D10.1 Report on EOSC Integration Suite and Execution Framework Requirements and Gap Analysis v.1

31/10/2024

Abstract

EOSC Beyond advances the state of the art of the European Open Science Cloud by improving its Core functionalities as well as extending with new ones. Among the new ones, the Integration Suite and the Execution Framework are discussed in this document. This document reports on the identified gaps and requirements on the Integration Suite and Execution Framework commissioned by the project during its early phase.

Funded by the European Union

EOSC Beyond receives funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101131875

Document Description

Report on EOSC Integration Suite and Execution Framework Requirements and Gap Analysis v.1_Final			
Work Package 10			
Due date	31/10/2024	Actual delivery date:	22/11/2024
Nature of document	Deliverable	Version	1.0
Dissemination level	Public		
Lead Partner	EGI Foundation		
Authors	Carlos Brandt (EGI), Levente Farkas (EGI), Enol Fernandez (EGI), German Molto (UPV), Miguel Caballer (UPV), Athanasios Mantes(AARC), Kostas Androutsopoulos (OpenAIRE), Paolo Manghi (OpenAIRE), Michal Kolomanski (Cyfronet), Pavel Weber (KIT), Tibor Kalman (GWDG), Kostas Koumantaros (GRNET), Themis Zamani (GRNET), Zdenek Sustr (CESNET), Viet Tran (IISAS), Bjorn Hagemeier (FZJ)		
Reviewers	Diego Scardaci (EGI), Roksana Wilk (Cyfronet)		
Public link	https://zenodo.org/records/14204589		

Description

Abstract	EOSC Beyond advances the state of the art of the European Open Science Cloud by improving its Core functionalities as well as extending with new ones. Among the new ones, the Integration Suite and the Execution Framework are discussed in this document. This document reports on the identified gaps and requirements on the Integration Suite and Execution Framework commissioned by the project during its early phase.
Keywords	EOSC, EOSC Federation, Interoperability Framework, IF Guidelines, Adapters, Integration Suite, Execution Framework

Revision History

Issue	Item	Comments	Author/Reviewer
V 0.1	Draft version	Content structure and key points of discussion	Levente Farkas, Diego Scardaci, Carlos Brandt

V 0.2	Draft version	Execution Framework gap analysis and requirements	Germán Moltó, Miguel Caballer, Athanassis Mantes, Kostas Androutsopoulos, Paolo Manghi
V 0.3	Draft version	Integration Suite adapters, gap analysis and requirements	Carlos Brandt, Levente Farkas, Pavel Weber, Athanasios Mantes, Roksana Wilk, Zdenek Sustr
V0.9	Reviewers version	Fixes according to internal reviewers comments	Carlos Brandt, Levente Farkas
V 1.0	Submitted version	Approved by reviewers	TCB

Copyright and licence info

Table of Content

Executive Summary

The European Open Science Cloud (EOSC) is a federation of research initiatives aimed at enhancing the discovery, access, and reuse of Open Science resources. EOSC Beyond expands EOSC in terms of integrated providers and active users by offering new core technical capabilities for scientific applications. Key advancements include the creation of two new Core services: the *Integration Suite*, a portfolio of software adapters for the EOSC Core and the EOSC Exchange services, and the *Execution Framework*, a system for the composability and deployment of multi-supply resources. These components collectively aim to streamline the development, integration, and adoption of scientific tools and services, thereby accelerating scientific discoveries and fostering collaboration across research communities.

The Integration Suite shortens the time for the creation of data-driven tools by providing a collection of software adapters. It includes software libraries that integrate all types of EOSC resources: applications/tools, services, research data, and etc. This open and extensible environment allows the sharing of libraries across research communities, fostering interdisciplinarity and enhancing interoperability. The suite features adapter libraries as implementation of the EOSC IF Guidelines.

The Execution Framework automates the composition and deployment of EOSC resources, reducing the human effort required for creating scientific workflows. Utilising the EOSC Interoperability Framework (EOSC IF), TOSCA (Topology and Open Specification for Cloud Applications) templates, and DevOps tools, it supports automated composition, provisioning and configuration of services, research products and virtualized infrastructure. An enhanced IF Registry enables interoperability-driven discovery, supporting both human-readable and machine-composable guidelines.

Together, the Integration Suite and Execution Framework significantly improve the provision, automation and interoperability of scientific resources within EOSC. By simplifying resource composition and deployment, they accelerate scientific discovery and foster collaboration, contributing to the success and sustainability of the EOSC initiative.

1. Introduction

EOSC Beyond is evolving the current European Open Science Cloud (EOSC) by advancing existing EOSC Core services as well as adding new ones with the aim to facilitate the set up of EOSC as a Federation of Nodes and support its growth both in terms of integrated providers and active users. The goal is to provide new core technical solutions that allow for easy composition of scientific data and analysis environments. EOSC Beyond target users (stakeholders) are system administrators, software developers, and data engineers, including researchers with expertise in those areas.

Among the new core services EOSC Beyond is developing, we find the Integration Suite (IS) and the Execution Framework (EF). The Integration Suite will provide tools to ease the use and integration of services within the Federation. While the Execution Framework will provide support for services composition and deployment automation. Both services — IS and EF — are descendants of the EOSC Interoperability Framework [1], designed and developed in the previous EOSC Future project[1]. The Interoperability Framework (IF) is responsible for managing EOSC Interoperability Framework Guidelines[2] (IGs), which are documents that guide technical users on how to integrate (or develop interoperable) services – a fundamental concept that will be extended by IS and EF to achieve their objectives.

In particular, both IS and EF will exploit the capabilities of the Interoperability Framework Registry (*aka*, IF Registry) – the IF central software responsible for hosting the IGs –, which will be extended by EOSC Beyond to provide machine-useful material next to IGs.

1.1. Scope and purpose of the document

This document reports the current state of the design of Integration Suite and Execution Framework as of month 6 of activities in EOSC Beyond with a focus on the requirements gathered by the EOSC Stakeholders and on the result of a gap analysis. We have identified EOSC services functionality gaps, mapped into requirements and defined the initial design of the two services. This deliverable will be used by WP11 and WP13 to start their work, translating the requirements and the design into appropriate development activities.

1.2. Structure of the document

The document is structured as follows:

- **Section 2**: introduces the EOSC Integration Suite (IS), drawing on the need through a **gap analysis**, its software **requirements**, and the implementation **design**;
- **Section 3**: introduces the EOSC Execution Framework (EF). We discuss the **gaps** EF is expected to fulfil, the definition of functional and non-functional **requirements**, and the initial **design**;
- **Section 4**: includes concluding remarks.

[1] https://eoscfuture.eu/
[2] https://open-science-cloud.ec.europa.eu/resources/interoperability

2. Integration Suite

The EOSC Integration Suite (IS) is an open and extensible environment of software libraries – called *Adapters* – to integrate tools, services, and research data for the creation of composed services or data-driven applications. The goal is to shorten the time-to-product of new tools and services.

The IS will be made up of documents, software, and repositories, structured to deliver Adapters (and related Guidelines) for EOSC services through the EOSC Marketplace and the Interoperability Framework Registry.

To create the IS, the Interoperability Framework Registry will be extended to support the association of Adapter(s) to the Interoperability Framework Guidelines. The EOSC Interoperability Framework Guidelines[3] (IGs) are documents specifying the interfaces to communicate with a compatible service. They define how to exchange data with this service (*e.g.* a client API), data standards to be adopted (*e.g.*, file format, metadata structure), and/or a communication protocol (*e.g.*, HTTPS). Adapters will be reference implementations of IGs that will allow providers of a third-party service to connect to services compliant with the IGs with reduced development cost.

IGs also detail the interfaces of the EOSC Federating Capabilities[4] and, therefore, adapters for these IGs will facilitate the integration of an EOSC Node service with the EOSC federating services. Then, adapters can be also seen as helpers offered by EOSC to Node operators and service providers to speed up their enrollment/onboarding in the EOSC Federation. However, it is important noticing that the use of Adapters is not mandatory, EOSC Node operators and service providers can also use technology of choice to connect to the EOSC federating capabilities.

The Integration Suite is designed to be an extensible environment. It is expected to grow with contributions from the EOSC community to increase the number of supported use cases. It will contain IGs and Adapters from all sorts of services offered by the EOSC Federation, beyond the core and horizontal services delivered by EOSC Beyond.

The EOSC IF Registry will store references to the IGs and Adapters of the IS and will make them available to the user through the EOSC Marketplace[5]. The Marketplace will allow users to find services and their related IGs and Adapters. It will also allow users to locate services that implement specific IGs, that have Adapters for a specific programming language or platform, or a combination of these filters.

2.1. Gap Analysis to support Integration Suite

A key challenge in integrating services and data sources into EOSC has been the lack of standardised, off-the-shelf solutions for interoperability. Currently, integration efforts depend

[3] https://zenodo.org/records/8399710
[4] https://zenodo.org/records/13939396
[5] https://marketplace.sandbox.eosc-beyond.eu

heavily on custom development, with new providers required to build their own software to interact with the federation services.

While the EOSC IF Guidelines[6] provide the necessary directions for developers to implement the software for services integration, the lack of ready-to-use solutions creates barriers for new providers joining EOSC. Without pre-configured adapters, services may experience inconsistent integration quality and increased time and complexity required for successful onboarding in EOSC.

The Integration Suite addresses this gap by offering pre-built, extensible software libraries – Adapters (REQ.IS.1)– that align with the existing EOSC IF IGs. These Adapters will simplify and accelerate the integration process by reducing the need for custom development. They provide a flexible framework, enabling service providers to integrate their services more efficiently while covering a broad range of technical and use-case requirements. For a breakdown of the identified gaps in existing EOSC Core services that must be addressed to facilitate discovery and use of EOSC services, see section Interoperability Framework Registry and section Marketplace.

An example of such adapters can be a Python library that interfaces a REST API of a service – such an adapter would bridge two different technologies and expand the use of the original service's interface to a whole new set of (Python) developers. Other examples are the transformation of data from a metadata standard to another or from a file format to another (e.g. from JSON - typically used in web-servers communication - to CSV tables - for direct data analysis).

Adapters could also help with the use of the EOSC services where the barrier to adopt them is not a service's API, but instead the complex exchange of messages and/or data between the EOSC core service and the connected user or service, which sometimes require translating or altering the messages flowing between the parties. In such cases, an Adapter could be more than a wrapper of the service's API in the preferred programming language of the EOSC service's consumer, but instead could alter the flow of messages/data between the two. An example is an adapter that translates the EOSC Profiles metadata in metadata standards widely adopted by research communities (e.g. DCAT) and vice versa.

IMPORTANT **Adapters are optional additions to their peer EOSC service**. In other words, the full set of capabilities of the EOSC services must remain available and accessible without requiring the use of any of the available Adapters.

Each Adapter should be associated with an IG to ensure that it is a compliant extension of some EOSC's interoperability rules (REQ.IS.2). Analogously, to align with EOSC open science ideals, ease adoption, extensibility, and community growth, Adapters should be open-source software (REQ.IS.3).

The EOSC Beyond project, through WP11 and WP12, will provide an initial set of Adapters for EOSC Core services and some strategic horizontal services. And from there, IS adapters are expected to grow through community contributions (REQ.DP.1 and REQ.DP.2).

[6] https://open-science-cloud.ec.europa.eu/resources/interoperability

Pilot nodes such as CESSDA and NI4OS [2] have demonstrated the complexity of integrating services using APIs, often requiring considerable time and effort. The new adapters promise a much faster and more streamlined process, adaptable to various integration needs. This will allow future service providers to integrate with the EOSC Core services more efficiently and in a standardised manner.

Although Adapters are expected to reduce time and simplify integration and use of services, their use may not be desirable but some participants of the federation, and so adapters adoption must be optional (REQ.IS.4).

2.1.1 Interoperability Framework Registry

The EOSC Interoperability Framework (IF) Registry provides tools to describe the interoperability capabilities of resources through IF Guidelines. It enables the annotation of EOSC resources across various research and infrastructure domains, indicating which guidelines each resource follows. This creates an interoperability layer over EOSC resources, allowing users to discover and assess resources based on their interoperability characteristics across different disciplines.

Currently, the EOSC IF Registry contains a collection of guidelines that serve as human-readable instructions for EOSC Nodes and Providers[7]. While these guidelines are crucial for defining functionalities and ensuring interoperability between EOSC services and research products, they are not machine-readable. As a result, service providers must manually design and configure their systems to align with technical interoperability requirements, and software developers must convert these guidelines into code to enable actual integration. This manual interpretation adds complexity, increasing the time and effort needed to achieve seamless interoperability.

To address this gap, the EOSC IF Registry should support the registration of software adapters linked to the Interoperability Framework Guidelines (REQ.IF.1). These adapters would allow service providers to automate parts of the integration process, translating guidelines into reusable, pre-built software solutions. By associating each guideline with one or more adapters (REQ.IF.2), EOSC nodes and service providers could easily implement technical requirements without having to develop custom solutions from scratch. This would significantly reduce the effort required for service integration and make interoperability more accessible across the EOSC ecosystem.

In addition, service providers should be able to offer optional software tools – the ‘adapters’ – to facilitate integration with their services. These adapters will be associated with the service’s profile in a one-to-many relationship (REQ.IF.3), where multiple adapters can support a single service. Each adapter will also reference the corresponding Interoperability Guideline in the EOSC IF Registry.

[7] See the EOSC IF Registry currently operated by the EOSC EU Node at https://open-science-cloud.ec.europa.eu/resources/interoperability

2.1.2 Marketplace

The EOSC Marketplace that resulted from the EOSC Future project currently lacks the functionality to search, filter, and visualise service and adapter (REQ.M.1) and guideline relationships being designed by the Integration Suite, which are key to streamline the discovery and utilisation of interoperable resources across the EOSC ecosystem.

By enhancing the Marketplace's search filters, users will gain a clear view of pre-configured services and adapters to follow EOSC's Interoperability Framework Guidelines. This functionality will be designed to simplify the discovery of services aligned with IGs, making adapters available as searchable, discoverable resources in the Marketplace and linking them directly with related services and IGs (REQ.M.2).

To achieve this, the Marketplace must expand its integration with the IF Registry, ensuring that adapters – representing practical implementations of IF Guidelines – are published and easily accessible to end-users. This will streamline the discovery and utilisation of interoperable services and resources across the EOSC ecosystem, providing users with a more intuitive way to find services that comply with IF Guidelines (REQ.M.3).

These developments are essential for aligning the Marketplace with the broader goals of the EOSC Integration Suite, ensuring that it supports the seamless integration and discovery of services and resources configured to EOSC guidelines and standards.

2.1.3 Provider Dashboard

The Provider Dashboard allows service owners to onboard and manage their service and/or associated Guidelines. The same interface must also allow onboarding Adapters (REQ.PD.1).

Another important aspect when related to the relationships between a guideline and one or more associated adapters is that adapters are independent entities from their associated guidelines, which could be supplied by different providers. Therefore, the Provider Dashboard must allow third-parties to contribute (onboard) an Adapter for an existing Guideline, and then subsequently manage it independently of the associated Guideline (REQ.PD.2).

2.2. Requirements to support Integration Suite

Table 1 below captures the requirements for the Integration Suite. This includes software improvements to the EOSC Interoperability Framework, the Provider Dashboard, and EOSC Marketplace, as well as requirements associated with the IS Adapters.

Table 1 - Requirements for Integration Suite

Service	ID	Description
Integration Suite	**REQ.IS.1**	Define Adapters as a new EOSC resource. Description, provider/supplier, licence type, and one or more links to artefact(s)s are needed for an Adapter.

Service	ID	Description
	REQ.IS.2	Each Adapter is associated with exactly one Guideline, ensuring each adapter is a compliant extension of EOSC Interoperability Framework. Guidelines can be associated with more than one Adapter.
	REQ.IS.3	All supported licence types for Adapters must be open-source.
	REQ.IS.4	Adapters must be optional, all features of the service they are associated with must be usable without Adapters. This allows flexibility for providers who may require unique integrations or wish to build custom adapters.
Interoperability Framework Registry	**REQ.IF.1**	Support Adapters.
	REQ.IF.2	Link Adapters and Guidelines. A Guideline can have multiple Adapters (1-to-N).
	REQ.IF.3	Link Guidelines to Services. A Service can have multiple Guidelines (1-to-N).
Marketplace	**REQ.M.1**	Support Adapters (search, filter, and visualise).
	REQ.M.2	Update search filters and queries to support the service-guidelines-adapters relationships, e.g. find services by supported Guideline(s) AND available Adapters(s).
	REQ.M.3	Update results visualisation to show the service-guidelines-adapters relationships.
Provider Dashboard	**REQ.PD.1**	Update the Provider Dashboard and the service/guideline onboarding process to allow adding one or more Adapter(s) for a Guideline.
	REQ.PD.2	Adapters must be independent entities that could have distinct owners from Service and/or Guideline owners and must be manageable independently from the Guideline they are connected to. This allows third-parties (e.g. communities to contribute the adapters they developed to use a particular service).

2.3. Integration Suite Service Design

The EOSC Integration Suite is designed to streamline the integration of tools, services, and research data across the EOSC ecosystem by providing modular, reusable Adapters that align with the Interoperability Framework (IF) Guidelines. The IS is made up of a collection of resources, guidelines, and software components managed by the IF Registry, and made available to the users through the Marketplace.

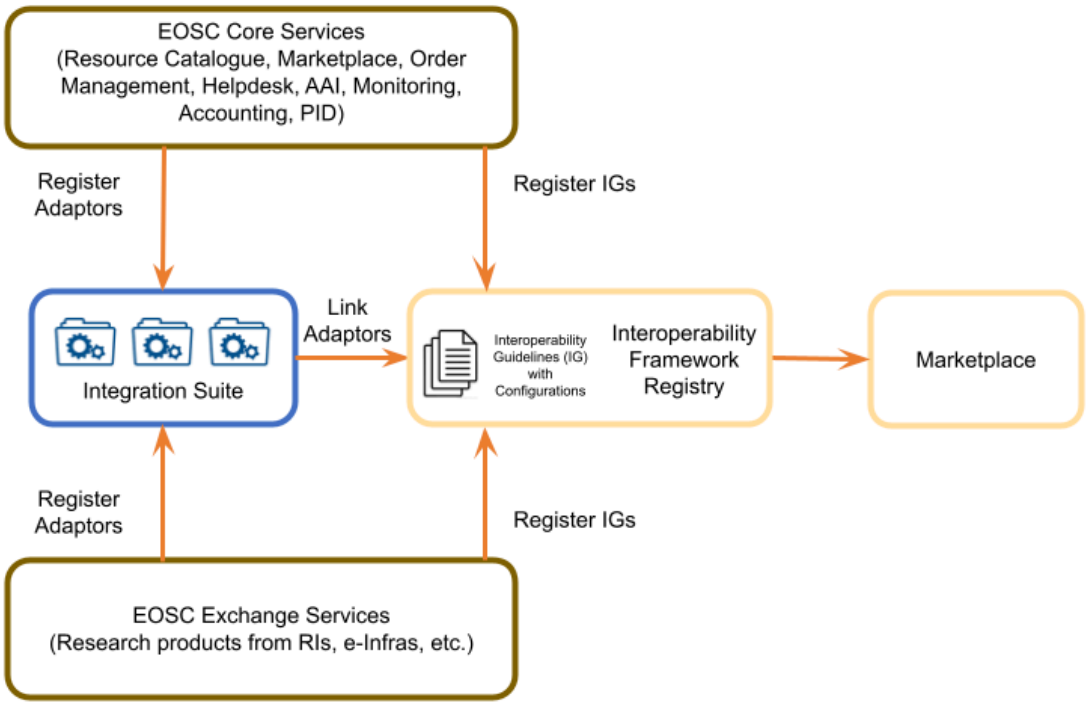

Figure 1 - EOSC Integration Suite architecture

2.3.1 Components

- **Adapters:** modular, pre-configured integration tools that implement the IF Guidelines. Adapters function in a one-to-many relationship with services and guidelines, allowing a single service to utilise multiple adapters and guidelines to be implemented by multiple adapters.
- **IF Registry:** an improved version of EOSC IF Registry that links Guidelines with corresponding adapters and services. The Registry is a central repository for Guidelines and will be extended to host the corresponding Adapters.
- **Marketplace:** an improved version of EOSC Marketplace to support end-users search and discovery of related Adapters, Guidelines, or Services.

2.3.2 Services Workflow and Interactions

1. **Adapter registration:** service providers register adapters in the Integration Suite.
2. **IF registry:** the IS links a registered Adapter to a specific Guideline in the IF Registry – this ensures that each adapter is associated with a guideline.

3. **Marketplace integration:** IF Registry exposes relationship of services, guidelines, adapters, enabling users to discover compliant components.

2.3.3 Key Considerations

- Modularity and Flexibility: the system is designed to handle multiple adapters and guidelines, regardless of the service type or use case.
- Seamless integration: by linking adapters with IGs, the system ensures smooth integration and compliance across the EOSC ecosystem.
- User-Centric discovery: the Marketplace provides a unique and intuitive interface for users to access interoperable services as well as adapters.

2.4. Proposed Adapters for EOSC services

This section describes the initial list of IS adapters that will be developed by EOSC Beyond for Core and Horizontal services in the project's scope. This list is expected to evolve during the project lifetime according to the requirements gathered from the research communities.

2.4.1 Helpdesk

The helpdesk service is based on Zammad[8], an open-source and self-hosted ticketing system platform which provides feature-rich helpdesk solutions that can handle a wide variety of tasks, including ticket management, multi-channel submission, knowledge base creation, automation of workflows, reporting and analytics.

The current software version struggles with scalability and flexibility, limiting integration with other helpdesk technologies. The lack of modularity and complex configuration hinder efficient setup.

Proxy Standalone adapter

The helpdesk adapter enables seamless bidirectional synchronisation between two helpdesk platforms. It does this synchronisation as a standalone proxy that facilitates data flow without requiring direct integration with the platforms themselves. This approach is essential, as it provides flexibility and allows organisations to perform minimal development effort on their systems. Through its REST API, it enables real-time exchange of tickets, user data, and status updates, ensuring efficient and instant communication across platforms.

The adapter will simplify the integration process, and broaden its compatibility across various helpdesk technologies. Key enhancements will include:

- **Graphical Workflow Construction:** Enables users to visually design integration workflows, simplifying configuration and customization.
- **Simplified Field Mapping and Configuration:** Makes mapping setup between helpdesk systems faster

[8] https://zammad.org/

- **Modular Architecture:** Allows easier extension and maintenance, enabling support for future integrations.
- **Support for Multiple Helpdesk Technologies:** the prerequisite for other helpdesk technologies is the support of REST API.

2.4.2 PID Service

The Persistent Identifiers (PID) Service provides a reliable infrastructure for managing and maintaining persistent identifiers. The PID service is based on B2HANDLE[9], which provides a distributed service for storing, managing and accessing persistent identifiers and essential metadata (PID records). The implementation of the service relies on the DONA/Handle[10] persistent identifier solution. It can be used by middleware applications, end-user tools and other services to reliably identify data objects over longer time spans and through changes in object location or ownership. The service encompasses utilisation of identifier namespaces (Handle prefixes), establishment of policies and business workflows, operation of Handle servers and technical services. At the same time a user-friendly Python and/or Java library for general interaction with Handle servers is available. The service is transparent to end-users, shielding them from the complexity of infrastructure details. The service has a library written in python with the main functionalities offered by.

The PID service offers a well-documented API with functionalities that support the full lifecycle of data management, along with libraries to facilitate integration. At the same time it will ensure that PIDs conform to the ecosystem standards, to community standards and use appropriate data exchange mechanisms, considering metadata standards.The EOSC PID Adapter will use the library offered by B2HANDLE service and extend pyhandle functionalities where needed. The development of the EOSC PID adapter for the PID service involves the use of an interface that enables seamless communication between the PID service (via pyhandle) and components, such as catalogues, using standardised protocols and APIs. In order to create a usable adapter, different use cases will be followed to create a list of requirements. This ensures reliable identification, resolution and management of research outputs. By supporting operations like PID creation, updating and resolution, the adapter enhances the discoverability and accessibility of research resources within the EOSC ecosystem, promoting efficient data management and reuse.

2.4.3 Service Catalogue

This component offers the underlying storage functionality and interoperability tools for the programmatic access, registration, and management (CRUD) of providers, services, and catalogues. It also provides the necessary API functionality for the interoperability of service catalogues from individual providers or aggregators (e.g., thematic, or regional catalogues) with the EOSC portal. Modifications to the entities maintained in the Service Catalogue are synchronised with the EOSC Research Graph, which aggregates the complete set of entities and their relationships in the platform.

[9] https://eudat.eu/service-catalogue/b2handle
[10] https://www.dona.net/

GAP

Integration with the Service Catalogue is very important for both EOSC Core Components (such as Marketplace, Monitoring, and Helpdesk) and external Regional or Thematic EOSC Nodes(such as NI4OS and CESSDA). In both cases, integration proved to be a common obstacle, in terms that integrators had to write code to use the offered API (and then test it to make sure everything works).

Moreover, minor changes in the API meant also changes from the side of the other Components or Catalogues. This was very common for the communication of the Service Catalogue and the Marketplace, but also with the Regional and Thematic Nodes when they onboarded or updated their services en masse. Introductions of adapters as wrappers for the already offered API will minimise implementation errors for the aforementioned integrations.

Service Discovery adapter

Will provide functionality for:

- Searching, browsing, and navigating the Services and Providers which are onboarded to a Service Catalogue instance
- Ability to broker/route metadata information to service providers and any other EOSC Core component that uses such data

Provider and Service Management adapter

Will provide functionality for:

- Providers, to register to the EOSC to become eligible for the onboarding of resources
- Providers, to onboard their services into the EOSC Service Catalogue
- Providers, to view the list of services registered in the EOSC portal and perform a variety of actions such as activate, deactivate, view usage statistics

Administration and Audit adapter

Will provide functionality for:

- Managing the onboarding process (approve, reject an application),
- Managing the catalogue of providers and services and audit the validity of the catalogue entries
- Interacting with providers through EOSC established channels (Helpdesk)
- Adding catalogues from regional or thematic Nodes

Insights and Statistics adapter

Will provide functionality for:

- Catalogue statistics
- Recommendations and insights based on the catalogue feedback and content

2.4.4 Interoperability Framework Registry

This component serves as a registry for Interoperability Framework Guidelines onboarded by providers. It currently offers an API for providers to register and manage Interoperability Framework Guidelines linked to resources onboarded to the Service Catalogue.

GAP

Interoperability Registry and Service Catalogue share a common API, i.e. they are different facets of the same backend implementation/system. As such, Interoperability Registry adapters try to bridge the same gap observed in the Service Catalogue; integrators have to write code to use the offered API (and then test it to make sure everything works). As in Service Catalogue, introductions of the above adapters as wrappers for the already offered API will minimise implementation errors and speed up for the aforementioned integrations.

Interoperability Framework Guidelines Discovery adapter

Will provide functionality for:

- searching, browsing, and navigating the Interoperability Framework Guidelines which are onboarded to a Interoperability Registry instance
- ability to broker/route metadata information to service providers and any other EOSC Core component that uses such data

Interoperability Guideline Management adapter

Will provide functionality for:

- providers, to register to the EOSC to become eligible for the onboarding of Interoperability Framework Guidelines
- providers, to onboard their Guidelines into the EOSC Interoperability Registry
- providers, to view the list of Guidelines registered in the EOSC portal and perform a variety of actions such as activate, deactivate, update newer versions, and view statistics of Guidelines referenced by services.

2.4.5 Monitoring

Monitoring is the key service needed to gain insights into an infrastructure. It needs to be continuous and on-demand to quickly detect, correlate, and analyse data for a fast reaction to anomalous behaviour before they affect end-users and ultimately the productivity of the organisation. Management teams can monitor the availability and reliability of the services from a high level view down to individual system metrics and monitor the conformance of multiple SLAs. The service is currently accessible only via an API, requiring each client to develop custom integrations to interact with it. This can be both time-consuming and complex, as developers must build tailored solutions for various use cases. To simplify this process, a Python-based adapter will be introduced. This adapter will act as a wrapper around the existing API, making it easier to connect to the Argo Monitoring Service by

managing the underlying API calls. With this adapter, developers can more easily integrate and automate the collection and exchange of metrics.

Key goals of the adapter include:

- **Streamlined Integrations**: It will focus on frequently used API calls, such as publishing metrics, to simplify the integration process.
- **Simplified Interface**: Complex API calls will be abstracted into easy-to-use Python functions, with common parameters automatically handled.
- **Error Handling**: It will centralise error management, converting raw API errors into meaningful Python exceptions and incorporating retry logic to enhance reliability.

2.4.6 Messaging

The ARGO Messaging Service (AMS) is a robust, real-time messaging platform designed to facilitate seamless communication between independent applications by enabling the exchange of messages. It acts as a middleware service, allowing different components, services, or applications to send and receive messages efficiently, even if they are developed independently or reside on different systems. This makes AMS ideal for microservices architectures, distributed systems, and other scenarios where multiple applications need to communicate reliably and at scale.

Currently, AMS offers a python library that provides the core functionalities of the service such as message publishing, subscribing, and queue management. However, to meet the evolving needs of various services within the project, there is a requirement to enhance this library by adding new features and extending existing ones. The project aims to build on the existing capabilities of the AMS library by introducing new features such as advanced message filtering, support for different message formats, and improved error handling.

The adapter's enhancements will be:

- **Extended Coverage**: within the scope of this project, we will expand the coverage of existing functionalities to improve the overall user experience.
- **Support for New Functionalities**: we will add support for new features to be developed, enhancing the adapter's capabilities and adaptability to various client requirements.

Some of the extra capabilities:

- **Batch Processing**: If the other services require, you could extend the AMS library to support batch message processing.
- **Custom Message Formats**: add support for custom message formats based on the schema support.

2.4.7 Accounting for Services

The Accounting Service is a robust platform designed to streamline metrics collection, aggregation, and exchange across diverse infrastructures, providers, and projects

Currently, the service is accessible only through an API, which implies that each client must develop its own integration to interact with. This approach can be time-consuming and complex, as developers need to build custom solutions for every use case. To address this challenge, the service will be enhanced with the introduction of a Python-based adapter. This adapter will serve as a wrapper around the existing API, simplifying the process of connecting to the Accounting Service. By handling the underlying API calls, the adapter will make it easier for developers to integrate and automate the collection and exchange of metrics.

The initial version of the adapter will focus on the most commonly used functionalities, including publishing and retrieving accounting metrics. This will streamline the process of building Accounting agents for various services, reducing development time and effort.

The adapter's aims will be:

- **Focus on the most frequently used API calls** such as metrics publishing aiming to simplify the creation of clients.
- **Simplified Interface**: It will abstract complex API calls into easy-to-use Python functions, automatically handling common parameters to simplify usage.
- **Error Handling**: It will centralise error handling by converting raw API errors into meaningful Python exceptions and it will include retry logic to improve reliability.

2.4.8 Research Product Catalogue

The Research Product Catalogue is designed to provide the necessary infrastructure for storing, managing, and ensuring the discoverability of various research outputs, such as datasets, publications, software, and more. This catalogue enables programmatic access and supports the registration and management of research products, allowing researchers and institutions to easily contribute their outputs to the broader EOSC ecosystem. The catalogue also ensures compliance with the FAIR (Findable, Accessible, Interoperable, and Reusable) principles, promoting open science and data sharing.

Research Product Discovery adapter

This adapter will provide functionality for searching, browsing, and navigating research products within the catalogue. Users will be able to find products based on various criteria such as subject area, licence type, or specific research interests.

Additionally, this adapter will enable metadata brokering between the Research Product Catalogue and other EOSC components, improving the interoperability of research products across platforms.

OAI-PMH adapter

This adapter will enable the Research Product Catalogue to interact with external repositories and data providers using the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). By implementing OAI-PMH support, the catalogue can both harvest metadata from various sources and expose its own metadata for harvesting by others. This enhances the interoperability and discoverability of research outputs across the EOSC ecosystem. The adapter will facilitate automated synchronisation of metadata records, ensuring that the catalogue remains up-to-date with the latest research products while adhering to international metadata standards and the FAIR principles.

2.4.9 Research Product Accounting

The Research Product Accounting service[11] is essential for tracking the usage and impact of the research products catalogued in the EOSC ecosystem. This service offers metrics on how frequently research products are accessed, downloaded, cited, and reused. By providing detailed insights, the service helps institutions and researchers assess the effectiveness of their data-sharing efforts and compliance with open science mandates.

Accounting of Research Products adapter

To benefit from this service, the data source has to be registered in OpenAIRE through the Provider Dashboard[12] for data source providers. Preliminary registration steps consist of accessing the service and via, the UsageCounts menu, obtain two identifiers crucial for the unique identification of the data source and its events for accounting:

- a Matomo-ID that associates the data source with its usage events in the Matomo platform;
- an authentication-ID that allows to track usage activity on the Matomo platform.

Such identifiers will have to be used to configure the adapters required to connect a given data source to the UsageCounts infrastructure. The adapters, which can be downloaded from GitHub, are provided for two main repository platforms (as extensions of the repository software) and one for all other cases:

- as a patch for various versions of DSpace[13]
- as an Eprints plugin[14] for version 3
- as a Python script[15] for all other cases

The GitHub repositories contain a README file that provides instructions on how to configure the Matomo identifiers and how to deploy the adapters.

[11] https://www.openaire.eu/guides-usage-counts
[12] https://provide.openaire.eu/
[13] https://github.com/openaire/OpenAIRE-Piwik-DSpace
[14] https://github.com/openaire/EPrints-OAPiwik
[15] https://github.com/openaire/Generic-Matomo-Tracker

Accounting and Metrics adapter

This adapter will track usage statistics, including downloads, citations, views, and instances of reuse. It will also report compliance with open-access and data-sharing mandates, helping researchers monitor the visibility and influence of their outputs.

The Accounting and Metrics Adapter will offer comprehensive reports, allowing users to gain insights into how their research products are being utilised within the scientific community.

Insights and Statistics adapter

This adapter will provide detailed analytics and feedback on research product usage, offering insights into how the outputs are reused or cited. The data collected will help researchers and institutions refine their dissemination strategies and ensure alignment with open science principles.

This adapter will present recommendations based on the gathered statistics, helping providers optimise their research product offerings and improve accessibility for the scientific community.

2.4.10 EOSC Data Transfer

Data Transfer[16] is a horizontal EOSC service, consisting of an API and a GUI, which is integrated with the EOSC Marketplace. Although the main client of this service is the EOSC Marketplace, its API can also be used directly by other EOSC services (e.g. the Deployment Service) and researchers. To facilitate this direct usage, adapters shall be developed for the Data Transfer API. Java and Python adapters are proposed.

GAP

The adapters for this service shall be linked to the Data Transfer IF Guidelines[17] and the service listing in the services catalogue, facilitating the findability of these resources that can make it easier for clients to use the service.

2.4.11 EOSC Deployment Service

The EOSC Beyond Deployment Service (DS) is designed for performing the deployment of complex application architectures across multiple Cloud sites, with enhanced functionality of advanced integration and composability of data sources, software tools and computing infrastructures.

JupyterLab libraries for service provisioning and data staging

This adapter is responsible for providing extensions in Jupyter that allow users to employ the functionality of the DS from a Jupyter Notebook. The goal is to facilitate the creation of reproducible experiments which include in the narrative of the experiment the (1) provisioning of the virtualized computing infrastructure required to reproduce the experiment, as a TOSCA template deployable through the DS, and (2) the staging of

[16] https://github.com/egi-federation/eosc-data-transfer
[17] https://doi.org/10.5281/zenodo.7925515

dataset(s) into storage allocated in step (1), from any supported open data repository, such as Zenodo or B2SHARE, or even from relevant data spaces.

This adapter may be used to introduce this support in existing managed Jupyter installations such as the EGI Notebooks service.

Metadata gathering of TOSCA templates

This adapter will facilitate the access of TOSCA template metadata offered by the DS through its OAI-PMH interface. These can be a curated set of TOSCA templates that are commonly used by the users of an EOSC Node. This will facilitate automated resource discovery and facilitate integration with third-party services in the EOSC Exchange that require the functionality of deploying virtualized infrastructure.

2.4.12 Data access across hosting environments

In addition to accessing and downloading remote data from public repositories as described above, users will also want to access their working data wherever they execute their Jupyter notebooks. The solution to the problem in general is typically involving sync-and-share services such as NextCloud files. In some environments, the accessibility of large-scale data, typically accommodated in HPC (file) systems, may also be desirable. At JSC, both integrations have been put in place, such that users can work with their data regardless of the exact environment they work in, be it Cloud or HPC.

2.4.13 Python libraries for EOSC core and exchange services

FedCloud client is a high-level Python package for an infrastructure agnostic command-line client (CLI) designed for interaction with the OpenStack services in the EGI infrastructure. The client can access various EGI services and can perform many tasks for users including managing access tokens, listing services, and mainly execute commands on OpenStack sites in the cloud computing infrastructure of the EGI Federation.

An EOSC node currently could not benefit from features implemented in the FedCloud client, even though principal Python classes that can use EGI services: AAI (EGI Checkin), service catalogue (GOCDB), Secrets Store, and OpenStack sites providing VM-based computing services are implemented.

Capitalising on the fact that EGI services are similar to some of the EOSC core and exchange services of an EOSC node, the tool will be extended so that it can be re-configured from operating on these services in the EGI Federation, to operate on the equivalent services in an EOSC node.

Furthermore, the FedCloud client's capabilities to provision storage in OpenStack sites will be extended with the ability to immediately stage datasets into the provisioned storage, by leveraging the procedures used for this regard by the Deployment Service.

3. Execution Framework

Most data-driven applications and scientific workflows are composed of multiple data resources, processing services, and other research objects. Researchers typically compose those environments manually — in EOSC, we are in a position where the set up of such environments can be automated, hence simplifying the process to our users.

The EOSC Execution Framework (EF) will provide the technical solutions required to automatically deploy complex applications, composed of multiple services and data resources.

The EF is the next stage of development after the EOSC Interoperability Framework (EOSC IF) [3]. EOSC IF defined qualitative aspects of services interoperability and materialised them through the Interoperability Framework Registry (IF Registry) and the Interoperability Framework Guidelines (IGs). In EOSC Beyond we will extend that Framework to allow machine-composability of interoperable resources.

To accomplish that, besides improving the Interoperability Framework to allow automation (ie, machine-understandable resources), we need to define a place where such automation happens – the engine that will exploit the capabilities of the improved IF Registry to automatically deploy composed services and applications. This is the role of the new Deployment Service.

3.1. Gap Analysis to support the Execution Framework

The list of EOSC Core Services coming from the EOSC Future project does not include an Execution Framework that features advanced integration and composability of data sources, software tools and computing infrastructures, which is important for two reasons: (1) to improve productivity of our researchers by removing unnecessary software setup – especially for those not trained to do so; (2) to provide reproducible environments – a fundamental and necessary aspect of science.

Below, we expose the gaps and requirements of the two principal components of the Execution Framework: the EOSC Interoperability Framework Registry (that needs to associate configurations to IGs to provide machine-readable instructions to instantiate services), and the Deployment Service (to deploy services on demand on top of hybrid Cloud infrastructures).

3.1.1 Interoperability Framework Registry

The EOSC Interoperability Framework (IF) Registry, a key component of the EOSC Platform, provides tools for describing the interoperability capabilities of resources (IF Guidelines), and to annotate EOSC resources with the guidelines they adhere to. This enables an initial exploration of resources based on their interoperability features, resulting in a basic assessment of the compatibility and composability capabilities of an EOSC resource.

Currently, the EOSC IF Registry is a collection of guidelines, which are human-readable instructions that EOSC Providers must implement to enable the desired functionality and interoperability between their EOSC services and/or research products. These guidelines may reference various existing standards, protocols, processes, in a manner that allows Providers to understand how to design and configure services to be part of the EOSC ecosystem. Although the IGs are helpful to humans, they do not allow for automated deployment or metadata consistency checks.

To overcome this, the enhanced EOSC IF Registry will evolve to support the machine-composability of resources (REQ.IF.4). To achieve this, the IF Registry will manage IGs configurations. These are structured metadata profiles that allow providers to detail the actual access parameters of their services according to the guidelines (REQ.IF.5); The EOSC Service Catalogue will be modified to accommodate service configurations related to specific IF Guidelines; The EOSC Service Provider Dashboard will be linked with the IF Registry, enabling providers to input their service configurations (REQ.IF.6); The IF Registry will be connected with Marketplace UIs to access, from a service profile, the associated IF Guidelines and related software libraries (REQ.IF.7).

3.1.2 Deployment Service

Deploying cloud infrastructure or a software service is a complex task that typically involves different sets of skills in the administration of systems as well as specific knowledge on the software being deployed. Particularly in the case of EOSC, where software services are used for applications across different disciplines, on different platforms, and by researchers , it is important to provide those elements – infrastructure plus software – as a service.

While there had been efforts in providing a distributed computing environment and standardised service components as a service (*e.g.*, EOSC Compute Platform Services[18]), there remains a gap in ensuring the seamless deployment of applications across a broad range of providers. To answer to this need, the EOSC EU Node has defined, among several functional service components, the Application Workflow Management (AWM), a service for the composition and execution of complex application workflows across a wide variety of providers and services.

In EOSC Beyond we plan to design the Deployment Service (DS), based on the Infrastructure Manager (IM)[19] tool, to extend and be technically compatible with the Application Manager Layer (AML) of the EOSC EU Node AWM service. AML is responsible for performing the deployment of complex application architectures across multiple Cloud sites. The DS will enhance the functionalities of the AWM to achieve the project vision of an automated integration and composability of data sources, software tools and computing infrastructures. The compatibility with the EU Node AWM service will foster potential retrofitting of the capabilities designed in EOSC Beyond in other EOSC initiatives, including, but not limited to, the EOSC EU Node. In line with these considerations, the Deployment

[18] https://www.egi.eu/eosc-compute-platform-services/

[19] https://im.egi.eu/

Service must support the TOSCA specification for application descriptions that is the standard language adopted by AWM to describe cloud topologies (REQ.DS.1).

The DS needs to be accessible from multiple interfaces to target both users and systems and service integrators (including EOSC Node operators). It is expected to provide a CLI (command-line interface), a REST API (Application Programming Interface), a web-based UI, and also be accessible from Jupyter Notebooks (REQ.DS.2). The latter is required to facilitate the creation of reproducible computational environments that include not only the precise hardware and software configuration but also the datasets to perform reproducible experiments in the EOSC.

As a key enhancement with respect to the EU Node AWM service, the DS will also support the deployment of datasets close to the compute services that will be used to process them. To achieve this objective, the TOSCA standard needs to be extended to allow to describe the datasets (REQ.DS.3), and the Deployment Service should support data staging capabilities to fetch the datasets from multiple repositories and make them available to the dynamically provisioned virtual infrastructure (REQ.DS.4).

The DS requires the addition to the EOSC IF Registry of *Guidelines for TOSCA deployable services* (REQ.DS.8), which denote services that are onboarded in the Service Catalogue together with a deployable version. For such services a dedicated configuration template would need to be filled in with the parameters needed by TOSCA to fetch the code and deploy the service. Furthermore, to securely access the services deployed, the Deployment Service should be integrated with the Dynamic DNS service to register and assign fully qualified and secure domain names to them (REQ.DS.5).

The EOSC Marketplace will enable end-users to find TOSCA-compliant deployable services of interest and, on request and input of the user, collect the computing platform requirements and fire their deployment on the DS (REQ.DS.6).

A DS service instance may be configured to support the deployment of a subset of curated TOSCA templates to satisfy certain use cases. Considering that each EOSC Node may have a deployment of their own DS, it is important to provide automated resource discovery of the metadata of each TOSCA template. This will be done supporting the OAI-PMH protocol to allow external harvesters to collect the metadata about the TOSCA templates supported by a certain instance of the DS (REQ.DS.7).

3.2. Requirements to support Execution Framework

Table 2 below captures the EF-related requirements. This includes software improvements to the EOSC Interoperability Framework and Deployment Service:

Table 2 - Requirements table for Execution Framework

Service	ID	Description
Interoperability Framework	**REQ.IF.4**	IF Registry must evolve to support machine-composability of resources.
	REQ.IF.5	IF Registry should manage IF Guidelines configurations and structured metadata profiles that allow providers to detail the actual access parameters of their services according to the guidelines.
	REQ.IF.6	IF Registry to enable providers to input their service configurations.
	REQ.IF.7	Enable Service-Guideline-Adapter relationships queries.
Deployment Service	**REQ.DS.1**	The DS must use the TOSCA specification for application descriptions.
	REQ.DS.2	The DS should be accessible from multiple interfaces including CLI, API, Web-based UI and from Jupyter Notebooks.
	REQ.DS.3	The TOSCA specification must be extended to support the definition of data sources alongside the application description required to process it.
	REQ.DS.4	The DS must use automated data staging capabilities to fetch the data sources from multiple open repositories and make it available to the dynamically provisioned virtual infrastructure.
	REQ.DS.5	The DS should be integrated with the Dynamic DNS service to register and assign fully qualified and secure domain names to the deployed services.
	REQ.DS.6	The DS must allow the EOSC Marketplace to delegate user

Service	ID	Description
		application deployment requests to it.
	REQ.DS.7	The DS must support the OAI-PMH protocol to allow automated harvesting of the metadata of the TOSCA templates supported by the DS.
	REQ.DS.8	Include Guidelines for TOSCA-deployable services in IF Registry

3.3. Execution Framework Service Design

The diagram below (Figure 2) provides a high-level overview of the service workflow and interaction within the Execution Framework. It outlines the interactions between the key components: Interoperability Framework Registry and Deployment Service (at the core of EF); Marketplace, Resource Catalogue, and EOSC cloud (infrastructure) resources.

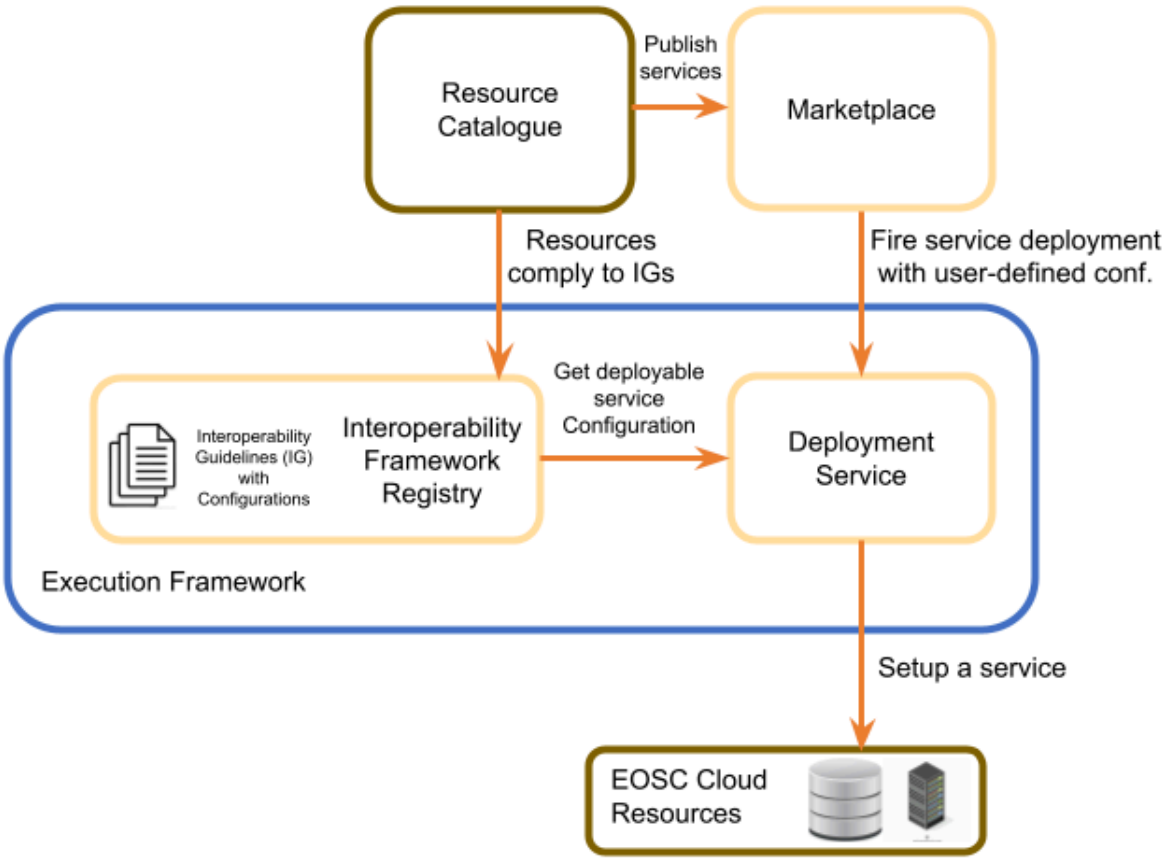

Figure 2 - EOSC Beyond Execution Framework architecture diagram

3.3.1 Components

- **Resource Catalogue:** manages a collection of interoperable EOSC resources.
- **Marketplace:** provides the user interface for discovering and selecting composable services, and facilitates user-defined configuration parameters to DS.
- **IF Registry:** manages IGs and corresponding configurations, ensuring services are deployed in a machine-composable manner.
- **Deployment Service:** the core component responsible for orchestrating the deployment of services requested by Marketplace using configurations from IF Registry.
- **EOSC Cloud Resources:** the infrastructure where the services/data are deployed.

3.3.2 Services Workflow and Interaction

1. **Resource Publication:** services and data products are published to the Resource Catalogue – which ensures that they comply with the Interoperability Framework Guidelines.
2. **Service Discovery:** users browse and select resources via the Marketplace, and define specific settings for that particular deployment (e.g. where to store results).
3. **Configuration Retrieval:** the Deployment Service retrieves the necessary service configurations from the Interoperability Framework Registry – this ensures that the selected services/resources are deployed in accordance with the IGs.
4. **Service Setup:** finally, the Deployment Service sets up the selected services and data products on EOSC infrastructure and handles access to the requester user.

3.3.3 Key Considerations

- Interoperability: ensures that resources comply with established guidelines, facilitating seamless integration and operation.
- Automation: automated setup of services minimises manual intervention, reducing complexity, time, and failures.
- User-Centric: Marketplace provides a unique and intuitive interface for users to choose, customise, and deploy (composed) applications.

3.3.4 Considerations on Composability

Two composability patterns are supported by the EF, enabling compatibility across both services and products:

- **Product-Service Composability**
  Compatibility: Identifies compatible services for specific research products. For example, a researcher interested in analysing Twitter data can find relevant/capable EOSC Services for their processing, or a scholar studying cultural heritage texts can locate compatible processing services.
  Machine Composability: Supports automated invocation of services by other services. For instance, an orchestrator service allows researchers to (i) discover repository services for data storage and (ii) deposit datasets without directly

interacting with the repository's web interface.

- **Service-Service Composability**
  Compatibility: Identifies services that could be combined in research settings. For example, a researcher that has developed a Galaxy service wants to discover other Galaxy compatible services capable of performing a given functionality.
  Machine Composability: Allows services to be automatically pipelined as workflow steps, for reproducible research workflows.

3.4. Composability-based Applications

EOSC Beyond has proposed three typical research application scenarios to serve (i) as exemplary applications and (ii) guide and test the development of the Execution Framework. The applications demand the composition of multiple services and data resources, and prove the concept around the Execution Framework from a practical perspective.

3.4.1 Scholarly Communication Metadata Aggregator

A typical task in data science is data mining, the process of sorting through large datasets to identify patterns and relationships through multiple data records. In this workflow, the user will (i) select a set of data repositories, (ii) define the metadata fields of interest from each repository, (iii) collect the values associated with the fields and record that in some local data storage for subsequent analysis. In data engineering terms, this is known as the Extract-Transform-Load (ETL) pipeline.

In the context of EOSC, the datasets are provided by OAI-PMH compatible repositories (e.g., Zenodo) providing stable metadata schema that can be programmatically queried, and the results are data files containing the aggregated metadata records for ease of use.

Specifically, this application is to provide an interface for the user to:

1. Select repositories supporting OAI-PMH protocol IF-Guidelines in EOSC-Exchange;
2. Select the metadata fields from each repository;
3. Define the destination storage for the records;
4. Submit the job request.

After that, the workflow service harvests the metadata from each and all repository endpoints, package the results, and return an access point (storage location) for the user to download the data.

3.4.2 Data Transfers

Since its first release, EOSC has been embraced by scientific community members because it allowed researchers to discover research products (datasets, papers, etc.) relevant for their scientific work, by using the EOSC Marketplace to browse and search for these inputs. In case of datasets, these users also needed a way to transfer the identified datasets, as a whole or just parts of it, to their research environment, most of the time motivated by the necessity for the data to be close to the compute resources available to these researchers.

These requirements were satisfied by the EOSC Data Transfer horizontal service delivered by the EOSC Future project, including the integration of the Data Transfer GUI into the EOSC Marketplace.

To make the Data Transfer service participate in the composability-based paradigm of the EOSC Execution Framework, and to allow full exploitation of the EOSC EF capabilities, the EOSC Execution Framework has to be able to take advantage of this horizontal service to perform data transfers, as needed.

Proposed workflow:

1. The user finds a product of interest via the EOSC Marketplace
2. The user selects the Data Transfer option and specifies the type of destination (e.g. FTP endpoint) and the relative parameters
3. Based on the location, protocol, type, and the access protocol of the product's data source, the EOSC DTS discovers Transfer Services in the EOSC Exchange and prompts the user with a choice;
4. The user selects the DTS of preference and fires the action.

3.4.3 Data Analysis Environment

A substantial amount of time in data-driven research is spent in exploring the data, visualising data, extracting features, and testing models. Known as Exploratory Data Analysis (EDA), this is a fundamental step in data science workflows and demands interactive operations towards the dataset at hand.

In such a scenario, we want to have data and analysis tools close to each other. Moreover, we need an environment where temporary data – working data – can be created and deleted as part of the process. Meaning, we want to deploy data and tools in a custom environment where the user is capable of modifying its assets.

In the context of EOSC, this application encompasses the following steps:

1. The user finds a product of interest in the Marketplace, which recommends the usage of a deployable service for its processing/analysis;
2. The user can configure via UI an environment where the data can be processed, and can set its access-point domain;
3. The user submits the environment deployment request.

At this point, the system takes care of staging the requested dataset into the allocated storage as well as deploying the necessary tools, eventually returning to the user the point of access to the application environment.

3.5. General workflow of data-driven applications in EOSC

The three applications highlighted above are specific cases of a generic scenario where the user wants (1) to process a dataset available in any public open data repository (2) with an

specific processing/analysis tool that will be made available as a deployable service (3) that will be deployed in available EOSC Cloud resources. This general scenario is depicted in Figure 3 below, summarising the workflow of such applications in the context of EOSC.

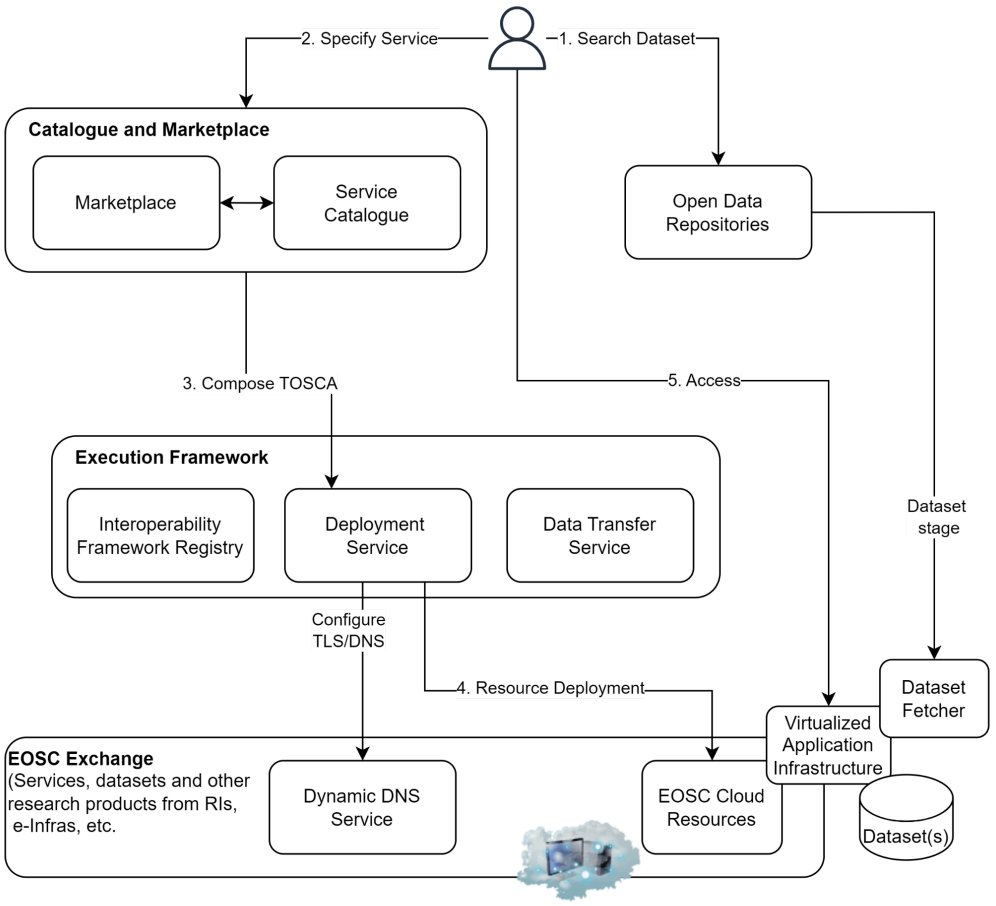

Figure 3 - Architecture of the Integration among the Components of the Execution Framework

Specifically, the following steps are involved in the process of find and deploying resources depicted in the figure:

1. **(User) Search Datasets:** The user initiates a search for datasets within open data repositories such as OSF, Zenodo, GitHub, and Dataverse.
2. **(User) Specify Services:** the user then selects/defines the appropriate service(s) to work on the chosen dataset(s) to deploy in a specified infrastructure using the Marketplace as Interface.
3. **(EF) Compose TOSCA:** The marketplace finds the TOSCA template associated with the specified service and adds the datasets to the TOSCA template that will be finally delegated to the Deployment Service.
4. **(EF) Deploy Resources:** The Deployment Service deploys the necessary environment to work with the selected dataset. This involves provisioning the required computational resources as well as staging the specified datasets in the deployed infrastructure. The deployment service is also responsible for configuring the necessary TLS/DNS settings to ensure secure and properly routed access to the deployed environment.

5. **(User) Access Resources:** Finally, the user can access the dataset(s) through the Virtualized Infrastructure, which now contains the staged datasets ready for analysis or other tasks.

To achieve the functionality the Deployment Service will support dataset staging within the provisioned virtualized infrastructure. This requires resolving PIDs (Persistent IDentifiers) for datasets (e.g., DOIs from repositories like Zenodo) and automatically retrieve and stage these datasets in the dynamically deployed infrastructure. Leveraging libraries such as DataHugger[20], this functionality will be integrated with Ansible Roles and TOSCA templates managed by the IM.

Additionally, the Deployment Service will interact with the Dynamic DNS Service to assign DNS entries during deployment, and automatically request TLS certificates, thus ensuring secure access (e.g., HTTPS) for deployed services without external dependencies such as Let's Encrypt. This automation minimises user intervention and provides secure, ready-to-use infrastructure for data-driven applications.

[20] https://j535d165.github.io/datahugger/

4. Conclusion

Figure 4 displays the Integration Suite and the Execution Framework in one architecture diagram, showing how the two services are expected to collaboratively enable services and resources composability in EOSC. The development of new components and implementation of necessary modifications will be carried out by WP11 (for the Integration Suite) and WP13 (for the Execution Framework).

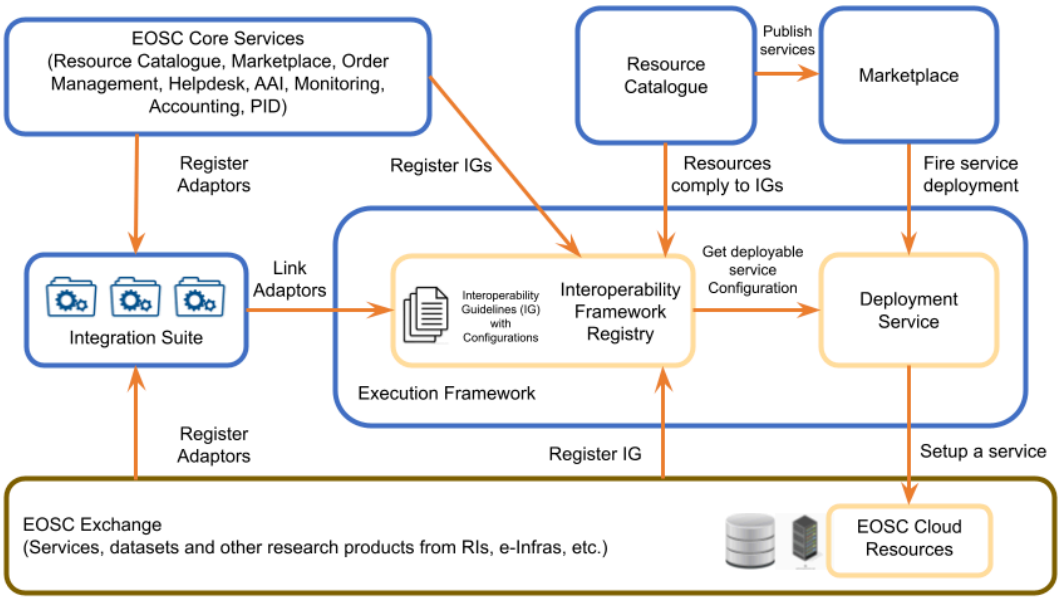

Figure 4 - EOSC Beyond Integration Suite and Execution Framework architecture

The development of the EOSC Beyond Integration Suite and Execution Framework represents a significant step forward in the automation and interoperability of data-driven applications within the EOSC ecosystem. By addressing the current gaps in service integration and deployment, the project paves the way for a more streamlined, scalable, and user-friendly environment that simplifies the onboarding process for service providers and enhances accessibility for users.

Through the introduction of standardised adapters, the Integration Suite enables seamless alignment with the Interoperability Framework Guidelines, reducing the complexity and time required for external providers to integrate their services. This standardised approach not only fosters a cohesive framework for interoperability but also ensures a consistent quality of integration across diverse services and infrastructures.

The Execution Framework complements this by providing a robust, automated deployment environment, which, in conjunction with the enhanced Interoperability Framework, enables the machine-composability of interoperable resources. The Deployment Service, as a core component of this framework, leverages the IF Registry to automate complex application setups, offering significant efficiency gains and reducing the manual effort required for multi-service orchestration.

In conclusion, the EOSC Beyond project establishes a scalable, automated, and interoperable foundation that is essential for supporting the growing demands of data-intensive research in the European Open Science Cloud. By continuing to refine and expand these tools, EOSC Beyond will provide a robust infrastructure that simplifies and accelerates scientific workflows, ultimately supporting a more collaborative and efficient research environment.

Acronyms

Acronym	Description
API	Application Programming Interface
DS	Deployment Service
EF	Execution Framework
IF	Interoperability Framework
IFG or IG	Interoperability Framework Guideline
IS	Integration Suite
JSON	JavaScript Object Notation
TOSCA	Topology and Open Specification for Cloud Applications
WP	Work Package

References

[1] Michelle Williams (GEANT), Mark Van de Sanden (SURF), Diego Scardaci (EGI), Paolo Manghi (OpenAIRE), Licia Florio (NORDUnet), Klaas Wierenga (GÉANT) , 'EOSC Future D3.2b: EOSC Architecture and Interoperability Framework', Jun. 2023. [Online]. Available: https://eoscfuture.eu/wp-content/uploads/2024/03/EOSC-Future-WP3-GEANT-D3.2b-EOSC-Architecture-and-Interoperability-Framework-2023-06-20.pdf

[2] K. Lechowska-Winiarz, R. Wilk, and M. Gutiérrez, 'EOSC Beyond D15.1 Service integration plans of the Data Spaces and EOSC Nodes', Aug. 2024. [Online]. Available: https://zenodo.org/records/13152335

[3] D. Scardaci et al., 'A landscape overview of the EOSC Interoperability Framework - Capabilities and Gaps', Oct. 2023. [Online]. Available: https://zenodo.org/records/8399710