D2.1 1st Release of the CECC framework and CECCM

Document Summary Information

Project Identifier	HORIZON-CL4-2022-DATA-01. Project 101093129		
Project name	Agile and Cognitive Cloud-edge Continuum management		
Acronym	AC^3		
Start Date	January 1, 2023	**End Date**	December 31, 2025
Project URL	www.ac3-project.eu		
Deliverable	D2.1. 1st Release of the CECC framework and CECCM		
Work Package	WP2		
Contractual due date	M08: 31st August 2023	**Actual submission date**	M08: 31st August 2023
Type	R- Document, report	**Dissemination Level**	PU – Public
Lead Beneficiary	ION		
Responsible Author	Ayman Radwan, Souvik Sengupta (ION)		
Contributors	Adlen Ksentini, Mohamed Mekki, Sofiane MESSAOUDI (EUR); Gleibis Camejo Castillo (ARS); Dimitrios Amaxilatis, Nikolaos Tsironis (SPA); Luis Angel Garrid Platero (IQU); Ibrahim Afolabi (FIN); Amadou Ba (IBM); Vrettos Moulos (UNIPI); Dimitris Klonidis (UBI), Ali Nikoukar, Arian Firouzbakhsh (ION).		
Peer reviewer(s)	George Tsolis (CTX) & Georgios Katsikas (UBI)		

Funded by the European Union

AC^3 project has received funding from European Union's Horizon Europe research and innovation programme under Grand Agreement No 101093129.

Revision history (including peer reviewing & quality control)

Version	Issue Date	% Complete	Changes	Contributor(s)
v1.0	12/04/2023	5%	Initial Deliverable Structure (ToC)	Ayman Radwan, Souvik Sengupta (ION)
V1.1	14/06/2023	20%	Initial version of the Architecture has been introduced	Adlen Ksentini (EUR)
V1.2	19/07/2023	70%	Provided inputs for Functional architecture and functional and non-functional requirements	Souvik Sengupta (ION); Adlen Ksentini, Mohamed Mekki (EUR); Gleibis Camejo Castillo (ARS); Dimitrios Amaxilatis, Nikolaos Tsironis (SPA); Luis Angel Garrid Platero (IQU); Ibrahim Afolabi (FIN); Amadou Ba (IBM); Vrettos Moulos (UNIPI); Dimitris Klonidis (UBI)
V1.3	24/07/2023	90%	Prepared the initial full draft	Souvik Sengupta (ION), Dimitrios Amaxilatis, Nikolaos Tsironis (SPA)
V1.4	02/08/2023	95%	Received internal reviewers' feedback	George Tsolis (CTX) & Georgios Katsikas (UBI)
V1.5	23/08/2023	100%	Prepared the final draft	Souvik Sengupta, Ali Nikoukar, Arian Firouzbakhsh (ION); Adlen Ksentini, Mohamed Mekki (EUR); Gleibis Camejo Castillo (ARS); Dimitrios Amaxilatis, Nikolaos Tsironis (SPA); Luis Angel Garrid Platero (IQU); Ibrahim Afolabi (FIN); Amadou Ba (IBM); Vrettos Moulos (UNIPI); Dimitris Klonidis (UBI)

Disclaimer

Copyright message

Table of Contents

List of Figures

List of Tables

Glossary of terms and abbreviations used

Abbreviation / Term	Description
AC³	Agile and Cognitive Cloud edge Continuum management
AG	Application Gateway
API	Application Programming Interface
CECC	Cloud Edge Computing Continuum
CECCM	Cloud Edge Computing Continuum Manager
CFN	Compute First Networking
CFRA	Cloud Federation Computing Reference Architecture
CNCF	Cloud Native Computing Foundation
CRUD	Create, Read, Update, and Delete
FHS	Federation hosting service
GUI	Graphical User Interface
IDSA	International Data Spaces Association
IoT	Internet of Things
KPI	Key Performance Indicator
LCM	Life-Cycle Management
LMS	Local Management System
ML	Machine Learning
NBI	Northbound Interface
OSR	Ontology and Semantic aware Reasoner
OWL	Web Ontology Language
PaaS	Platform as a Service
PLI	Profile Language Interpreter
RDF	Resource Description Framework
SBI	Southbound Interface
SDWAN	Software-defined wide area network
SIIF	Standard for Intercloud Interoperability and Federation
SLA	Service Level Agreement
WAN	Wide Area Network

1 Executive Summary

The document is deliverable "D2.1: 1st release of the CECC framework and CECCM" of the AC^3 (Agile and Cognitive Cloud Edge Continuum Management) project funded under the Horizon Europe Research and Innovation Action programme. This deliverable describes the high-level functional architecture of the AC^3 framework, along with its key components and their interfaces; that aims to facilitate the agile and cognitive management of the Cloud Edge Computing Continuum (CECC). It is worth mentioning that this deliverable corresponds with the preliminary outcome of task T2.2 titled "Reference Architecture for CECC".

The Federation and Orchestration of Cloud, Edge, and Far Edge computing platforms hold immense significance in the modern computing ecosystem. This fusion empowers enterprises to build a robust infrastructure, leveraging each platform's strengths, resulting in improved performance, scalability, real-time responsiveness, and cost-efficiency across diverse applications. Cloud Computing ensures on-demand access to shared resources over the internet, offering scalability and flexibility. Edge Computing processes data closer to its source, reducing latency, while Far Edge Computing extends this concept for ultra-low latency scenarios.

The integration of these platforms brings several key benefits. Firstly, it reduces latency and enhances application responsiveness, crucial for real-time analytics, video streaming, and industrial automation. Secondly, it optimizes resource allocation, minimizing data transfers and ensuring efficient resource utilization. Moreover, it improves bandwidth efficiency, alleviating network congestion and enhancing overall resilience. Lastly, integration enhances data privacy and security, making it crucial for sensitive applications, while also providing scalability and adaptability for varying workloads. In summary, the Federation of Cloud, Edge, and Far Edge computing creates a dynamic ecosystem, optimizing performance, responsiveness, resource management, and data privacy. This holistic architecture empowers enterprises to deploy applications across diverse use cases efficiently, ensuring operational excellence and a competitive edge in the evolving digital landscape.

Considering these facts, in this deliverable the consortium primarily presents a thorough description of the high-level architecture of the Agile and Cognitive Cloud Edge Continuum Management (AC^3) framework. AC^3 has unveiled a state-of-the-art architectural design explicitly crafted to serve the evolving needs of the Cloud Edge Computing Continuum (CECC). This architecture is systematically segmented into three well-defined planes: the **User plane**, the **Management plane**, and the **CECC plane**. The User and Management planes function as integral components of the overarching CECC Manager (CECCM). Their primary role is to enable smooth and direct interactions with application developers who are venturing into the realm of cloud-native applications.

Importantly, the User plane Crafted with a focus on user-friendly interfaces, the user plane is the nexus for developers. It is fortified with tools and components such as the Application Gateway, which acts as a conduit for developers to engage with the CECCM. While the Service Catalog provides an extensive library of application descriptors and vital data source information. Furthermore, the Ontology and Semantic aware Reasoner stands out as an essential tool, decoding the intricate policies employed by diverse CECCM stakeholders. This plane is complemented with various interfaces, which are intended to facilitate basic operations for computing and networking activities as well as data-related tasks.

On the other hand, the Management plane is being positioned as the heart of the CECCM, which epitomizes administrative excellence. It is intended for seamlessly integrating features such as application and resource management, ensuring the smooth life cycle of applications while also overseeing the CECC infrastructure. The consortium has given a particular emphasis is laid on factors like energy conservation and strategic geographical positioning of the Management plane. Data management is yet another crucial aspect for AC^3, which is intended to govern by the stringent Gaia-X specifications. According to the consortium, the abstraction and federation layer stands out by streamlining the varied infrastructure layer CRUD API, making operations more intuitive.

The CECC plane is more than just a component—it is a testament to AC^3's vision of the future. It encapsulates the foundational infrastructure, housing vital elements like data sources and computing nodes. The design philosophy here pivots on the principles of federation, drawing inspiration and guidelines from established models such as NIST/IEEE for infrastructural needs and the revered Gaia-X model for data and service federation. A standout aspect of this plane is the empowerment it provides to CECCM owners. They are endowed with the autonomy to oversee and manage their resources—an autonomy that becomes particularly pronounced when cloud service providers take the reins of the edge infrastructure.

AC^3's commitment to data excellence is exemplified in its Data Management Platform-as-a-Service (PaaS). This platform adopts a three-pronged structural approach: **Northbound API** which is the custodian of the Data Catalogue, Monitoring module, and Data Federation Services. Together, they manage the exhaustive data inventory, ensure real-time monitoring of data streams, and regulate stringent data access measures. **Data Management Engine,** which is central to the PaaS and houses of the Data Manipulator, Internal Data Broker, and the Semantic Reasoner. These collectively facilitate data transformations, streamline data flow, and make informed, intelligent decisions rooted in semantic methodologies. While **Southbound API** focuses on integration, with tools like Data Mappers and Data Source Connectors. They guarantee seamless integration with external data repositories in AC^3, all while ensuring data consistency through a unified data format. To encapsulate, AC^3's architecture is a harmonious blend of innovation and functionality. It presents a robust and user-centric data management framework, simplifying complex data processes through its diverse modules and interfaces, setting a benchmark in architectural excellence.

The outcome of this document will be considered as an input for task T2.3 where all the relevant technological tools will be selected for implementing and developing the different components of the CECCM and Local Management System (LMS) in accordance with the pre-defined architecture model and target defined metrics. Later, the task T2.2 final output will be finalized the CECC framework and CECCM and will be documented in D2.2 on M24 of the project.

2 Introduction

In today's digitally interconnected environment, the increasing demand for real-time data processing, low latency interactions, and optimized user experiences has necessitated a cohesive integration of Cloud, Edge, and Far Edge technologies. This synergy is essential to develop a federated computing continuum that efficiently leverages centralized cloud resources while capitalizing on the immediacy of edge devices. This not only ensures seamless data flow and processing but also accommodates the expansive data influx from a myriad of sources. Considering these facts, the AC^3 project is set to redefine the landscape of federated computing with its groundbreaking architecture that encompasses a federated Cloud Edge Continuum infrastructure, extending naturally to the far edge. Central to this vision is the Cloud Edge Computing Continuum Manager (CECCM), which not only stands as the linchpin of the initiative but also emphasizes the criticality of resource federation, from centralized hubs to the farthest edges. This expansive integration aims to boost resource availability, with added trust and security mechanisms, ensuring that the entire system operates with enhanced resilience and adaptability.

A cornerstone of the AC^3 framework is its data management strategy. By integrating data management as a Platform as a Service (PaaS) within the CECCM, the project paves the way for smooth application development and deployment. This focus on data is comprehensive, covering facets from data retrieval to storage and monitoring. In enhancing its user-oriented approach, AC^3 is dedicated to delivering an intuitive interface, simplifying data requests through web semantics and ontologies. Alongside, there is an unwavering commitment to innovate with a zero-touch management and application Life Cycle Management (LCM) approach, harnessing machine learning, AI, and context-aware insights. This direction promises a boost in predictive capabilities, aiding the CECCM in making strategic decisions, from microservice placements to life cycle strategies. Lastly, sustainability and adaptability are core tenets of the AC^3 project. Through a green-centric, zero-touch approach to infrastructure management, buttressed by AI/ML insights, the project emphasizes energy-efficient operations. This drive towards sustainable practices ensures a balance between green energy use and maintaining application service levels. Complementing this is AC^3's emphasis on network programmability across the CECC landscape. By leveraging state-of-the-art technologies such as SD-WAN and CFN, the project ensures agile responses to network shifts, exemplifying the dynamic and responsive environment AC^3 aims to cultivate.

In this document, the consortium’s aim to delineate the preliminary version of overarching architecture of the Agile and Cognitive Cloud Edge continuum management framework, an intricately crafted system structured into three distinct planes: the User Plane, Management Plane, and Infrastructure Plane. Each of these planes is an ensemble of meticulously designed functional components that employ state-of-the-art, well-defined interfaces, facilitating seamless communication both within their respective planes (intra-plane) and between different planes (inter-plane). Central to this complex architecture lies the Cloud Edge Computing Continuum Manager (CECCM), envisioned as the cognitive nexus of the entire system. The CECCM is imbued with dual functionalities, overseeing both the precise management of application life cycles and the adept orchestration of Cloud Edge Computing Continuum (CECC) infrastructure resources. This dualism in function is not just a testament to its centrality but is underpinned by a synergistic amalgamation of Machine Learning (ML), Artificial Intelligence (AI), and cutting-edge semantic and context-aware algorithms. The finesse with which CECCM operates ensures a two-fold commitment: a staunch focus on energy efficiency, and unwavering adherence to the rigorous standards set by Service Level Agreement (SLA) benchmarks.

Further enriching the narrative, this document introduces the consortium's proposed initial iterations of the AC^3 framework's federation models, both for infrastructure and for data and services. To this end, the consortium has opted for the well-regarded IEEE/NIST model for infrastructure federation, a strategic move that equips the consortium to effortlessly implement a system that promotes the smart sharing and governance of resources across the vast expanse of the infrastructure ecosystem. In parallel to these

structural elucidations, the document sheds light on preliminary workflow schematics pertinent to application deployments and AC³'s unique data management Platform as a Service (PaaS). Such representations, detailed and exhaustive in their scope, provide invaluable insights into the mechanics of the AC³ framework and its holistic approach to Cloud Edge continuum management.

2.1 Mapping AC³ Outputs

The purpose of this section is to map AC³ Grant Agreement commitments, both within the formal Deliverable and Task description, against the project's respective outputs and work performed.

Table 1: Adherence to AC³ GA Deliverable & Tasks Descriptions

AC³ GA Component Title	AC³ GA Component Outline	Respective Document Chapter(s)	Justification
DELIVERABLE			
D2.1 1st Release of the CECC framework and CECCM			
TASKS			
SOTA of the related research works and innovation for defining the detailed specifications of the CECC architecture and identification of Functional and Non-Functional requirements	**Task T2.1:** The aim of this task is to conduct a detailed analysis on the techno-economic aspects of CECC resource federation, to identify the interactions between the stakeholders, as well as the new challenges that might arise in a federation model that goes beyond the NIST proposal. **Task T2.2**: This task will be dedicated to defining a detailed specification of the CECC architecture, key components, and features of the CECCM, and the required interfaces and protocols to perform federation and interact with the different infrastructures in a secure and trusted manner.	Section 3, Section 4, & Section 5.1	Revisiting the defined objectives and investigating the current related research and innovation initiatives (e.g., Gaia-X/IDSA, NIST, etc.), provides the guidelines for defining the functional and non-functional requirements
CECC Component's	**Task T2.2:** This task will specify the role of each	Section 4.2- 4.7, 4.9, Section 5.1	Present the initial functional architecture with detailed

specifications and interfaces	component composing the CECCM, by defining its functions, building blocks, and needed communication interfaces. Also, this task will select the federation model to be considered in AC^3, and accordingly define a unified API to be used by the infrastructure providers since AC^3 relies on a resource federation architecture involving different infrastructure types such as the core cloud, edge, far edge, network equipment, and data sources, from different un-trusted domains.		specification of functional blocks and identifies the inter-plane and intra-plane interfaces
Defining the Data management Model	**Task 2.2:** This task the data management model relying on the concept of PaaS will be defined in terms of functions and interfaces with the other CECCM components, the data sources and the application developer.	Section 4.8, Section 5.2- 5.4	Presented the initial data management model and also the sequence diagrams for the corresponding operations of data management model.

2.2 Deliverable Overview and Report Structure

In this section, a brief description of the deliverable's structure has been provided as follows:

- Section 3 provides a state-of-the-art of relevant initiatives and existing innovations which will help to develop the functional architecture for CECC. In this section, the consortium has mainly given their utmost focus on studying the latest Gaia-X/IDSA and NIST specifications for infrastructure and data federation.
- Section 4 initially provides all the functional and non-functional requirements for developing the CECC architecture. Then, it describes the initial version of the Functional architecture of the CECC framework. Moreover, a thorough description of different building blocks and architectural components is presented, including their functionalities. A detailed summarization of all inter-plane interfaces and intra-plane interfaces of all the functional components are documented in two separate tables.
- Finally, Section 5 presents the initial version of workflows and the interaction between the components included in the architecture regarding functionalities offered by the CECCM of AC^3.

3 Related Work

This section focuses on the study of relevant initiatives and existing innovations that will help to develop the functional architecture for AC^3. Following various ongoing projects, the AC^3 consortium has identified that certain latest specifications of Gaia-X and NIST can be considered as a guideline for developing the AC^3 functional architecture. At the intersection of these concepts, we have adopted a hybrid federation approach that combines elements from both Gaia-X and NIST. Gaia-X emphasizes secure cross-border data sharing and establishes a federated data infrastructure. NIST, on the other hand, focuses on efficient cloud resource allocation and management.

3.1 Gaia-X: an initiative to develop a federated secure data infrastructure

Gaia-X[1] is a European initiative that aims to create a federated digital ecosystem in which data and services can be created, integrated, and shared securely and openly according to common rules based on transparency, controllability, portability, and interoperability. It is worth mentioning that the Gaia-X Architecture supports the integration of data-infrastructure ecosystems by combining the features described in the Gaia-X Conceptual Model, the Gaia-X Operational Model, and the Federation Services, as well as the Gaia-X Trust Framework.

The participants in the Gaia-X ecosystem commit to adopting and fulfilling a set of common rules that Gaia-X defines regarding the architecture, policy, labelling, and compliance for data and services to achieve the objectives of the initiative. These guidelines are described in the Gaia-X Trust Framework[2] document.

The Gaia-X Trust Framework foresees to use verifiable credentials and linked data representations as the cornerstone of its future operations. This entails that, for ensuring the alignment of the project with Gaia-X, the **AC^3 stakeholders will be required to describe not only themselves as participants but also the services they offer** by writing credentials signed and verified through the Gaia-X compliance web-portal (gx-compliance)[3] and the trust anchors[4]. These specifications that set how Gaia-X works in terms of functional and technical requirements, infrastructure, data portability, data consent for managing, data traceability, credential management, policy negotiation, identities, and so on are developed by different working groups that publish their outcomes in Gaia-X deliverables[5]. In parallel, several Gaia-X initiatives named “lighthouse projects” implement the guidelines included in these documents into practical use cases.

Within its objectives and goals, the AC^3 project also promotes the definition of procedures that apply to such technical and functional requirements, so that this deliverable could be used as a reference documentation for exploring compatibility between both initiatives. Besides that, one of the main aims of the AC^3 project is to adopt the guidelines provided by the Gaia-X project on data/infrastructure federation for ensuring a trustworthy environment to share the data among the different participants of AC^3.

The Gaia-X ecosystem consists of domain specific governance with conformant data or infrastructure providers and consumers which use federation services to exchange data, services and infrastructure. Thus, in the AC^3 project, the consortium has planned to adopt the Gaia-X provided guidelines for Federation to enable and facilitate interoperability and portability of Services and Resources, including Data Resources. The primary goal of the federation is to enable and facilitate resource interoperability and portability within and across the AC^3 framework, as well as to provide Data Sovereignty by ensuring trust between or among Stakeholders/Participants to make resources searchable, discoverable, and consumable, and to provide means for Data Sovereignty in the AC^3-provided distributed ecosystem environment.

[1] https://gaia-x.eu/
[2] https://gaia-x.gitlab.io/policy-rules-committee/trust-framework/gaia-x_trust_framework/framework/
[3] https://compliance.lab.gaia-x.eu/development/docs/
[4] https://gaia-x.gitlab.io/policy-rules-committee/trust-framework/trust_anchors/
[5] https://gaia-x.eu/what-is-gaia-x/deliverables/

Following the Gaia-X standards, the envisioned AC^3's CECC architecture must rely on three conceptual pillars to provide data sovereignty, transparency and trustworthy data sharing among various stakeholders/participants. The three pillars are as follows: (**a**) Gaia-X Compliance for decentralized services to enable objective and measurable trust; (**b**) Data Spaces / Federations: Interoperable and portable (Cross-) Sector datasets and services; and (**c**) Secure and trustworthy data exchange via anchored contract rules for access and data usage. To comprehend the above conceptual pillars, the AC^3 envisioned CECC framework must adhere to the Gaia-X provided guidelines for developing the **Trust Framework** and **Labelling Framework.** The **Trust framework** will enable the safeguarding of data protection, transparency, security, portability, flexibility, sovereignty and European Control by using verifiable credentials and linked data representation. Whereas the **Labelling framework,** jointly with the Gaia-X Compliance and rules, can flexibly accommodate the specific and evolving requirements that any entity can define to obtain their desired level of trust in the AC^3 ecosystem. Thus, for that purpose, CECC will follow four types of rules:

- serialization format and syntax
- cryptographic signature validation and validation of the keypair-associated identity
- attribute value consistency
- attribute veracity verification

One of the objectives of this document is to specify the initial version of the CECC framework and ensure data sovereignty, transparency, and trustworthy data sharing between different stakeholders/participants of the AC^3 ecosystem. Therefore, it is relevant here to reassess some of the key concepts of Gaia-X innovations that the AC^3 consortium has planned to adopt into the envisioned CECC framework.

3.1.1 Gaia-X – Service Federation

The primary goal of Gaia-X Federation Services is to enable and support the interoperability and portability of Services and Resources, especially Data Resources, inside and across entire ecosystems. According to Gaia-X, Federation Services do not conflict with the economic models of other ecosystem members, particularly Providers and Consumers. The Federation is a loose group of interacting actors/participants who consume, produce, or offer linked Resources directly or indirectly. A Federator, a Gaia-X ecosystem player, is one of the primary facilitators of Federation Services. It is worth mentioning that multiple Federators can participate in the ecosystem and each of them are independent. Federators make Federation Services possible by mandating Federation Service Providers to offer specific Federation Service Instances. The Federation Services are formed by the aggregate of all Federation Service Instances. Federation Services, according to the Gaia-X, must comply with common standards to provide interoperability, portability, and data sovereignty across multiple ecosystems and communities. These standards (for example, those relating to service Self-Description, digital identities, data sharing transaction logging, and so on) must be unambiguous and are thus specified by the Gaia-X Association. The Trust Framework and any related legislation or governance components are owned by the Gaia-X Association. Different entities may take on the role of Federator and Federation Services Provider. Federation technically enables connections and a web of trust between and among different parties in the ecosystem(s). It is important to understand that the services will not be provided by a central authority, but that each Federation will be able to use the reference open-source code of the Gaia-X Federation Services toolbox to then build apps and services that match the requirements in their respective Federation. Notably, by adhering to the Gaia-X guidelines for data-infrastructure federation, AC^3's ambition is to develop a trustworthy, sovereign and transparent ecosystem.

3.1.2 Gaia-X – Data Spaces

It is critical for the success of a Federated ecosystem that data services and the underlying infrastructure can communicate seamlessly with one another. As a result, portability and interoperability are two critical needs for Gaia-X's success because they are the foundations of a working platform and provide a fully functional federated, multi-provider environment. Gaia-X endorses Data Spaces, which presents a virtual data

integration concept, where data is made available in a decentralized manner, for example, to combine and share data stored in different cloud storage backends. In general, data ecosystems allow participants to use data as a strategic resource in an inter-organizational network without being constrained by a fixed specified partner or central keystone company. Data must be made available in cross-company, cross-industry ecosystems for its full potential to be realized. As a result, data ecosystems not only enable significant advances in the data value chain, but also provide the technical means to achieve Data Sovereignty. Such sovereign data sharing addresses multiple tiers and allows for a wide range of business models that would otherwise be impossible. Trust and control mechanisms promote the acceleration of data exchange and the spread of ecosystems. Through the development of data space connectors (e.g., Eclipse Dataspace Connector/ IDS Connector), Gaia-X, along with IDSA and Eclipse foundation, ensured to build such a sovereign data ecosystem for many vertical industries. Following that innovation, AC^3's will also aim to develop such a data ecosystem for enabling transparent data sharing between multiple stakeholders/participants.

3.2 NIST & IEEE for Federation

Cloud Computing has emerged as a fundamental framework for societal operations, playing a pivotal role in supporting various domains, including autonomous vehicles and the Internet of Things. Therefore, standardization is crucial for deep interoperability and federation. This interoperability allows service providers to collaborate and connect with each other seamlessly. Indeed, a federation is a virtual security and collaboration context that is not necessarily owned by any one user or organization. In what follows, we will introduce the federation designed by the NIST [6] and the IEEE SIIF[7] and how it can be relevant to AC^3 project.

3.2.1 NIST CFRA

The National Institute of Standards and Technology proposes the Cloud Federation Reference Architecture (CFRA) [1] shown in Figure 1. The goal of this conceptual model is to identify the fundamental federation functions that may be important to different participating stakeholders in different application domains.

[6] https://www.nist.gov/
[7] https://www.ieee.org/

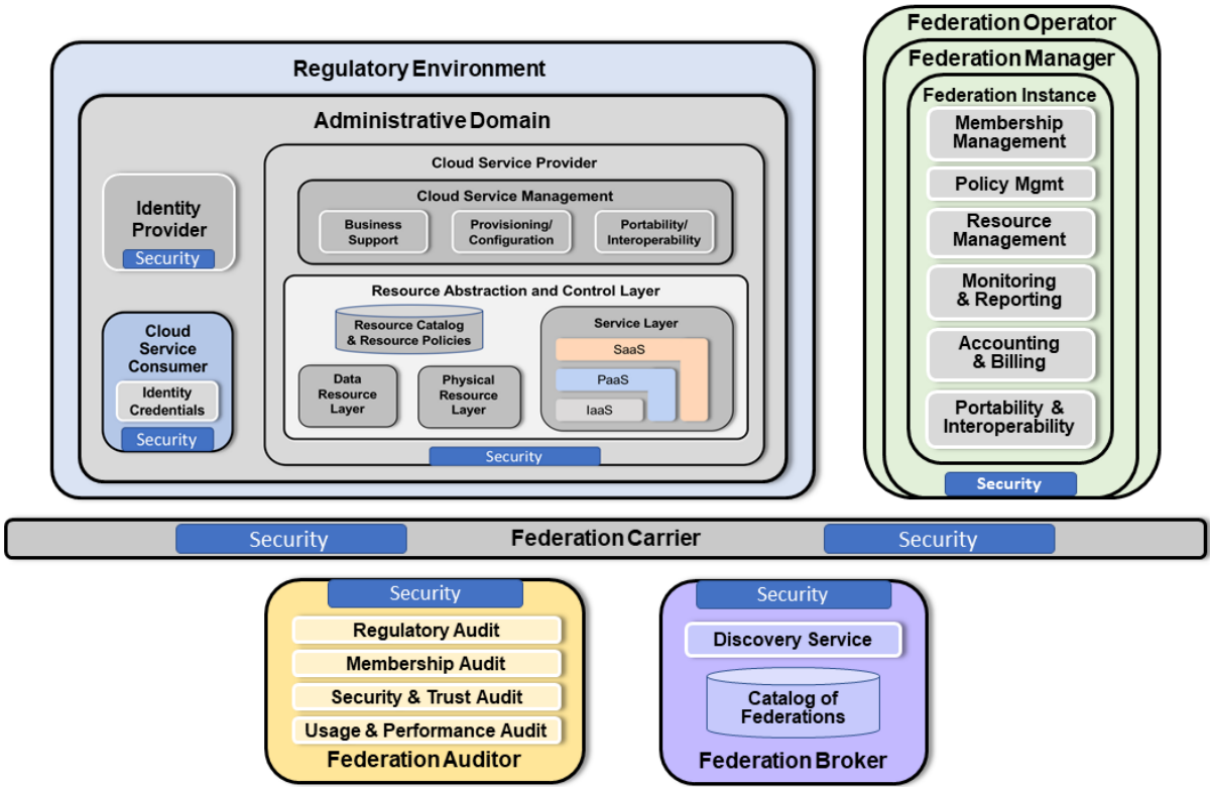

Figure 1. The NIST Cloud Federation Reference Architecture Actors [1]

In this architecture, **Administrative Domains** (AD) typically operate as independent, autonomous environments. The domain administrators will issue identity credentials, deploy services, and define the policies for who can access what. An AD is essentially comprised of:

- An **Identity Provider**, where there are many different types of identity providers and many different types of identity credentials that they issue to users. A user's identity is associated with a number of roles or attributes. Resource access policies can be defined based on these roles or attributes.
- A **Cloud Service Provider** that includes all the components responsible for Cloud Service Management, such as the Business Support, and for Resource Abstraction and Control by providing an abstraction that enables it to effectively manage all types of resources.
- And a **Cloud Service Consumer** or simply user, representing a person or organization that has a business relationship with, and uses the services from, a Cloud Service Provider.

All ADs exist within a **Regulatory Environment**. Meaning all users and service providers exist within the jurisdiction of governmental entities and must observe all relevant regulations defined by those entities.

For federation purposes, the architecture introduces the **Federation Manager** (FM). The FM is the conceptual entity that offers essential management functions throughout the lifespan of a federation. An FM can support multiple federation instances that may encompass multiple Administrative Domains.

In practical deployments, the FM is not necessarily a single, separate third party. Federated environments may consist of one or more FMs, each of which are operated by a Federation Operator, but a single Fed Operator may operate multiple FMs. FMs may exist in centralized or decentralized deployments. As the scale and magnitude of the federation increases, the presence and activities of the Fed Operator will become more pronounced. These are all, however, deployment issues.

The Federation Manager is operated by a **Federation Operator.** This entity has the capability to manage, maintain, and oversee multiple Federation Managers. This entity is depicted as superior to the Federation

Manager and Federation Administrator. At sites that participate in multiple separate and distinct federations, a Federation Operator will coordinate the activities of the Federation Managers and provide administrative support and maintenance by collecting, processing, and resharing individual federation metadata while following the common policies and legal frameworks shared between federations. However, not all cloud federations have a need for a Federation Operator. In simpler instances, the Federation Manager may be as simple as a server that does the simple management of a federation. The rest of the components present in the architecture are described with details in the **NIST Special Publication 500-332** [1].

3.2.2 IEEE SIIF

IEEE Standard for Intercloud Interoperability and Federation (SIIF) [2] extends the NIST CFRA into a concrete federation hosting service (FHS) based on three RESTful APIs using best practices of Open API Specification [1]. In this standard, the functionality of the conceptual federation manager proposed by the NIST Cloud Federation Reference Architecture is provided by an FHS. This standard identifies and examines all issues concerning the implementation and deployment of practical FHSs. Practical FHSs can exist in a range of deployments: from simple, bare-metal deployments, to distributed edge computing environments, to on-demand, cloud-native deployments with a global footprint. An FHS is capable of hosting one or more federation or federation instances, where several roles are identified:

- **FHS provider role**: An FHS provider is a cloud provider that can host extensible federations replete with computing services or information products. FHS providers may be a cloud provider and offer their own cloud services for use in federations, or they may be a cloud broker and re-offer services provided by other cloud service providers. In the case where one or more federation members provide cloud resources and services to the federation, the FHS provider is a cloud broker that offers Service Intermediation or Service Aggregation to enable federation operations.
- **Federation hosting services operator (FHSOperator) role**: Every FHS is deployed and operated by an FHSOperator. The FHSOperator can be an end-user or non-cloud operator that deploys one or more FHSs for their own use. It may also be a cloud provider that offers FHS services on-demand in the cloud. Finally, it can be a cloud broker and re-offer services provided by other cloud service providers.
- **Federation members**: Within any federation instance, there are three predefined member roles:
    - **Federation administrator**: A FedAdmin is a special privileged account at an FHS that has the authorization to create, manage, and terminate federation instances. FedAdmins are members of the federation they create and administer. FedAdmins can grant and revoke federation membership, along with authorization roles and attributes.
    - **Service/Resource owner**: A ServiceOwner is a member of a federation that has the authorization to register services or resources, making them available within a federation. ServiceOwners have the authority to define and update the discovery unilaterally and access policies for their resources based on the roles and authorization attributes known to the federation. This authorization is granted by the FedAdmin
    - **Member**: A federation member can discover and access resources within a federation based on the roles or attributes that have been granted. While member is a general category, a FedAdmin can create as many roles as possible and attributes necessary to differentiate what different members can do.

3.2.2.1 FHS Reference Model:

While the FHS provides a logical set of functions to manage federations, from a software design perspective, FHSs may have a centralized or distributed implementation. As per the CFRA, a centralized implementation is possible and would be easier to deploy and operate. However, for reasons of scalability and supporting different governance models, distributed implementations should be considered. These considerations produce the federation hosting service model, as illustrated in Figure 2. FHS A and FHS B can take on a wide range of actual implementations. They can be two different federations provided by the same FHS at two or more locations, they can be two FHSs each providing a federation from differing locations or one FHS, providing two federations from the same location.

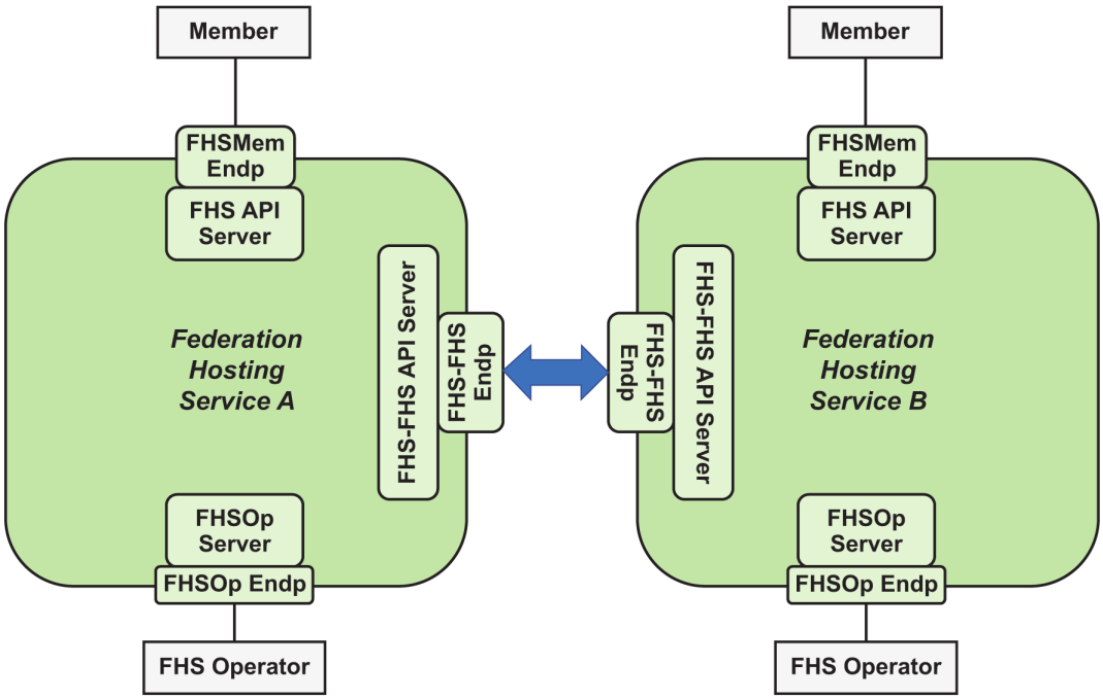

Figure 2. Federation hosting service reference model [2]

This standard defines a RESTful service API. Each FHS has the following three APIs:

— **The FHS Operator API**: Each FHS is a service that should be operated just like any other service, such as a web server. Hence, an FHS Operator instantiates or deploys an FHS and maintains its proper functioning. However, the FHS Operator does not directly manage individual federation instances.

— **The FHS Member API**: This is the API through which federation members interact with their federation. This includes the three pre-defined member roles: FedAdmin, ServiceOwner, and member.

— **The FHS-FHS API**: Since distributed implementations should be considered, this API is used by FHSs to exchange information about specific federation instances. As with all distributed systems, any FHS implementations will have to address latency, consistency, and fault tolerance issues.

3.2.3 Adoption of IEEE SIIF and NIST CFRA by AC³

In section 4, which describes the AC³ Functional architecture, the Adaptation & Federation layer acts as the interface between the management functions of the CECCM and the federated infrastructure. The adoption of the IEEE SIIF architecture by AC³ will be enabled via the Resource Discovery module; the latter will act as the communication point with the Federation hosting Service to discover the service available in the federation. Note that The CECCM will be a member of the federation with permission to discover and use services offered by the resource infrastructures.

For a resource Infrastructure to be part of the managed CECC, it needs to join the federation created at the FHS level with a service/resource owner membership. Once it joins the federation, the main services provided by the resource infrastructure described in section 4, which are the NBI service and the Resource exposure service of the Local Management System, need to be registered at the FHS level. From there the Resource Discovery module will get the services information and provide it to the Adaptation Gateway at the Adaptation & Federation layer.

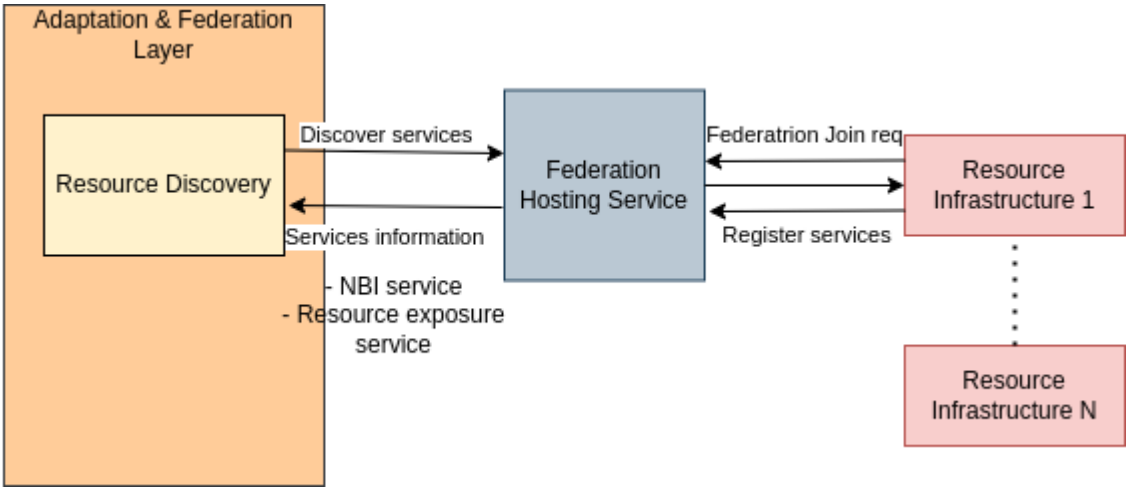

Figure 3. Interface between CECCM and FHS

Finally, we consider the Federation Hosting Service as an external entity to the CECCM Framework that needs to be operated and administered as an independent component. Figure 3 depicts the envisioned interaction between CECCM and FHS.

3.3 Other existing initiatives

While Gaia-X and NIST propose frameworks, policies and reference architectures for addressing challenges in IT, such as data privacy, security and federation, in an increasingly digital world, they aim to design how different functional components behave against a well-defined set of goals. There are other organizations like the Cloud Native Computing Foundation (CNCF) which is a non-profit organization that seeks to advance the development and adoption of cloud-native technologies, by maintaining an ecosystem of upstream projects related to multiple topics.

The CNCF provides a vendor-neutral home for open-source projects that are essential for building and running cloud-native applications. It fosters collaboration among developers, end-users, and vendors to drive innovation and standardization in the cloud-native ecosystem. By promoting collaboration, standardization, and open-source development, the CNCF plays a crucial role in accelerating the adoption of cloud-native technologies. It enables organizations to build scalable, resilient, and portable applications that leverage containerization, microservices, and dynamic orchestration systems like Kubernetes. One of the main advantages of CNCF is that it hosts several projects with practical implementations, most of them open source and accessible by anyone interested in adopting them or contributing to them.

4 AC³ Functional Architecture

This section is dedicated to describing the overall functional architecture of AC³. Before presenting the preliminary version of the functional architecture, the consortium has identified functional and non-functional requirements which must be satisfied and addressed by the envisioned functional architecture.

4.1 Requirements

4.1.1 Functional Requirements

We defined a list of 21 requirements that we believe are representative of what the CECCM functional blocks should consider and provide to run micro-service-based applications on top of a federated Cloud Edge Computing infrastructure. These requirements constitute the basis of the initial architecture of the AC³ project.

- **Req. 1:** CECCM should support GUI-based and API-based CRUD operations exposed to the application developer.
- **Req. 2:** CECCM should support micro-service-based applications.
- **Req. 3:** CECCM should provide blueprints for SLA and application deployment.
- **Req. 4:** CECCM should support security methods (authentication, privacy, and trust).
- **Req. 5:** CECCM should use computing and networking resources that use green energy.
- **Req. 6:** CECCM should support automated and zero-touch management of the Lifecycle of micro-service-based applications.
- **Req. 7:** CECCM should operate over a federation of resources and services.
- **Req. 8:** CECCM should allow the definition of micro-service-based applications using intents.
- **Req. 9:** CECCM should expose PaaS for data management.
- **Req. 10:** CECCM should use CRUD/Monitoring exposed by Cloud.
- **Req. 11:** CECCM should use CRUD/Monitoring exposed by edge Cloud.
- **Req. 12:** CECCM should use CRUD/Monitoring exposed by far Cloud.
- **Req. 13:** CECCM should use the API exposed by SDN controllers to request a route update for a given application.
- **Req. 14:** CECCM should discover and access data sources and lakes.
- **Req. 15:** CECCM should support resource abstraction provided by the heterogeneous LMS.
- **Req. 16:** CECCM should dynamically discover and access new LMS resources.
- **Req. 17:** CECCM should support and connect to data (hot data) provided by IoT devices.
- **Req. 18:** LMS should support the capability to expose the resource utilization (such as CPU, memory, and storage).
- **Req. 19:** LMS of Data source should expose methods to access and discover data.
- **Req. 20:** LMS Cloud should expose CRUD/Monitoring functions to CECCM.
- **Req. 21:** LMS Edge Cloud should expose CRUD/Monitoring functions to CECCM.

4.1.2 Non-functional Requirements

- **Availability:** The system should be highly available, with minimal downtime or disruption to users even if there is some degradation in any IoT devices or cameras
- **Compatibility:** The system should be compatible with existing cloud/edge architectures.
- **Compliance/Trust:** The system should be accepted by main stakeholders in the cloud/edge business.
- **Cost:** The systems should operate in a cost-effective and energy-efficient way, with the ability to scale up or down based on the needs of the business.
- **Data replication and backup:** The file storage should be designed to ensure the data replication and backup, ensuring the generated contents can be retrieved in case of storage failure.

- **Extensibility:** The system should be open to easily update with new functionalities and hardware compatibility (e.g., new sensors).
- **Flexibility:** The system should be flexible and able to support changes in business requirements or new features.
- **Maintainability**: The system should be easy to maintain and update, with minimal downtime or disruption to users.

4.2 Overall Functional Architecture

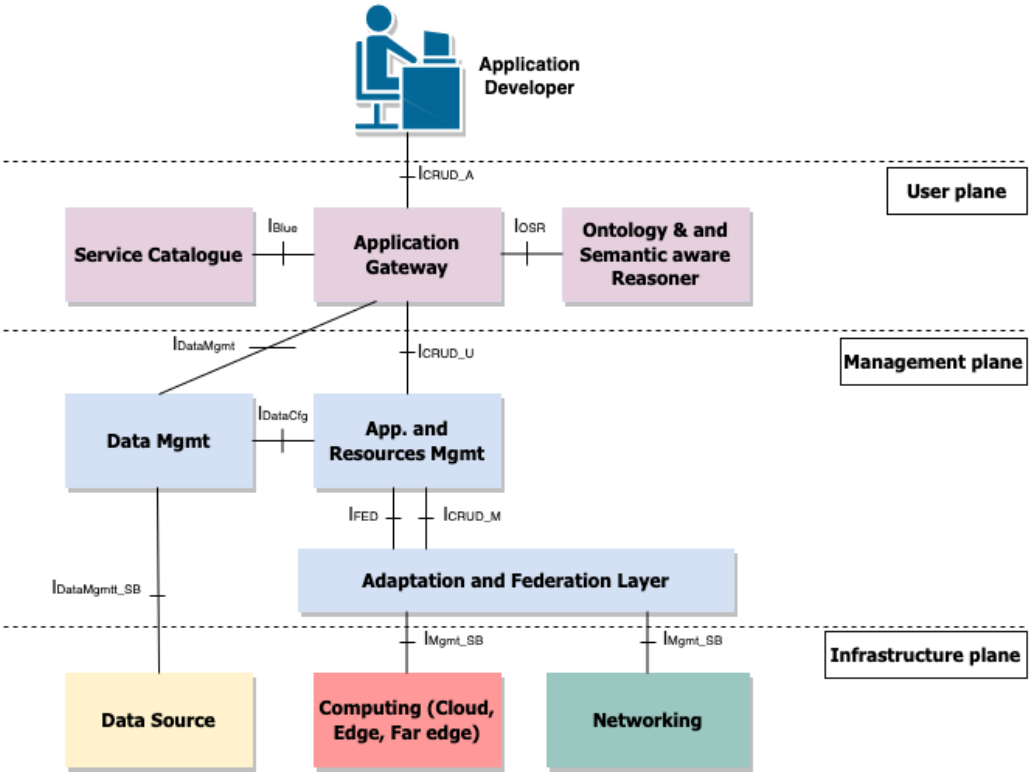

Figure 4. High-level architecture of AC³

In this section, we will introduce the high-level architecture adopted by AC³, the key components, and the different interfaces. The high-level architecture adopted in AC³ for the agile and cognitive management of the Cloud Edge Computing Continuum (CECC) is illustrated in Figure 4. The proposed architecture is composed of three planes: the user plane, the management plane, and the CECC plane. Each plane has its own components and uses well-defined interfaces to communicate with the other planes. While the user and management plane are components of the CECC Manager (CECCM), the CECC plane corresponds to the infrastructure constituting the CECC, i.e., Data source, computing nodes (central cloud, edge, and far edge). The innovation behind the user plane is to group all the necessary functionalities to allow an application developer to build data-driven and micro-service-based applications using a common descriptor to define needed resources in terms of data, compute, and networking. To ease this task the user plane includes templates and blueprint to define applications and describe SLA, while data management is provided as PaaS. The management plane is fully automated through the usage of AI/ML algorithm to achieve zero touch management. The innovation of AC³ consists of building application profiles and infrastructure profiles in order to predict the performance and optimally deploy applications' components over the cloud edge continuum. The resource management mechanisms combine AI and eXplainable AI (XAI) to derive optimal decisions to ensure SLA of the application. In AC³, the infrastructure resources are built using a federation as

stated and detailed earlier. It uses NIST/IEEE model for the infrastructure federation and Gaia-X for data and service federated. It should be noted that the architecture also considers the case that the CECCM owner can also own and manage its own infrastructure resources. Here, we can mention the case of a centralized cloud provider that also owns edge infrastructure in certain locations. In this context, AC^3 innovates by adding a federation layer that abstracts the CECC infrastructure and eases the integration of new resource provider. The federation layer includes necessary mechanisms to enable dynamic discovery of resources and their integration to the resource pool of the CECCM.

The user plane: The first plane includes the components that interact directly with the application developers. Here, the application developers can correspond to the application owners, such as a vertical application (Industry 4.0 or video platform). The only requirement here is that the applications are cloud-native, relying on the concept of micro-services. It includes the Application Gateway that allows an application developer to interact with the CECCM to develop, deploy, and manage applications' life cycle. The service catalog includes a blueprint of the application's description, which can be extended or adapted by the application developer to create new applications to be deployed by the CECCM. The service catalog also includes information on data sources, particularly cold data (i.e., data lake) federated using the Gaia-X approach. Finally, the last component of the user plane is the Ontology and Semantic aware Reasoner that translates and interprets all policies used by different CECCM actors (e.g., data source and application developers).

On the other hand, the user plane, and particularly the Application Gateway, interacts with the application developer using the I_{CRUD_A} interface, which allows the classical Create Read Update and Delete (CRUD) procedures. This interface can be in the form of a Graphical User Interface (GUI) and a set of Application Programming Interfaces (API) that can be called by another system. Moreover, the Application Gateway will act as a translator of the intent expressed by the application developer for SLA and application descriptors. Indeed, in AC^3, the SLA and application description can be done using intents, mainly human-expressed text, which will be translated by the Application Gateway into machine format (YAML or JSON) forwarded to the management plane. Regarding the interactions with the latter, two interfaces are envisioned: I_{CRUD_U} for forwarding and adapting the procedures related to CRUD focusing on computing and networking, which will be enforced by the application and resource management component of the management layer; $I_{DataMgmt\_SB}$ to interact with the data management component that provides PaaS for everything related to hot and cold data connection and interaction.

The management plane: It is the core function of the CECCM. It integrates all the necessary management and orchestration functions, implementing data-driven and AI-based solutions, to handle both the Life Cycle Management (LCM) of applications and their related data, as expressed by the application developer through the user plane, while considering the CECC infrastructure resources. The management plane includes three key components:

- Application and resource management: it is in charge of the LCM of the applications and the CECC infrastructure management and orchestration. It is a termination point of the interface I_{CRUD_U} towards the user plane to (i) instantiate applications over the CECC infrastructure by running placement algorithms considering the application profile, the infrastructure resources, energy consumption, geographical locations, etc.; (ii) handle the runtime of the application to guarantee SLA, by considering micro-service migrations and resource scalability, network programmability (e.g., update routes); (iii) interfacing with the abstraction and federation layer (through I_{FED} and I_{CRUD_M}) to select a resource from the federation and enforce LCM decisions on top of the existing CECC infrastructure; (iv) interface with the federation and abstraction layer to monitor the application behavior by measuring KPI from the CECC infrastructure.
- Data management: its role is to manage access to cold and hot data. Concerning hot data, the data management module will execute all the necessary procedures allowing to register, connect and extract data from sensors owned by the application developer or a third tier IoT provider. For both hot and cold data, the data management will follow the Gaia-X procedures to access data spaces and IoT data sources federated with AC^3.

- The abstraction and federation layer: in AC^3 we envision that the owner of the CECCM may use federated resources (computing and networking) from different infrastructure providers, thus the abstraction and federation layer role is needed. Its role is to abstract the heterogenous infrastructure layer CRUD API (southbound) to the application and resource management component. To this aim, the abstraction and federation layer will expose common API (I_{CRUD_M} and I_{FED}) in order to: (i) allow resource discovery of the federated infrastructure; (ii) CRUD over the federated CECC infrastructure; (iii) translate I_{CRUD_M} to infrastructure specific API via the I_{Mgmt_SB}.

Federated CECC infrastructure plane: AC^3 envisions a federated CECC infrastructure that allows the CECCM to use different infrastructure providers to deploy micro-service-based applications. The federated CECC infrastructure is composed of Public/Private cloud resources, edge resources, far-edge resources, and networking resources. The resources can belong to the owner of CECCM and use federation to complement resources (no more edge resources available in a specific location), or the CECCM can be an entity that does not own infrastructure but relies on the resource federation. As indicated earlier, AC^3 will use the NIST model to build the infrastructure federation. The next section will give more details on the key functional blocks of the CECCM.

4.3 Application Gateway

The Application Gateway (AG) sits atop the CECC architecture, and it will provide a Graphical User Interface (GUI) and an Application Programming Interface (API) to provide developers with a way to execute the CRUD operations. The specification of the GUI will be tightly coupled with the functionalities of the Application Gateway, exposing the underlying available CECCs capabilities to the developer in a very intuitive and visual manner. The CRUD operations provided by the Application Gateway are:

- Create: It consists of creating the application and deploying it on the CECC. It consists of specifying the application components, requirements (Service Level Agreement), on-board, and instantiates the micro-services composing the application. The CECCM assist through the Application Gateway to specify the data needed by the application through the data management PaaS. It should be noted that the onboarding starts by translating application intents into corresponding SLAs, which it then uses to build the application descriptor. In this setting, one single descriptor generated by the AG should be sufficient to execute and deploy any application. These intents and the resulting descriptors should contain information about the amount of resources requested by the application and the type of resources, as well. The application developers can also specify desired policies for scaling and certain aspects of the application management that they consider adequate for the application. However, the ultimate policies regarding application management will be obtained from the AI-based LCM management mechanisms that perform these tasks posterior to application onboarding.
- Read: It consists of collecting and visualising the data regarding the performances of the application allowing the application developer to track the SLA status or update the resources assigned to the micro-services composing the application.
- Update: It consists of updating: (1) the assigned computing (CPU, Memory, Storage) assigned to the application's micro-service; (2) the networking resources (update the SD-WAN routes).
- Delete: It consists of removing and off boarding the resources assigned to the application's microservices.

4.4 Application and resource management

As stated earlier, one of the key components of the CECCM is the application and resource management component that handles the LCM of the micro-service-based applications as well as the resources dedicated to the applications using the federated CECC infrastructure. Figure 5 shows the internal structure of the

component and its constituting modules, which contribute to the objectives of handling the application life cycle and their dedicated resources (data, computing and networking). These modules are:

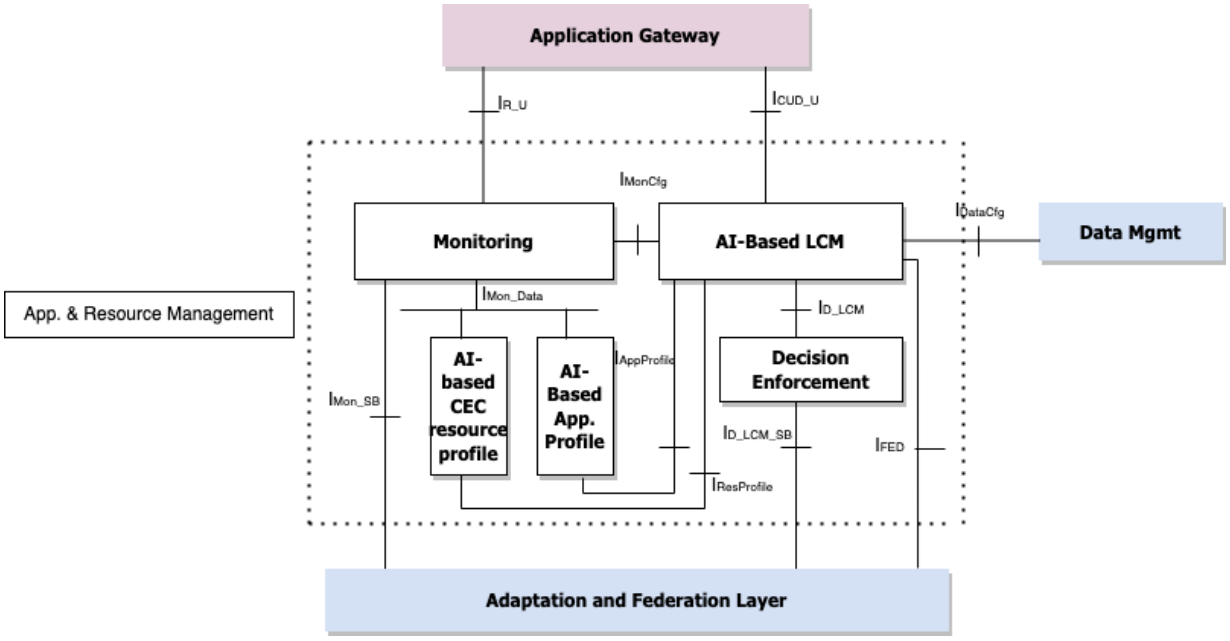

Figure 5. Application and resource management architecture

4.4.1 AI-based LCM

This module handles the LCM of the micro-service-based application, which consists of instantiating applications on the CECC infrastructure and managing their resources during their life cycle. This module receives a request to create an application (i.e., instantiate the application) through the I_{CUD_U} (sub-interface of I_{CRUD_U} introduced in the precedent section) interface in the form of a descriptor (e.g., JSON or YAML). The descriptor will include the software image of the application (i.e., several images, one for each micro-service) along with information on the SLA, such as the needed CPU, memory, networking (e.g., latency, bandwidth), availability, etc. Note that, the application description and SLA are translated by the Application Gateway from a high-level language (close to human language).

Besides these inputs, the AI-based LCM will use inputs from the AI-based App. Profile and AI-based CEC resource profiles modules to decide about the initial placement of the micro-services composing the application. Indeed, the latter modules provide information on the application profile, if it exists, such as the type of traffic, the pattern of the traffic, data-intensive, etc.; and the status of the CECC resources available for the CECCM, such as computing, type of energy used, networking, etc. Those inputs will be used by the AI-based algorithms to derive an optimal decision for placing the micro-services, thus ensuring SLAs, while optimizing the CECC resources (e.g., energy consumption). Apart from the initial placement, this module is in charge of the runtime management of the application, which consists of tracking the status of the application SLA (i.e., micro-service states), through the AI-based App. Profile and AI-CECC resource profiles to detect any degradation and react accordingly by, for instance, increasing the computing resources dedicated to micro-services (run a scale-up process) or migrating part(s) of the service from one computing to another computing node or updating the network routing configuration (e.g., request for more bandwidth). Finally, this module configures the level of monitoring of a newly created application (KPI to collect as well as the frequency to collect the monitoring data) to the monitoring module using I_{MonCFG} interface.

4.4.2 Monitoring

This module handles the collection of the monitoring data from the CECC infrastructure. It uses the API exposed by the adaptation and federation layer (I_{MonSB}) to configure and collect data on the computing resources, networking resources, and applications' used resources. All the collected data will be formatted using a common data model. The collected data will be made available to the AI-based Application profile as well as the AI-based resource management through a common communication bus (I_{Mon_Data}). The data is published using the *application_Id* or *infrastructure_Id* (Unique identifiers assigned by the CECCM to all applications and infrastructure under usage) as topics. While the set of KPIs to collect on the application performance are configured by the AI-LCM module when the application is instantiated, KPIs on the infrastructure are collected through the adaptation and federation layer, which collects this information through a resource discovery process.

4.4.3 AI-application profile

In this module, the AI and ML algorithms will run to create and build the application profiles, relying on measurements obtained from the monitoring module. The application profile will be first initiated using the application definition provided by the application developer and enriched by the data management module regarding the data sources and the storage policy. Then, the profile will be constantly improved through ML algorithms that monitor the application behavior to answer critical questions, such as what type of traffic flows between two microservices? When and where is the traffic and data generated? What is the data pattern? The built profiles will be made available to the AI-based LCM, which in conjunction with the SLA and the infrastructure available (or predicated) resources, realizes procedures related to the initial placement of the micro-services and during the run-time management of the application life cycle (e.g., migration, cloning, resource scaling). Application profiles are exposed to the AI-based LCM module using the $I_{AppProfile}$ interface.

4.4.4 AI-based CECC resource profile

Similarly, to the AI-based application profile, this module creates profiles of the CECC infrastructure used by the CECCM (owned or federated). The resource profile uses the consumed monitoring data to create the profiles, which may include current information on the available computing resources (i.e., CPU, memory, storage), CECC network link status (bandwidth, latency, reliability, etc.), and type of used energy (green, brown); but also predicted values that will be obtained using ML models. As stated earlier, the resource profiles will be combined with the application profile to drive LCM actions to be decided by the AI-based LCM module. Resource profiles are exposed to the AI-based LCM module using the $I_{ResProfile}$ interface.

4.4.5 Decision enforcement

This module enforces the AI-based LCM module's decisions, such as deploying a micro-service (instantiate) on the selected infrastructure, migrating a micro-service from one infrastructure node to another, and removing a micro-service (stop). The decisions will be forwarded to the abstraction and federation layer to be translated to SBI request as exposed by the infrastructure manager. Further, the decision enforcement module should discover for a specific infrastructure the adaptation agents that should handle the LCM decision at the low level of the system. The communication with the federation layer is done through the interface ID, which is detailed in Table 2 and Table 3.

4.5 Adaptation and federation layer

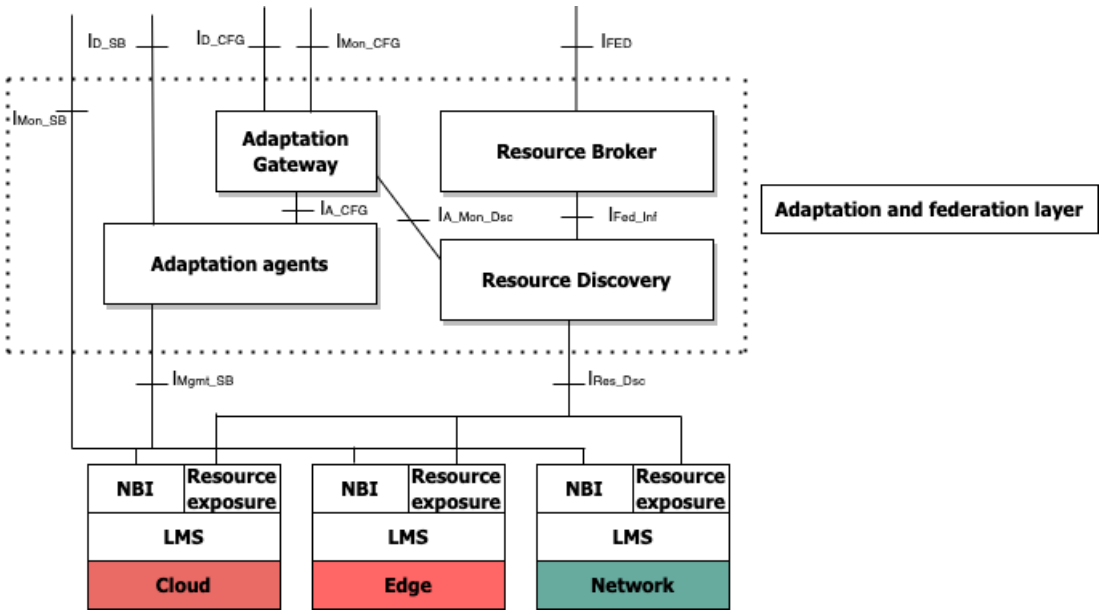

Figure 6. Adaptation and federation architecture

AC^3 highly relies on resource federation, which necessitates an adaptation and federation layer that acts as an interface between the management functions of CECCM and the federated infrastructure. In AC^3, we assume that each resource infrastructure is managed by its own LMS (Local Management System), which corresponds to the technology adopted by the infrastructure to manage and orchestrate the resources. For example, Kubernetes or OpenShift for Cloud and Edge based on cloud-native for computing and SDN controller for networking. Each LMS has its own NBI for CRUD operations, which is exposed to the adaptation and federation layer as well as the management layer (for monitoring). Further, each LMS exposes several pieces of information regarding the resources it manages, which can be discovered by the resource discovery algorithm of the adaptation and federation layer. The resource exposure of LMS may include information such as geographical location, geographical coverage, type of resources managed by the LMS, the amount of available resources, the characteristics of the resources, the supported QoS, the available KPIs to monitor, as well as the type of energy used (green or brown). Besides, the resource exposure of LMS integrates information on how to access the LMS resources (NBI description and API endpoints). Figure 6 illustrates a detailed view of the adaptation and federation layer, which is composed of:

4.5.1 Resource discovery module

The resource discovery module discovers and collects the available information on the different federated infrastructure resources using the I_{Res_Dsc} interfaces (connected to all LMS's resource exposure API). This information is then formatted in a common model and exposed via I_{Fed_inf} and I_{Mon_Dsc} to the Resource Broker and the Adaptation Gateway modules.

4.5.2 Resource broker module

The resource broker module selects an infrastructure provider when the management layer needs to deploy using a resource that needs to be complemented by a resource from the federation. This selection may be needed when, in the federation, more than one infrastructure can satisfy and provide the requested resources. The broker module integrates a selection algorithm that chooses the infrastructure that may host a micro-service.

4.5.3 The adaptation gateway module

The adaptation gateway module associates an adaptation agent with a LMS of an infrastructure, which will be used later to enforce LCM decisions issued by the management layer (i.e., the decision enforcement module). The agent is configured with the NBI endpoint of the LMS using the discovery module. Per the request of the decision enforcement module (using I_{D_CFG} interface) indicating the LMS of the infrastructure, the adaptation gateway provides the endpoint to use, which is typically an adaptation agent address. Finally, the adaptation gateway also exposes to the monitoring module the NBI endpoint exposed by LMS discovered via the resource discovery module.

4.6 Ontology and Semantic aware Reasoner

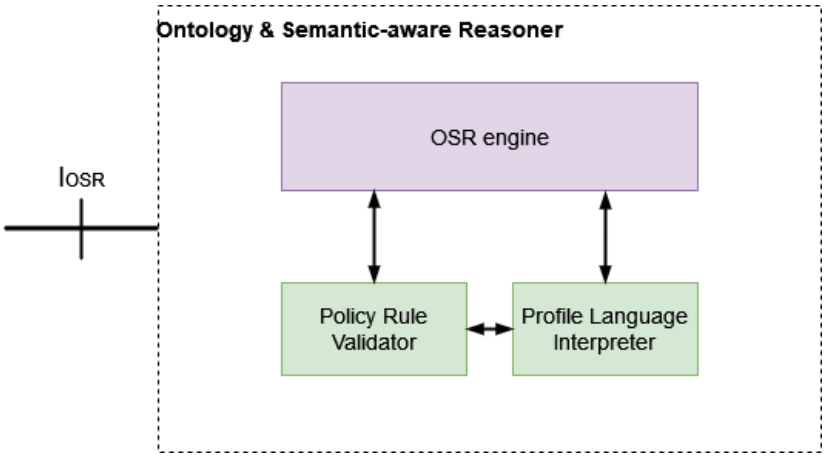

Figure 7. Ontology & semantic-aware reasoner architecture

The OSR is a major component of the AC^3 functional architecture. It functions as an intelligent reasoning engine, understanding, interpreting, and adapting policies and intents pertaining to the management of microservice-based applications in the cloud-edge continuum. The OSR makes use of semantic web technologies, such as ontologies and reasoners, to achieve a machine-processable representation of the cloud-edge computing domain, the microservices paradigm, and application management rules. The OSR takes as input the ontology defining the different actors in the system; such as data source, users, application, and Cloud-Edge Computing infrastructure. This ontology provides the semantic model that the OSR needs to understand the domain. The OSR also takes as input the policies and rules that define the behaviour of the different actors. These can be provided in a human-readable format using the newly defined templating language. The job of the OSR is then to translate these human-readable policies into a machine-processable format by translating them into ontology instances and semantic rules that can be interpreted by the reasoner using its Profile Language Interpreter (PLI) as shown in Figure 7.

The OSR uses a variety of reasoning techniques, including deduction, induction, abduction, and analogy, to derive new knowledge from the provided ontology and data. It can also integrate external data sources, such as databases, web services, and knowledge graphs, to enhance its inference capabilities. The OSR can detect dependencies and relationships between microservices and data sources by leveraging the provided ontology. It can also detect any conflicting or contradictory rules in the defined policies and either resolve or flag them to the user. The OSR also validates, verifies, and adapts provided policies to the runtime conditions of the application and cloud-edge infrastructure by using its Policy Rule Validator (PRV) as shown in Figure 7 that are used to properly manage the application. These outputs can be provided in a human-readable format through a dashboard, as well as in a machine-readable format to directly configure the cloud and application management platforms.

This component can support various input and output formats, such as RDF, OWL, JSON-LD, CSV, XML, and plain text. It can also generate visual representations of the ontology, such as diagrams or graphs, to help

users better understand the structure and relationships of the domain's concepts. Finally, its main purpose is to introduce an intelligent semantic layer between the human-defined policies and the actual application runtime environment. By understanding the semantic meaning and relationships between the different management concepts, the OSR can reason about the policies and adapt them to achieve the overarching goals related to optimizing the application performance, efficiently utilizing resources, and ensuring a good quality of service. The OSR transforms what would otherwise be static policies into dynamic, context-aware, and optimized application lifecycle management.

4.7 Service Catalogue

The Service Catalogue is located near the top of CECC architecture and will provide a listing of all the applications available in AC^3 along with their capabilities, requirements and dependencies. It also plays a significant role in the AC^3 functional architecture. It serves as a central component designed to assist AC^3 developers in managing and maintaining the AC^3 blueprints. The purpose of the Service Catalogue is to provide developers with a unified view of all their software, services, libraries, machine learning (ML) models, and datasets. The Service Catalogue functions as a showcase area where all available blueprints are displayed. It keeps track of ownership and metadata associated with these blueprints, making them easily discoverable for all users who access the catalogue. This enables developers to have a comprehensive understanding of the available services within the AC^3 environment.

The CRUD operations provided by the Service Catalogue are:

- Create: It consists of registering the application blueprint in AC^3.
- Read: It consists of providing the information of a specific application blueprint.
- Update: It consists of updating: (1) the blueprint's metadata (e.g., name, description, contact information, etc.); (2) the computing details of the blueprint (update the applications business logic).
- Delete: It consists of removing and off boarding the blueprint of the application.
- Search: It consists of listing all the applications that conform with specific user-defined criteria and returning the appropriate blueprints that can be used.

To facilitate this functionality, the catalogue relies on metadata files that are stored alongside the corresponding code. These metadata files contain relevant information about the blueprints, such as their purpose, usage, dependencies, and other pertinent details. The Service Catalogue harvests and visualizes this metadata, presenting it in a user-friendly manner within the catalogue interface. The catalogue can be populated in two ways: statically or dynamically. In the static approach, the platform maintainers manually input and manage the information about the blueprints. In the dynamic approach, additional code repositories can be registered with the Service Catalogue, allowing new blueprints to be automatically added and displayed as they become available.

Overall, the Service Catalogue serves as a valuable tool for AC^3 developers, providing them with a centralized location to explore and access the various blueprints while also managing ownership and maintaining essential metadata for each resource.

4.8 Data Management

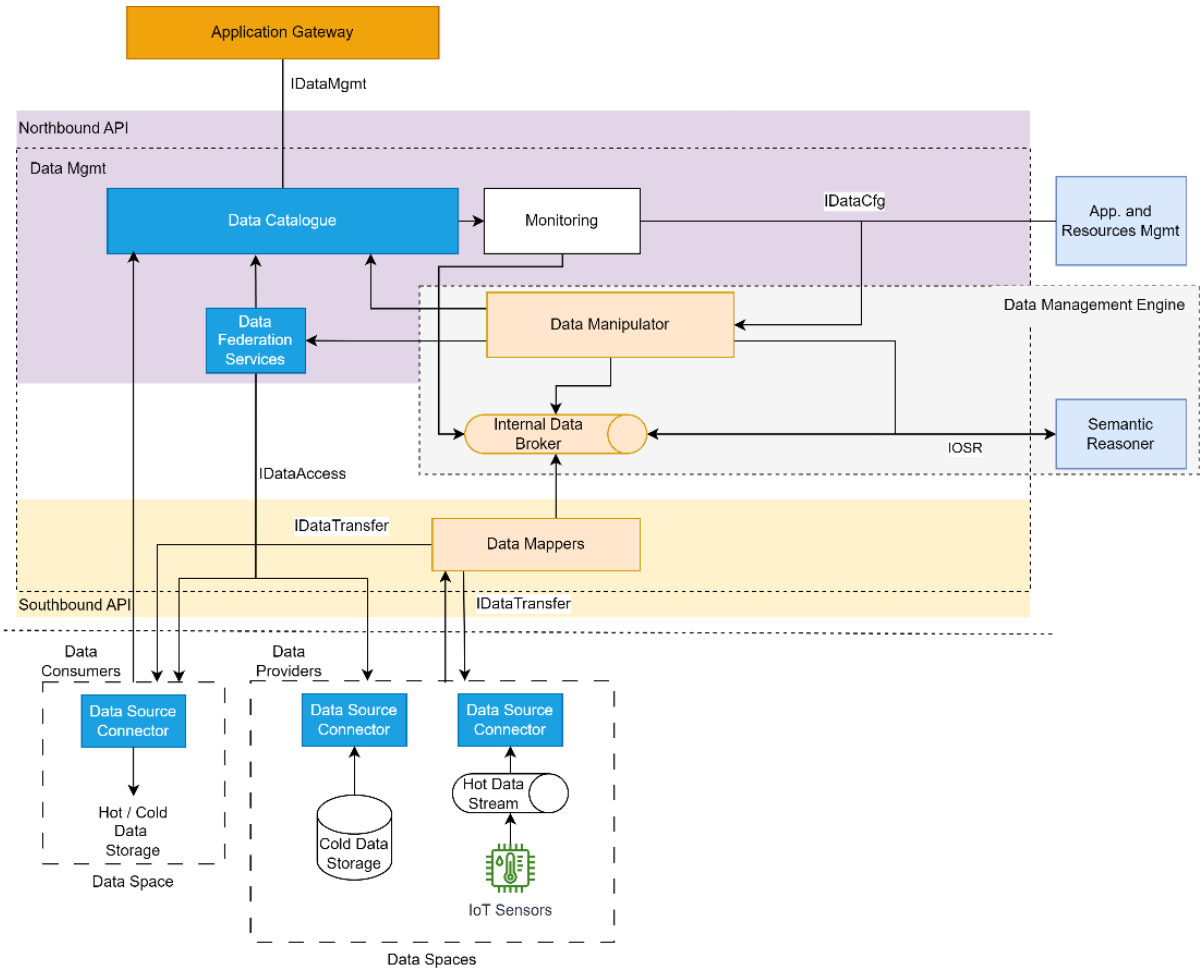

Figure 8. Architectural framework for AC³ data management PaaS

Figure 8 presents the structure of the AC³ Data Management PaaS. The data management PaaS will be organized as a **northbound** API, an **engine**, and a **southbound** API. Both APIs support semantic awareness and ontology-based policy specification for managing customizable applications in a smooth and user-friendly manner. By incorporating all the showcased components and their interactions, the CECCM establishes a robust data management framework. It allows users to efficiently navigate, manipulate, and monitor data through a user-friendly interface. The Northbound API provides comprehensive data cataloguing, monitoring, and federation capabilities. The engine ensures efficient data processing and intelligent decision-making. The southbound API facilitates seamless integration with diverse data spaces.

4.8.1 Northbound API

The northbound API encompasses several essential components: the Data Catalogue, the Monitoring module, and the Data Federation Services.

The *Data Catalogue* provides a comprehensive inventory of available data, including information about formats, types (live or stored), locations, and access methods. It offers all the necessary procedures and interfaces to register, federate, manage, and provide access to data spaces of application developers or third party IoT infrastructure providers.

The *Monitoring* module ensures the real-time monitoring and tracking of data streams, enabling efficient data management. It is responsible for checking the status of the data sources federated with the Data Management PaaS, as well as the usage of the data sources by the applications running in the context of the AC³ CECCM.

The *Data Federation Services* are responsible for managing access and permissions. In this service, data providers will also be able to provide rules for accessing their data and their use as well as conditions under which access would be revoked by the system. This is needed to protect data sovereignty, ensure transparency and fairness, and thus facilitate the generation of economic value for both data providers and data consumers.

4.8.2 Data Management Engine

The data management engine (grey part in Figure 8) is a critical component in the data management architecture. This engine is comprised of the Data Manipulator, the Internal Data Broker, and the Semantic Reasoner components.

The *Data Manipulator* component allows users to manipulate data and perform operations such as transformation, aggregation, and filtering. It can also facilitate the transfer of data to an external data sink as needed. Additionally, the *Data Manipulator*, with the help of the *Monitoring* module, communicates with the "App. and Resources Mgmt" component through the $I_{DataCfg}$ interface regarding the statistics on the operations on the data sources used, and the load occurred due to the processing of the data in the context of the Data Management PaaS. This interface facilitates the exchange of data configuration information, enabling efficient coordination and management of applications and resources within the CECCM.

The *Internal Data Broker* serves as a communication channel within the CECCM, facilitating data flow and exchange between different components. It is a central point that receives data from the southbound API requested by applications executed in the CECCM and delivers them to the appropriate applications (based on the conditions defined by the *Data Federation Services*).

The *Semantic Aware Reasoner* is described in more detail in Section 4.6 and is utilized from the Data Management PaaS to leverage semantic and ontology techniques, interpret and process data-related policies, ensuring compliance and intelligent decision-making in the data management process.

4.8.3 Southbound API

The southbound API comprises the *Data Mappers* and *Data Source Connectors*, which establish communication with the various external data spaces that are integrated with AC^3. AC^3 can handle both hot (i.e., live sensor data streams, video feeds) and cold (i.e., data lakes, databases, object storages) data sources.

The Data Mappers and Data Source Connectors enable seamless integration and interaction with these data sources, providing standardized access and retrieval mechanisms. Both modules are responsible for exposing each Data Space through a common interface and a common format. This common format is very important for AC^3 application developers to have a common way of handling data regardless of their origin.

4.9 Interfaces

We summarize all the interfaces of the AC^3 architecture in Table 2 and Table 3. Table 2 describes all inter-plane interfaces while Table 3 all intra-plane interfaces.

Table 2: Inter-plane interfaces and their functionalities

Interface Name	Involved modules	Sub-Interface(s)	Description
I_{CRUD_A}	Application Developer, Application Gateway	-	Interface allowing the application developer to request CRUD procedures (i.e., deploy, instantiate, monitor, and delete micro-services composing an application
I_{CRUD_U}	Application Gateway,	I_{R_U}	Monitoring Management and measured KPI presentation

	Application & Resources Mgmt.	I_{CUD_U}	Deployment, instantiation, and deletion of micro-services
I_{OSR}	Application Gateway and OSR	-	Provides the OSR with the semantic models and policy rules and the OSR produces ontology instances and validated/verified semantic rules
$I_{DataMgmt}$	Application Gateway, Data Mgmt	-	Provides the Application Gateway with access to the available data sources as well as with access to their management operations
$I_{DataAccess}$	Data Federation Services, Data Source Connector	-	Provides access and permissions management for the available data sources.
$I_{DataTransfer}$	Data Mappers, Data Source Connector	-	Provides access and data transfer operations to the actual data stored in the data spaces connected to the AC^3 platform.
I_{FED}	Application & Resources Mgmt, Adaptation & Federation layer	-	Select an infrastructure provider from the Federation
I_{CRUD_M}	Application & Resources Mgmt, Adaptation & Federation layer	$I_{D\_LCM\_SB}$	Further divided into two sub-interfaces: LD_CFG and LD_SB. The first one is to discover the adaptation agent handling an infrastructure. The second one is to enforce LCM decisions using the agent discovered by LD_CFG
		$I_{Mon\_M\_SB}$	Further divided into two sub-interfaces: Mon_CFG and Mon_SB. The first one is needed to discover the NBI of a LMS to collect monitoring. The second one is to collect monitoring data from the LMS discovered using Mon_CFG.
$I_{DataMgmt\_SB}$	Application & Resources Mgmt, Data infrastructure	$I_{DataAccess}$, $I_{DataTransfer}$	Provides interaction with federated data sources and data transfer operations.
I_{Mgmt_SB}	Adaptation & Federation layer, CECC infrastructure	I_{RES_DSC}	Discover resources and NBI exposed by LMS participating to the federation
		I_{Mgmt_SB}	Enforce LCM decisions using NBI exposed by LMS

Table 3: Intra-plane interfaces and their functionalities

Interface Name	Concerned module	Description

$I_{Mon\_CFG\_M}$	Application & Resources Mgmt.	Configure the monitoring module (KPI to measure) per application
I_{Mon_Data}	Application & Resources Mgmt.	Expose measured KPI by application and used-infrastructure.
$I_{AppProfile}$	Application & Resources Mgmt.	Expose learned Application profile
$I_{ResProfile}$	Application & Resources Mgmt.	Expose resources profile including current resource availability
I_{A_CFG}	Adaptation & Federation layer	Configuration (instantiation) of an agent per infrastructure LMS.
I_{A_Disc}	Adaptation & Federation layer	Discover NBI exposed by LMS for monitoring (R) and CUD.
I_{FED_Inf}	Adaptation & Federation layer	Discover LMS exposed resources.

5 Sequence Diagrams

Based on the architecture unveiled in section 4, we describe the initials among the components included in the architecture regarding functionalities offered by the CECCM of AC^3.

5.1 Application related deployment

5.1.1 Application Deployment

We first start with the application deployment workflow (see Figure 9. Service Creation workflow with focus on the App & Resources Mgmt component), which is considered a user-driven workflow.

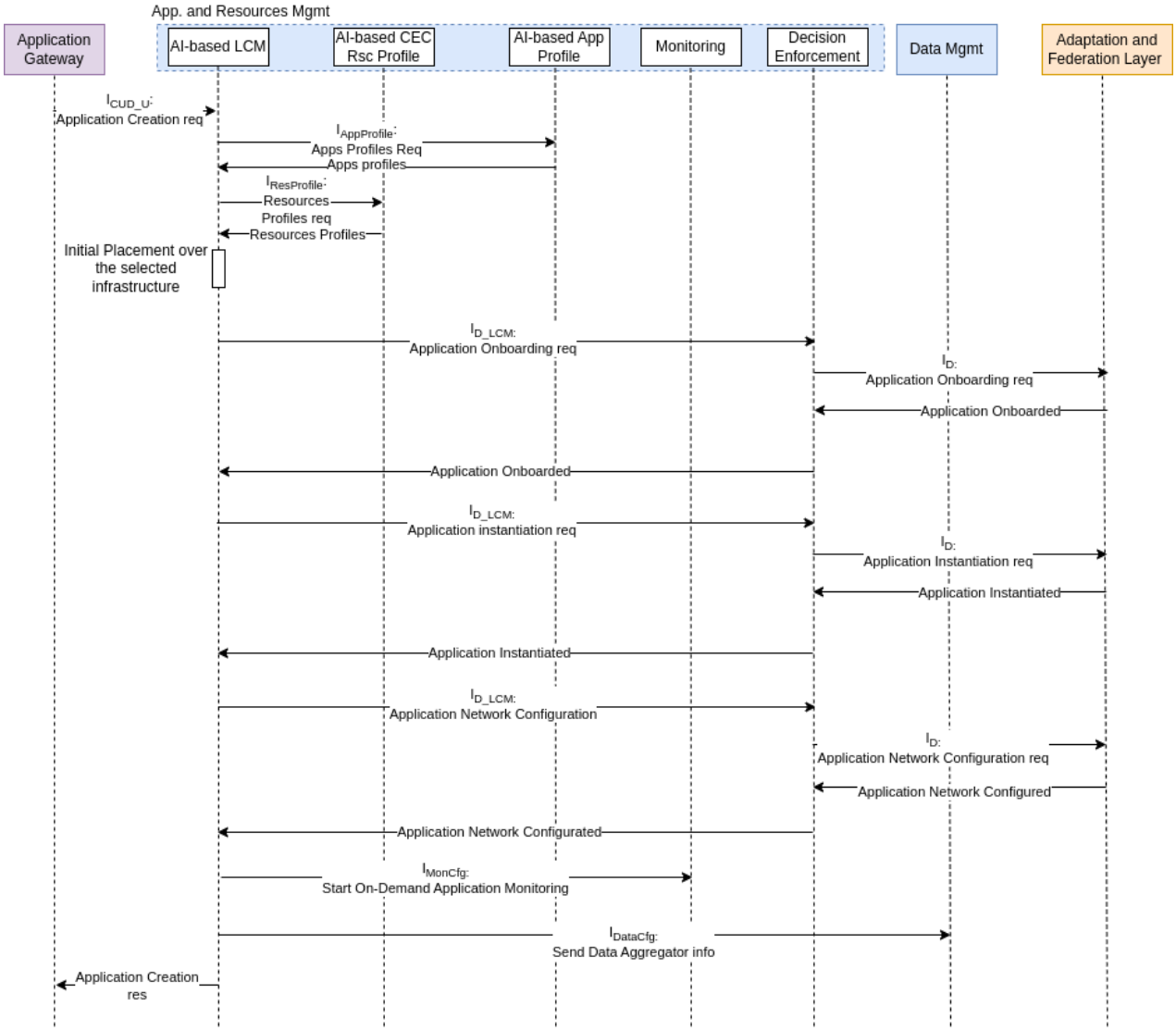

Figure 9. Service Creation workflow with focus on the App & Resources Mgmt component

The process of creating an application begins with the application gateway, through which the application developer requests the creation of the application using the application descriptor.

The process of creating an application begins with the application gateway, through which the application developer requests the creation of the application using the application descriptor.

The Application Gateway then forwards the Application Creation request to the App and Resources Mgmt where the AI-based LCM receives the request. The latter requests the application profile from the AI-based App Profile and the resources profile from the AI-based CEC Rsc Profile. Based on the two pieces of information, the AI-Based LCM decides the initial placement and resources of the application over a target infrastructure.

Once the decision is made, the AI-Based LCM starts the application deployment following 3 steps: Onboarding each microservice of the application by loading the container image of the microservice. Instantiating the application by starting the application with the allocated resources in the selected infrastructure and configuring the application network by creating the respective traffic rules to expose the application for RU (Read and Update) operations.

For each step, the AI-based LCM sends a request to the Decision Enforcement components, that in turn sends the request to the Adaptation Agent at the Adaptation and Federation Layer as shown in Figure 10. Then, the Adaptation Agent forwards the request to the infrastructure specific Local Management System.

Once the application is deployed an application monitoring request is sent from the AI-Based LCM to the Monitoring component to start the monitoring of the application's microservices and networking. In the case where the application needs to consume data from IoT devices, Data aggregator information is sent to the Data Management module to configure the appropriate data publishers.

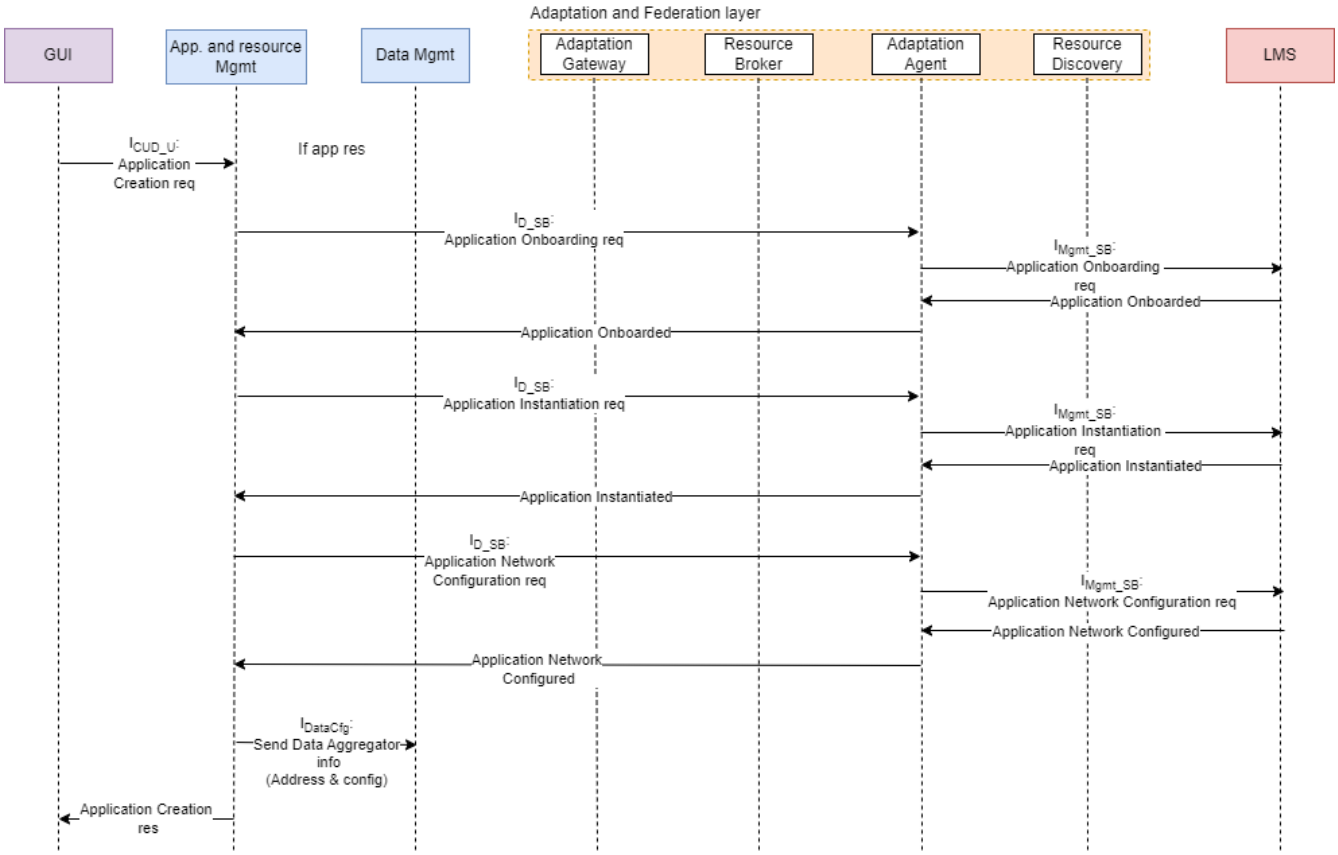

Figure 10. Service Creation workflow with focus on the Adaptation & Federation layer

Finally, once the application is created the AI-Based LCM returns information about the application such as the endpoints of its services and the monitoring information source for the application.

5.1.2 Adaptation Agent creation

When an application needs to be deployed in a region for which the Adaptation Agent has not yet been deployed, the Decision Enforcement module, from the App & Resources Mgmt Component, sends a request to configure an adaptation agent to the Adaptation Gateway. The latter then gets information about the NBI endpoint of the LMS present at the desired infrastructure, and using this information it creates and configures an Adaptation Agent. Afterwards the endpoint of the Adaptation agent is returned to the Decision Enforcement module (see Figure 11).

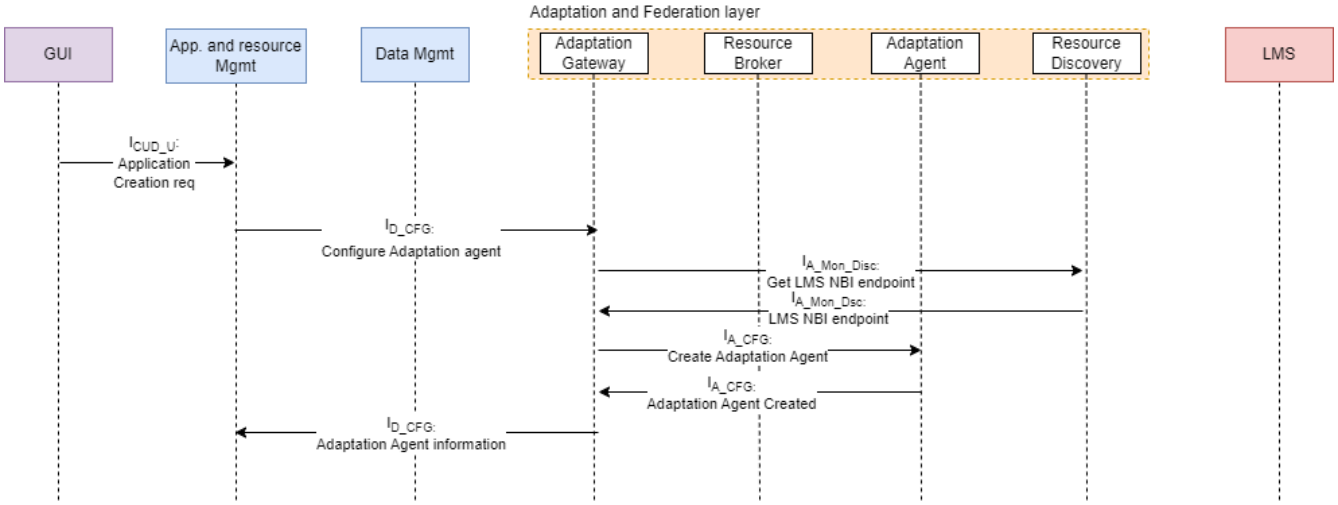

Figure 11. Adaptation Agent Creation workflow

5.2 Data Space Registration

Regarding the registration of new data spaces to the AC^3 Data Management PaaS we showcase in the following workflow all the needed steps from the initial request to the monitoring notifications sent back to the AC^3 user.

The "Add Data Space" sequence diagram (Figure 12) presents a comprehensive overview of the data space registration process in the Data Management module.

The flow begins with the Actor initiating the request to "Add Data Space" through the Application Gateway. Subsequently, the Application Gateway communicates with the Data Catalogue to register the new data space. The Data Federation Services are then engaged to check the availability of the data space, ensuring its readiness for use.

Once verified, the Data Mappers validate the data space connection, confirming its accessibility. The Data Federation Services and Data Catalogue coordinate to finalize the confirmation of data space availability, and the confirmation is relayed back to the Actor.

Additionally, the Monitoring component plays a crucial role, requesting status updates from the Data Catalogue and acquiring access permissions from the Data Federation Services. This seamless coordination of interactions ensures a smooth data space registration process.

To maintain the data space's operational integrity, the Monitoring module continuously checks for availability at predefined intervals. During these checks, the Data Mappers verify the data space connection with the Data Source Connector, and the availability report is shared with the Monitoring module. In the event of an offline data space, the Monitoring module promptly notifies the Data Catalogue, which subsequently communicates the data space's offline status to the Actor.

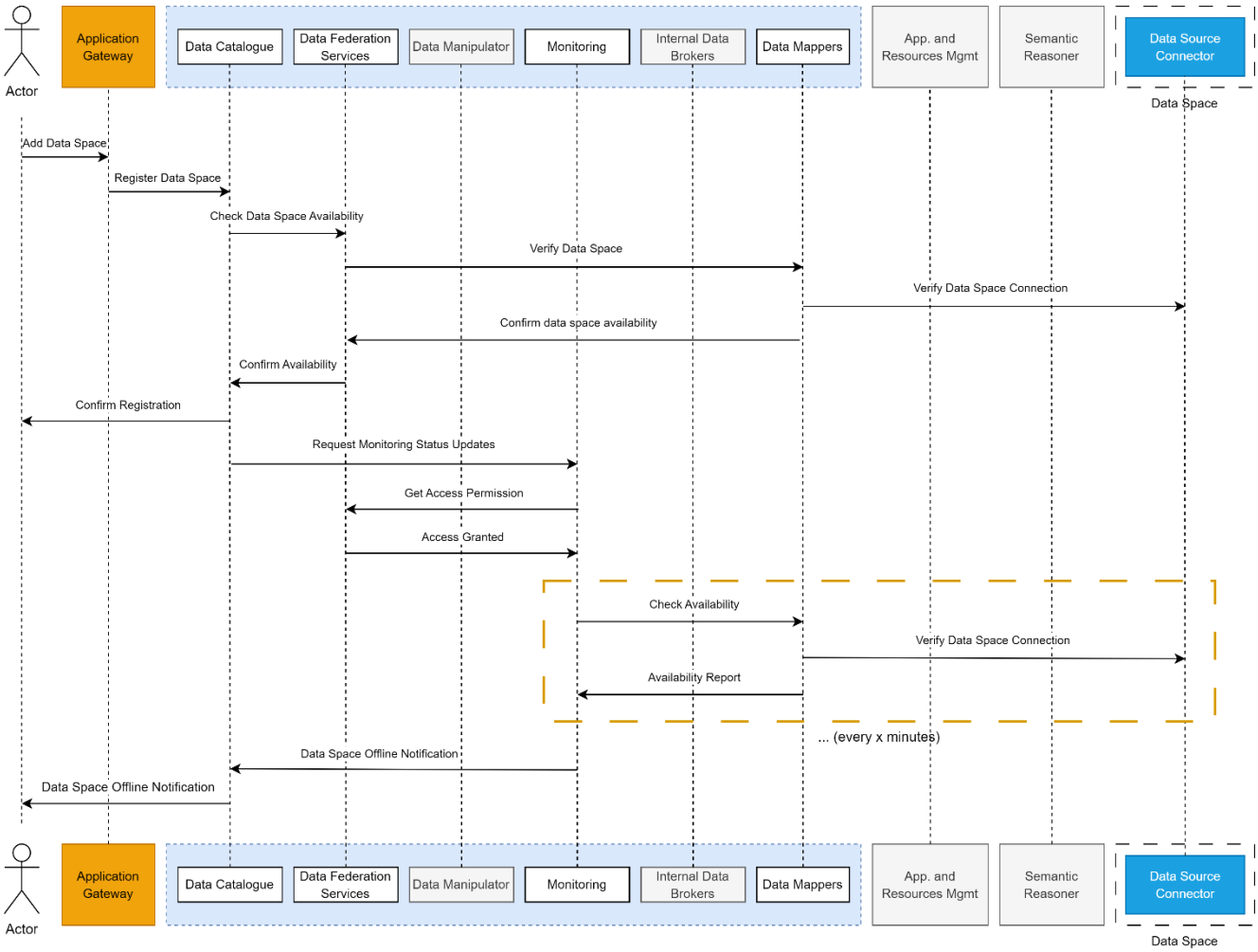

Figure 12. Add Data Space Sequence Diagram

5.3 Cold Data Retrieval

The "Cold Data Retrieval" sequence diagram (Figure 13) offers a comprehensive representation of the data retrieval process in the AC^3 Data Management PaaS.

The flow commences with the Application Gateway deploying an application that needs to retrieve some cold data from one of the Data Spaces associated with AC^3, enabling seamless access to data. The App. and Resources Mgmt module then communicates the request for "Cold Data Retrieval" to the Data Manipulator, which proceeds to retrieve vital information and data space availability from the Data Catalogue.

Upon successful confirmation, the Data Manipulator gains access permission from the Data Federation Services. With granted access, the Data Manipulator directs the Data Mappers to retrieve the required cold data using the Data Source Connector.

The fetched data are forwarded to the Internal Data Broker of the PaaS to be distributed to the Data Manipulator. Subsequently, the Internal Data Brokers deliver the cold data to the Data Manipulator, allowing further reasoning on the retrieved data by the Semantic Reasoner. To ensure effective monitoring, the Data Manipulator shares data monitoring information with the Monitoring module. Finally, the App. and Resources Mgmt module receives the computation result, further enhancing the project's data retrieval capabilities.

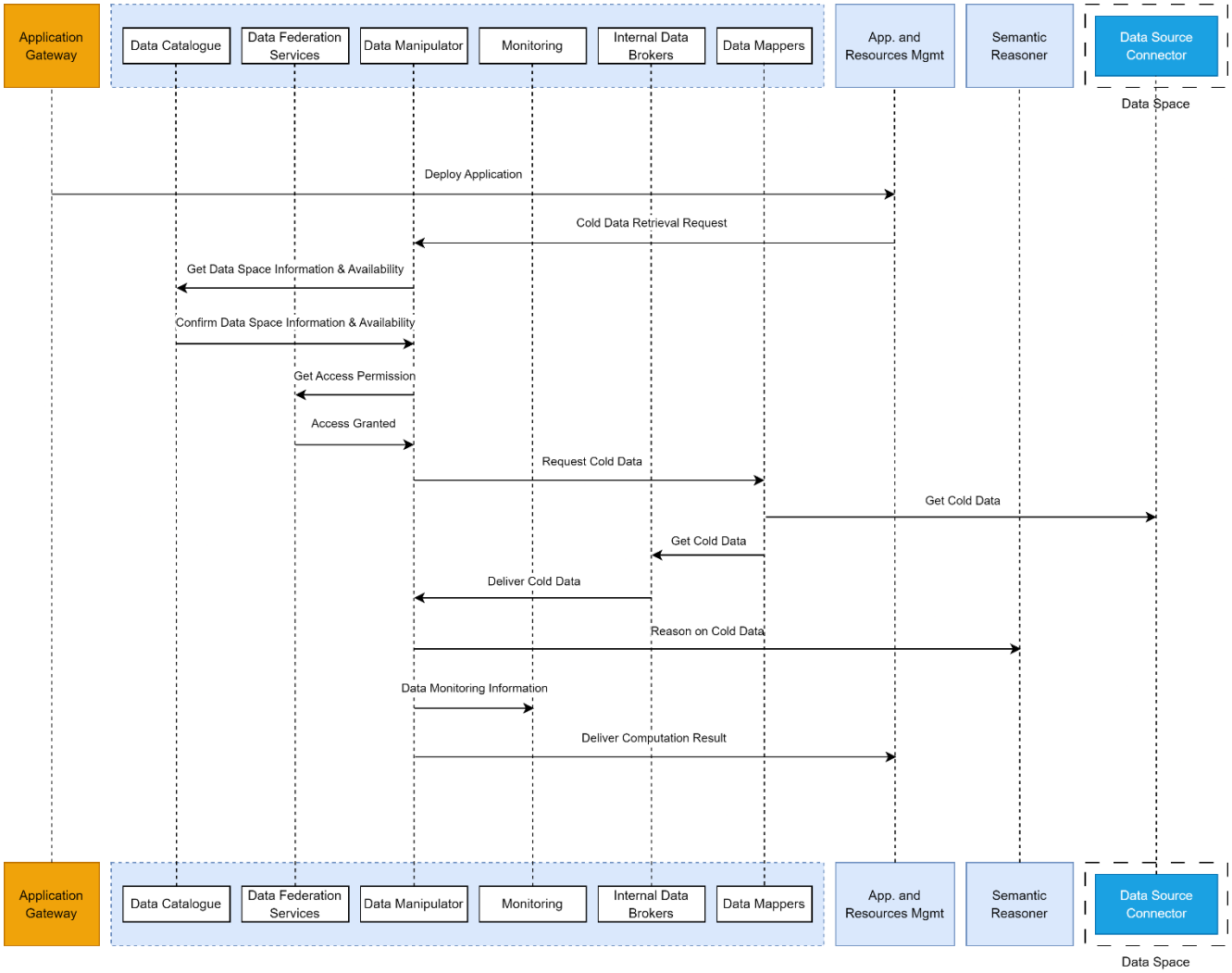

Figure 13. Cold Data Retrieval Sequence Diagram

5.4 Hot Data Retrieval

The "Hot Data Retrieval" sequence diagram (Figure 14) provides a comprehensive depiction of the dynamic data retrieval process within the AC^3 Data Management PaaS.

The flow commences with the Application Gateway deploying the application, facilitating smooth access to data. Upon user request, the App. and Resources Mgmt module communicates the "Hot Data Retrieval" request to the Data Manipulator.

Subsequently, the Data Manipulator retrieves essential data space information and ensures its availability by communicating with the Data Catalogue. Once confirmed, the Data Manipulator gains access permission from the Data Federation Services. The Data Mappers are then directed to subscribe the AC^3 Data Management PaaS to the hot data from the Data Source Connector.

As the hot data is updated, the Data Source Connector promptly relays the updates to the Data Mappers. The Internal Data Brokers facilitate the process by efficiently distributing the hot data to the Data Manipulator. Further, the Semantic Reasoner plays a pivotal role in reasoning on the retrieved hot data.

To maintain real-time monitoring, the Data Manipulator shares data monitoring information with the Monitoring module, ensuring seamless operations. Finally, the App. and Resources Mgmt module receives the computation result, enhancing the project's data retrieval capabilities.

In case of the need for cancellation, the "Hot Data Retrieval" process is adeptly modified, efficiently revoking access permissions and unsubscribing from the hot data source. With these crucial steps repeated until cancellation, the "Hot Data Retrieval" process showcases the module's versatility, reinforcing the project's ability to effectively handle dynamic data retrieval needs.

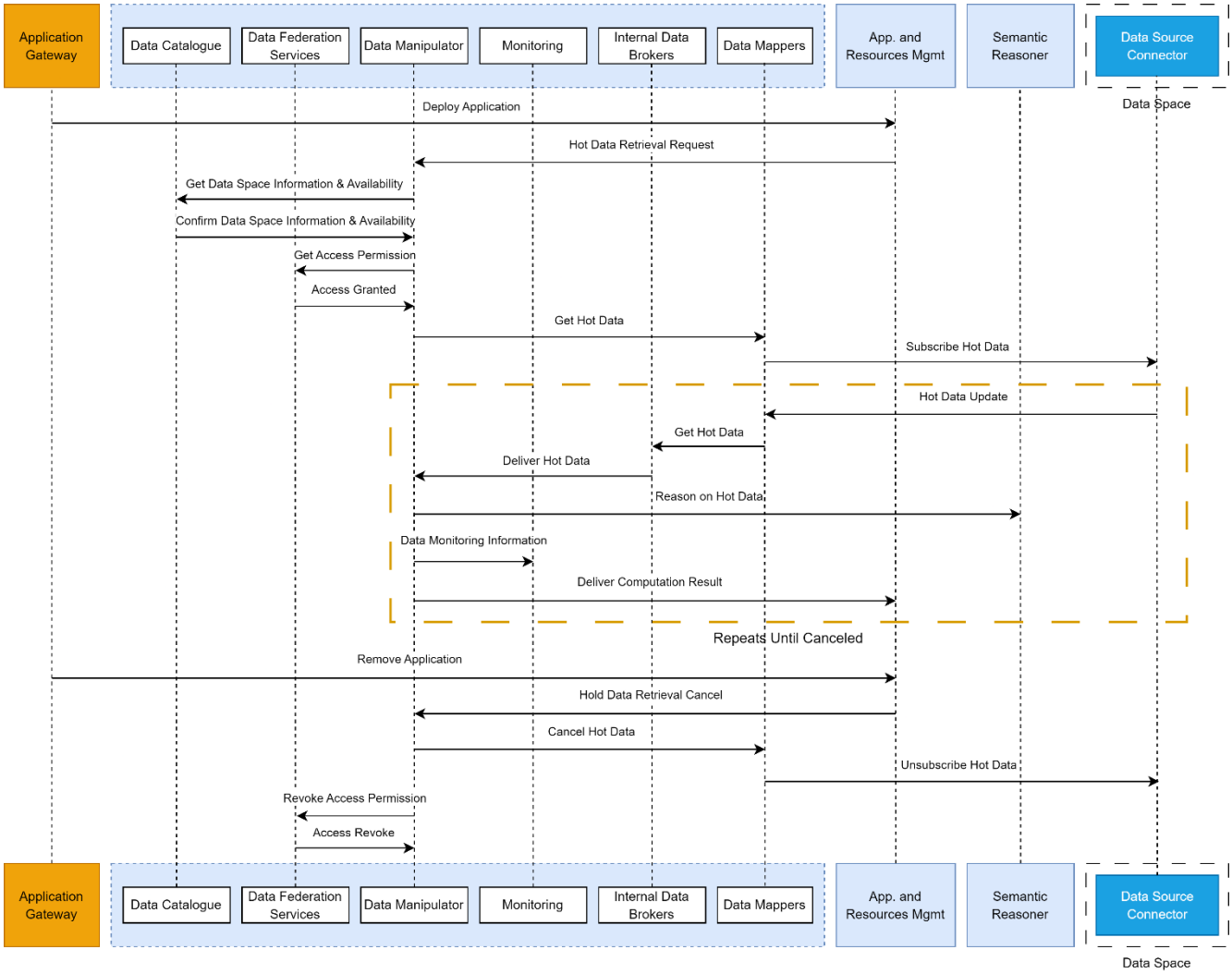

Figure 14. Hot Data Retrieval Sequence Diagram

6 Conclusion

This document amalgamates the outcomes of tasks T2.1 (partially) and T2.2, serving as the foundational baseline for the architecture. This foundation is not static; it will be further refined and enhanced based on insights from WP2 activities as well as feedback garnered from WP3 and WP4 endeavours. In this deliverable, the initial rendition of the CECC functional architecture is unveiled. It provides an exhaustive description of the AI/ML-based LCM and resource management block, encapsulating all other functional blocks such as the Application Gateway, OSR, and the like. Additionally, the intricate inter-plane and intra-plane interactions of these diverse functional blocks are meticulously identified and documented.

Taking cues from the NIST cloud federation and aligning with the Gaia-X/IDSA innovations and specifications, this document introduces the inaugural version of the data-infrastructure federation. Complementing the textual descriptions, four sequence diagrams are integrated into this deliverable, illustrating the dynamic interactions amongst various components and stakeholders or participants within the AC^3 framework.

View this document as an evolving blueprint that sets the stage for the forthcoming activities in WP3 and WP4. As the project progresses, specifically by M24, the final version of the CECC framework, along with the CECCM, will be unveiled. This refined version will seamlessly integrate the federated and open API, enhance the functional blocks of the CECCM, and incorporate invaluable feedback sourced from WP3 and WP4.

7 References

[1] Bohn, R., Lee, C. and Michel, M. (2020), The NIST Cloud Federation Reference Architecture, Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, [online], https://doi.org/10.6028/NIST.SP.500-332 (Accessed August, 2023).

[2] IEEE Standard for Intercloud Interoperability and Federation (SIIF), IEEE 2302-2021, C/CCSC - Cloud Computing Standards Committee, 2022-03-08.

[3] Miller D., Whitlock J., Gardiner M., Ralphson M., Ratovsky R., Sarid U, "OpenAPI specification v3.1.0", (latest editor's draft, 2021). https://github.com/OAI/OpenAPI-Specification