github.com/FIWARE-

TMForum/) for monetization. The emerging **Model Context Protocol (MCP)** can be used for AI tool integrations. The interface heavily relies on **RESTful APIs** with JSON(-LD) payloads and is consistently identified as a core interface that cannot be easily substituted.

The emphasis on this interface is primarily on:

- **Kubernetes API** dominance aligns with ETSI NFV Release 5+ evolution toward cloud-native orchestration
- **CAMARA APIs** for application lifecycle management reflect the Linux Foundation's major telco API standardization initiative
- **Model Context Protocol (MCP)** shows forward-thinking AI integration, consistent with emerging cloud-native requirements.

Missing considerations: Intent-based orchestration patterns like those in Nephio could strengthen this interface.

2. Platform to Service Orchestration

Key Technologies:

- Kubernetes API (https://kubernetes.io/docs/reference/)
- OpenStack API (https://docs.openstack.org/api-ref/)
- OpenNebula API (https://docs.opennebula.io/7.0/product/integration_references/system_interfaces/)
- GitOps (https://opengitops.dev/)
- OpenTelemetry (https://opentelemetry.io/docs/)
- NeoNephos CobaltCore (https://cobaltcore-dev.github.io/docs/)
- NeoNephos IronCore (https://ironcore.dev)
- NeoNephos Gardener (https://gardener.cloud)
- NeoNephos Platform Mesh (https://platform-mesh.io)
- Sylva (https://sylvaproject.org)

Synthesis: This interface connects orchestration logic with platform services, translating high-level deployment intents into executable actions on virtualized and containerized environments. It provides observability and automation for resource allocation and service chaining. This interface represents the most mature and standardized layer, with general adoption of **Kubernetes API** as the core technology. The NeoNephos foundation (https://neonephos.org) provides comprehensive coverage through Platform Mesh, Gardener, CobaltCore (OpenStack), and IronCore APIs. For deployment and operation, **GitOps-based service provisioning** and **OpenTelemetry for observability** can be used. The interface supports both traditional VM management through OpenStack/OpenNebula/CobaltCore and cloud-native workloads through Kubernetes/Gardener/IronCore/Sylva, making it the foundational layer for hybrid-cloud operations.

This interface represents the industry's best practice with:

- **Kubernetes API adoption** matches ETSI NFV's shift toward declarative, cloud-native MANO.
- **GitOps-based service provisioning** aligns with Nephio's configuration-as-data approach.
- **OpenTelemetry for observability** follows CNCF graduated project standards.

- Interfaces heavily rely on **RESTful APIs** with JSON payloads.

State of the art validation: ETSI NFV specifically calls for simplified, declarative orchestration interfaces exactly as the partners propose.

3. Federation (Multi-Cloud-Edge Coordination)

Key Technologies:

- Kubernetes API (https://kubernetes.io/docs/reference/)
- GSMA EWBI API (https://www.gsma.com/solutions-and-impact/gsma-open-gateway/)
- NeoNephos Platform Mesh (https://platform-mesh.io/)
- NeoNephos Katalis (https://github.com/NeoNephos-Katalis)
- Federated Learning APIs

Synthesis: This interface supports interoperability between multiple providers, enabling resource sharing and workload migration across federated clouds. It ensures consistent APIs for cross-domain orchestration and facilitates collaborative use cases like distributed AI and data exchange. On the federation interface enabling multi-operator and multi-cloud coordination, the **GSMA EWBI API** provides a solution for secure resource sharing between operators. Other options include the NeoNephos Platform Mesh and **Kubernetes federation via Liqo** (https://liqo.io/docs/). Open-source projects like NeoNephos Katalis are potential references for federation in multi-cloud-edge environments.

Strong points:

- The definition of a federation interface helps greatly towards the implementation of a common European Multi-Provider Cloud-Edge Continuum.
- **GSMA EWBI API** for operator federation is one of the current industry standards and it is well present in IPCEI implementations.
- **Kubernetes federation via Liqo** represents a further multi-cloud orchestration option.
- Possible synergies and interactions with open-source projects on the topic.

Areas needing attention:

- **Federated learning APIs** are still nascent; industry standards are not yet mature
- Frameworks for European sovereignty to be further elaborated, considering all the potential technologies which will strengthen sovereignty through the federation environment

4. Virtualization to Platform

Key Technologies:

- Kubernetes API (https://kubernetes.io/docs/reference/)
- NeoNephos Gardener (https://gardener.cloud)
- NeoNephos CobaltCore (https://cobaltcore-dev.github.io/docs/)
- NeoNephos IronCore (https://ironcore.dev/)
- OpenStack API (https://docs.openstack.org/)
- OpenNebula API (https://docs.opennebula.io/7.0/product/integration_references/system_interfaces/)
- TOSCA (https://www.oasis-open.org/committees/tosca/)

- Cluster API (CAPI, https://cluster-api.sigs.k8s.io/)
- Sylva (https://sylvaproject.org/)

Synthesis: This interface bridges physical infrastructure and platform services by exposing virtualized resources through standardized APIs. It enables dynamic provisioning of VMs, containers, and clusters, ensuring portability and scalability across heterogeneous environments. It bridges physical infrastructure with platform services through Kubernetes Cluster API (CAPI), NeoNephos Gardener, NeoNephos CobaltCore, NeoNephos IronCore, OpenStack, and OpenNebula. A Sylva Cluster API is another alternative for Kubernetes cluster creation and management. TOSCA provides an Infrastructure-as-Code language for topology definition, providing expressiveness for translation into target orchestration languages.

This interface is considered to match adequately the current virtualization evolution:

- **Kubernetes Cluster API (CAPI)** is the de facto standard for cloud-native cluster lifecycle management
- **TOSCA 2.0 for IaC** remains relevant for telco workload description, though cloud-native extensions are needed
- **OpenStack, CobaltCore, and OpenNebula** integration interfaces reflect hybrid cloud realities in telco environments
- **IronCore** can be used for modern cloud-native environments
- Interfaces heavily rely on **RESTful APIs** with JSON payloads

Enhancement opportunity: From a telco perspective, integration with O-RAN cloud-native architectures could strengthen edge virtualization aspects.

5. Management to Platform

Key Technologies:

- Kubernetes API (https://kubernetes.io/docs/reference/)
- OpenTelemetry (https://opentelemetry.io/docs/)
- Agent Composition Platform
- AI Model Health APIs

Synthesis: This interface provides monitoring, logging, and SLA enforcement for platform services. It integrates observability frameworks and automation tools to maintain performance, reliability, and compliance across distributed cloud-edge deployments. It focuses on comprehensive platform monitoring and management through **OpenTelemetry** for observability and logging. A partner provides App Deployment Monitoring APIs with Kafka Bus integration and OTEL collectors, emphasizing performance SLA compliance. Multiple partners contribute **Agent Composition Platform APIs** for low-code/no-code application development and monitoring. The interface includes specialized **AI Model Health APIs** for monitoring inference latency, resource utilization, and service outages in AI workloads. Sylva is mentioned by a partner too.

This interface aligns well with industry evolution:

- **OpenTelemetry-centric** observability follows CNCF best practices

- **AI Model Health APIs could** anticipate the AI-driven telco cloud future
- **Agent Composition Platform APIs** reflect the trend toward low-code/no-code telco service creation

State-of-the-art match: The multi-partner focus on AI monitoring aligns with emerging requirements in cloud-native operations.

6. Physical Resources to Virtualization

Key Technologies:

- Redfish API (https://www.dmtf.org/standards/redfish)
- OpenStack API (https://docs.openstack.org/)
- NeoNephos IronCore Bare Metal API (https://ironcore.dev/baremetal/)
- libvirt (https://libvirt.org/)
- Ceph (https://docs.ceph.com/en/latest/)
- WASM (https://webassembly.org/)

Synthesis: This interface abstracts hardware resources into virtualized entities using standard protocols. It manages compute, storage, and networking allocation, ensuring efficient utilization and enabling advanced features like bare-metal provisioning and hardware-specific optimizations. It handles the lowest-level infrastructure abstraction using industry standards where possible. **Redfish API** is a common API for hardware management, with proprietary drivers for GPUs and specialized networking hardware. The interface utilizes **libvirt and Ceph** for virtualization and storage management. A partner adds **WASM resource management capabilities**, directly interfacing with physical resources for containerized workload optimization, considering also Cubbit. NeoNephos Bare Metal API leverages the Redfish protocol for full automation based on Kubernetes mechanisms.

Industry-standard approach with:

- **Redfish API** for hardware abstraction follows DMTF standards
- **libvirt and Ceph** represent mature open-source virtualization stacks
- **WASM resource management** shows innovation in edge computing optimization

Potential steps forward: From a telco perspective, **O-RAN specifications** for RAN-specific hardware abstraction could be a valuable addition.

7. Service Orchestration-Telco Networks

Key Technologies:

- 3GPP 5GS (https://www.3gpp.org/specifications)
- 3GPP LTE (https://www.3gpp.org/specifications)
- GSMA OP (https://www.gsma.com/solutions-and-impact/technologies/networks/operator-platform-hp/)
- REST APIs (https://restfulapi.net/)
- JSON (https://www.json.org/json-en.html)
- IPX

Synthesis: This interface connects the Service Orchestrator with the specific telco network capabilities with the purpose of collecting user equipment location information or the serving gateway, while also performing actions from the SO. The concept of edge is closely related to the proximity of the edge server to the subscriber, furthermore when the terminal connected to the edge is moving across the telco network, the optimal edge server may change and consequently the telco gateway that supports the telco service. GSMA standards support this interface called Operator Platform's SBI-NR based on 3GPP specifications and elements as NEF/SCEF or NWDAF. Partners are working on a network integration model with the goal of end-to-end SLA guarantee in all the edge scenarios. The key technologies are based on 3GPP specifications (architecture, elements involved, services, *Multi-Access Edge Computing* (MEC), procedures, policy and charging, specialized mediation elements, APIs) and GSMA architecture and APIs, also ETSI ISG MEC is working on edge.

This interface is being standardized and developed in:

- **ETSI:** Edge concept was introduced by ETSI and then developed by 3GPP in 2014, now this concept is renamed to MEC, https://www.etsi.org/technologies/multi-access-edge-computing). The most important ETSI specifications are ETSI GS MEC 002 MEC requirements and ETSI GR MEC 031 guidance on MEC integration with 3GPP 5G systems.
- **3GPP:** 4G and 5G architecture are entirely described in the TS documents with all the procedures for establishing communication and its lifecycle management. MEC is standardized in TS 23.558. The architecture where MEC is included is described in TS 23.501 and procedures in TS 23.502, policy and charging control in TS 23.503. NEF is the 5G function which can expose the network capabilities to 3rd parties, NBI specified in TS 29.522. NWDAF is also important to collect statistical data and network information specified in TS 29.520.
- **GSMA PRDs:** develop the concept of edge and introduce the Service Orchestrator called Operator Platform (https://www.gsma.com/solutions-and-impact/technologies/networks/gsma_resources/gsma-operator-platform-group-july-2025-specifications/). There are two distinct groups: OPG for OP and EPG for edge. The OP architecture and SBI-NR are defined in GSMA: OPG.02 defines the OP architecture where SBI-NR is an essential part, OPG.11 is about edge services requirements and OPG.03 describes SBI-NR APIs.

Strong aspects: This set of standards and best practices becomes the basis for integrating edge into the telco network. **Missing considerations:** Network integration depends on the maturity of the technologies. There are important open issues about mobility, UPF/SMF reselection and collecting end-to-end delay information or transport mechanisms to provide the optimal path. There are distinct groups working on these tasks.

8. Security

This interface enforces security and compliance policies across all layers, integrating identity management, encryption, and threat detection into operational workflows. It ensures consistent application of security controls during orchestration and runtime. The operational execution of security and compliance, as described in the CSF, will be further detailed following the progress in the 8ra project.

9. Sustainability (Environmental Optimization)

Key Technologies:

- **OpenLCA** (https://www.openlca.org/)
- **Carbon Footprint Assessment**
- **Cloud-Edge Platforms**
- **Kubernetes APIs**
- **Telemetry APIs**
- **Orchestration tools**

Synthesis: This interface introduces mechanisms for energy-aware orchestration and carbon footprint monitoring. It enables workload placement strategies that optimize resource consumption for sustainable cloud-edge operations. It addresses environmental impact through **energy consumption monitoring** and **carbon footprint calculation**. For example, outcomes are foreseen to be generated with OpenLCA, a tool for lifecycle assessment and resource consumption tracking (water, energy, CO2). Another option is doing ex-ante carbon footprint evaluation and runtime data acquisition for energy-saving policies. The functional interface will also support operational optimization of (federated and secure) digital services infrastructure by providing incentives, doing optimization, and providing reporting.

The following key technologies are currently being involved, among others:

- **Platform APIs** to minimize redundant data creation and transfer, promoting optimized application deployment for sustainability benefits. Standards are still being defined, while appropriate metering and monitoring solutions are still being explored. Among others, Sylva, EFN, OpenNebula, and Camara are being considered.
- **Telemetry APIs** enable standardized energy measurement collection, aggregation, and reporting. Examples include many open-source technologies such as Prometheus, Kepler, Alumet, but also SNMP, Redfish, and IPMI, and others.
- **Kubernetes APIs** allow applications and/or platform controllers to label resources and schedule workloads across clusters according to sustainability parameters. Examples of relevant technologies here are *Kubernetes Control Plane* (KCP), IronCore, Gardener, and *Kubernetes Resource Model* (KRM), among others.
- **Orchestration APIs** enable sustainability management by linking services and resources, and automating workflows for administrative processes and supervising service-impacting changes. Examples of open-source technologies that provide interfaces in this domain are OpenNebula, Surf Workflow Orchestrator, Netbox, and others.

6. Reference Architecture Roles

The Reference Architecture is designed to support unified pan-European cloud-edge infrastructure and services capable of executing diverse use cases by using different, yet compatible and interoperable software stacks. This flexibility is essential for accommodating the varied requirements of different sectors and applications across the IPCEI-CIS project, ensuring that the infrastructure can adapt to a wide range of operational demands while maintaining a consistent and

integrated framework. The openness and standardization of interfaces facilitate the necessary portability and scale in a multi-provider environment.

This model is **technology-agnostic** and allows the deployment, when required, of customized software stacks for specific use cases.

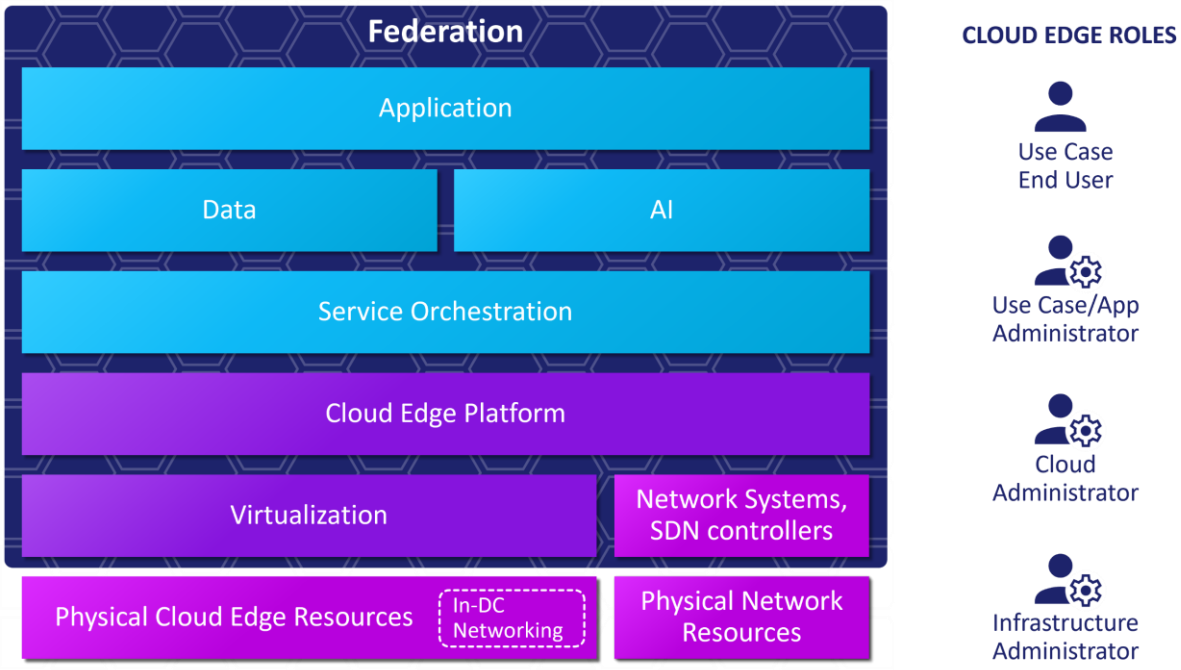


Figure 6.1: Different roles in the ICRA.

The different roles in this architecture model are described in Figure 6.1. The infrastructure resources are managed by **Infrastructure Administrators**, ensuring their availability through well-defined APIs to manage bare metal resources through the Physical Infrastructure Manager (in a multi-provider scenario), or directly through specific hardware resource managers.

The **Cloud-Edge Administrator** is responsible for the management of the container, serverless and virtual infrastructure resources, interacting with the corresponding orchestrators (VIP, MCM, serverless orchestrator) or directly with the specific managers (VIM, CISM, etc.) in the case of single-provider scenarios.

The **Use Case Administrator** works with the Application Layer components to design and package the use case application to prepare it for its deployment, defining the state it must reach for a proper execution (resource model). The Use Case Administrators hand this resource model over to the Service Orchestrator for its deployment and lifecycle management. By using the cloud-edge platform layer components, it deploys the cloud-edge runtime environment with the necessary software stack to reach the desired state. This runtime environment is tailored to meet the specific infrastructure and functionality needs of the use case, providing the necessary resources for the execution of the use case applications. The automation of this deployment process ensures that the environment is rapidly configured and aligned with the predefined requirements, reducing the time and effort needed to set up the infrastructure.

Once the cloud-edge runtime environment is deployed, the Use Case Administrator manages the operation of the environment through the orchestrators, that provide the flexibility to handle workloads efficiently, whether they are running in containers or on virtual machines.

Finally, the **End Users** of the specific **Use Case** are responsible for developing and deploying the workloads within this environment, ensuring that the applications and services required for the use case are designed and packaged properly.

7. Summary and next steps

This document has described the Reference Architecture, which serves as the common framework for the IPCEI-CIS integrated project. It is used to allow the description and comparison of solutions, workstream activities, integration clusters, and pilots run in the project, and to facilitate the integration of the diverse technological components the IPCEI-CIS is generating.

This second version of the ICRA describes functional components, organized into layers to help clarify the complexity of systems needed to support the European Multi-Provider Cloud-Edge Continuum. However, it is not intended as an implementation guideline. Compatibility, interoperability, and portability among different IPCEI-CIS implementations are ensured through standard open interfaces that provide access to various services and enable service federation.

Compared to the first version, this iteration provides enhanced detail on sustainability and security, and introduces new sections on Federation and Interfaces. Future incremental versions will include additional details and use cases. A comprehensive definition of domain components within each layer and their interconnections will also be addressed in subsequent document versions. For practical purposes, this document includes only selected examples of domain components per layer, with more detailed information available through documents produced by specific IPCEI-CIS workstreams.

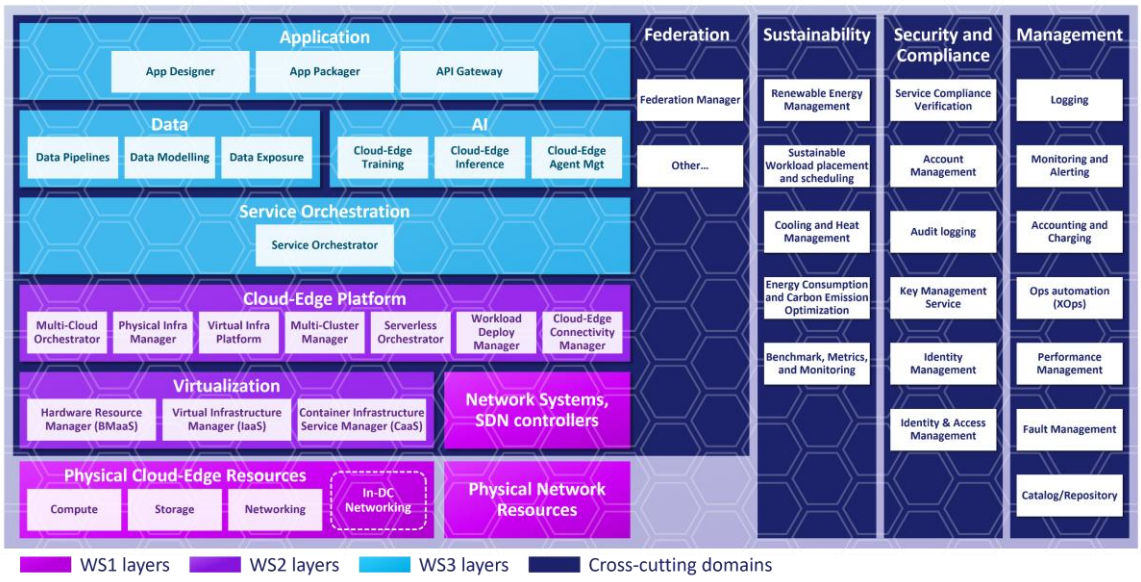


Figure 7.1: Layers, Domains and Components at the ICRA.